Copyright Information

Copyright © 2010 Internetwork Expert, Inc. All rights reserved.

The following publication, CCIE R&S Lab Workbook Volume I, was developed by Internetwork Expert, Inc. All rights reserved. No part of this publication may be reproduced or distributed in any form or by any means without the prior written permission of Internetwork Expert, Inc.

Cisco®, Cisco® Systems, CCIE, and Cisco Certified Internetwork Expert, are registered trademarks of Cisco® Systems, Inc. and/or its affiliates in the U.S. and certain countries.

All other products and company names are the trademarks, registered trademarks, and service marks of the respective owners. Throughout this manual, Internetwork Expert, Inc. has used its best efforts to distinguish proprietary trademarks from descriptive names by following the capitalization styles used by the manufacturer.

Copyright © 2010 Internetwork Expert

Disclaimer

The following publication, CCIE R&S Lab Workbook Volume I for CCIEv4.0, is designed to assist candidates in the preparation for Cisco Systems' CCIE Routing & Switching Lab Exam. While every effort has been made to ensure that all material is as complete and accurate as possible, the enclosed material is presented on an "as is" basis. Neither the authors nor Internetwork Expert, Inc. assume any liability or responsibility to any person or entity with respect to loss or damages incurred from the information contained in this workbook.

This workbook was developed by Internetwork Expert, Inc. and is an original work of the aforementioned authors. Any similarities between material presented in this workbook and actual CCIE lab material is completely coincidental.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Table of Contents

Workbook Diagrams 2

Bridging & Switching 6

1.1 Layer 2 Access Switchports 6
1.2 Layer 2 Dynamic Switchports 6
1.3 ISL Trunking 6
1.4 802.1q Trunking 6
1.5 802.1q Native VLAN 6
1.6 Disabling DTP Negotiation 7
1.7 Router-On-A-Stick 7
1.8 VTP 7
1.9 VTP Transparent 7
1.10 VTP Pruning 8
1.11 VTP Prune-Eligible List 8
1.12 Layer 2 EtherChannel 8
1.13 Layer 2 EtherChannel with PAgP 8
1.14 Layer 2 EtherChannel with LACP 8
1.15 Layer 3 EtherChannel 9
1.16 802.1q Tunneling 9
1.17 EtherChannel over 802.1q Tunneling 10
1.18 STP Root Bridge Election 10
1.19 STP Load Balancing with Port Cost 11
1.20 STP Load Balancing with Port Priority 11
1.21 Tuning STP Convergence Timers 11
1.22 STP PortFast 11
1.23 STP PortFast Default 11
1.24 STP UplinkFast 12
1.25 STP BackboneFast 12
1.26 STP BPDU Guard 12
1.27 STP BPDU Guard Default 12
1.28 STP BPDU Filter 12
1.29 STP BPDU Filter Default 13
1.30 STP Root Guard 13
1.31 STP Loop Guard 13
1.32 Unidirectional Link Detection 13
1.33 MST Root Bridge Election 14
1.34 MST Load Balancing with Port Cost 14
1.35 MST Load Balancing with Port Priority 14
1.36 MST and Rapid Spanning Tree 15
1.37 Protected Ports 15
1.38 Storm Control 15
1.39 MAC-Address Table Static Entries & Aging 15
1.40 SPAN 15

Copyright © 2010 Internetwork Expert

1.41 RSPAN. 16
1.42 Voice VLAN. 16
1.43 IP Phone Trust and CoS Extend. 16
1.44 Smartport Macros. 17
1.45 Flex Links. 17
1.46 Fallback Bridging. 17
1.47 Private VLANs. 18
1.48 PPP 18
1.49 PPP AAA Authentication. 19
1.50 PPPoE. 19

Frame Relay. 20
2.1 Inverse-ARP. 20
2.2 Static Mappings. 20
2.3 Point-to-Point Subinterfaces. 20
2.4 Multipoint Subinterfaces & Inverse-ARP. 21
2.5 Multipoint Subinterfaces & Static Mappings. 21
2.6 Hub-and-Spoke & Static Mappings. 21
2.7 Hub-and-Spoke & Inverse-ARP. 22
2.8 Hub-and-Spoke & Point-to-Point Subinterfaces. 22
2.9 Disabling Inverse-ARP. 23
2.10 Back-to-Back Frame Relay. 23
2.11 Frame Relay End-to-End Keepalives. 24
2.12 Frame Relay Broadcast Queue. 24
2.13 Frame Relay TCP & RTP Header Compression. 24
2.14 PPP over Frame Relay. 24
2.15 Bridging over Frame Relay. 25
2.16 Frame-Relay Switching. 25

IP Routing. 26
3.1 Routing to Multipoint Broadcast Interfaces. 26
3.2 Routing to NBMA Interfaces. 26
3.3 Longest Match Routing. 27
3.4 Floating Static Routes. 27
3.5 Backup Interface. 27
3.6 Reliable Static Routing with Enhanced Object Tracking. 28
3.7 Policy Routing. 28
3.8 Reliable Policy Routing. 29
3.9 Local Policy Routing. 30
3.10 GRE Tunneling. 30
3.11 GRE Tunneling and Recursive Routing. 30
3.12 Reliable Backup Interface with GRE. 31
3.13 On-Demand Routing (ODR). 31
3.14 OER Components Setup. 33
3.15 OER Profile Phase. 33
3.16 OER Measure Phase. 33
3.17 OER Apply Policy Phase. 34

Copyright © 2010 Internetwork Expert

3.18 OER Control & Verify Phase 34

RIP 36

4.1 Basic RIP Configuration 36
4.2 RIPv2 Authentication 36
4.3 RIPv2 Split Horizon 36
4.4 RIPv2 Auto-Summary 36
4.5 RIP Send and Receive Versions 37
4.6 RIPv2 Manual Summarization 37
4.7 RIPv2 Convergence Timers 37
4.8 RIPv2 Offset List 37
4.9 RIPv2 Filtering with Passive Interface 37
4.10 RIPv2 Filtering with Prefix-Lists 37
4.11 RIPv2 Filtering with Standard Access-Lists 38
4.12 RIPv2 Filtering with Extended Access-Lists 38
4.13 RIPv2 Filtering with Offset Lists 38
4.14 RIPv2 Filtering with Administrative Distance 38
4.15 RIPv2 Filtering with Per Neighbor AD 38
4.16 RIPv2 Default Routing 38
4.17 RIPv2 Conditional Default Routing 38
4.18 RIPv2 Reliable Conditional Default Routing 39
4.19 RIPv2 Unicast Updates 39
4.20 RIPv2 Broadcast Updates 39
4.21 RIPv2 Triggered Updates 39
4.22 RIPv2 Source Validation 39

EIGRP 40

5.1 EIGRP Network Statement 40
5.2 EIGRP Auto-Summary 40
5.3 EIGRP Split Horizon 40
5.4 EIGRP MD5 Authentication 40
5.5 EIGRP Key Chain Rotation 41
5.6 EIGRP Unicast Updates 41
5.7 EIGRP Default Network 41
5.8 EIGRP Summarization 41
5.9 EIGRP Summarization with Default Routing 42
5.10 EIGRP Summarization with Leak Map 42
5.11 EIGRP Floating Summarization 42
5.12 EIGRP Poisoned Floating Summarization 42
5.13 EIGRP Metric Weights 43
5.14 EIGRP Traffic Engineering with Metric 43
5.15 EIGRP Unequal Cost Load Balancing 43
5.16 EIGRP Convergence Timers 43
5.17 EIGRP Stub Routing 43
5.18 EIGRP Stub Routing with Leak Map 44
5.19 EIGRP Filtering with Passive Interface 44
5.20 EIGRP Filtering with Prefix-Lists 44

Copyright © 2010 Internetwork Expert

5.21 EIGRP Filtering with Standard Access-Lists 44
5.22 EIGRP Filtering with Extended Access-Lists 44
5.23 EIGRP Filtering with Offset Lists 45
5.24 EIGRP Filtering with Administrative Distance 45
5.25 EIGRP Filtering with Per Neighbor AD 45
5.26 EIGRP Filtering with Route Maps 45
5.27 EIGRP Bandwidth Pacing 45
5.28 EIGRP Default Metric 45
5.29 EIGRP Neighbor Logging 46
5.30 EIGRP Router-ID 46
5.31 EIGRP Maximum Hops 46

OSPF 48
6.1 OSPF over Broadcast Media 48
6.2 OSPF over Non-Broadcast Media 49
6.3 OSPF DR/BDR Election Manipulation 49
6.4 OSPF Network Point-to-Point 49
6.5 OSPF Network Point-to-Multipoint 49
6.6 OSPF Network Point-to-Multipoint Non-Broadcast 49
6.7 OSPF Network Loopback 50
6.8 OSPF Path Selection with Auto-Cost 50
6.9 OSPF Path Selection with Cost 50
6.10 OSPF Path Selection with Bandwidth 50
6.11 OSPF Path Selection with Per-Neighbor Cost 50
6.12 Repairing Discontiguous OSPF Areas with Virtual-Links 50
6.13 OSPF Path Selection with Non-Backbone Transit Areas 51
6.14 OSPF Path Selection with Virtual-Links 51
6.15 OSPF Demand Circuit 51
6.16 OSPF Flooding Reduction 51
6.17 OSPF Clear Text Authentication 51
6.18 OSPF MD5 Authentication 51
6.19 OSPF Null Authentication 52
6.20 OSPF MD5 Authentication with Multiple Keys 52
6.21 OSPF Internal Summarization 52
6.22 OSPF Path Selection with Summarization 52
6.23 OSPF External Summarization 52
6.24 OSPF Stub Areas 53
6.25 OSPF Totally Stubby Areas 53
6.26 OSPF Not-So-Stubby Areas 53
6.27 OSPF Not-So-Stubby Areas and Default Routing 53
6.28 OSPF Not-So-Totally-Stubby Areas 53
6.29 OSPF Stub Areas with Multiple Exit Points 54
6.30 OSPF NSSA Type-7 to Type-5 Translator Election 54
6.31 OSPF NSSA Redistribution Filtering 54
6.32 OSPF LSA Type-3 Filtering 54
6.33 OSPF Forwarding Address Suppression 55

Copyright © 2010 Internetwork Expert

6.34 OSPF Default Routing 55
6.35 OSPF Conditional Default Routing. 55
6.36 OSPF Reliable Conditional Default Routing. 55
6.37 OSPF Filtering with Distribute-Lists 56
6.38 OSPF Summarization and Discard Routes. 56
6.39 OSPF Filtering with Administrative Distance. 56
6.40 OSPF Filtering with Route-Maps. 56
6.41 OSPF NSSA ABR External Prefix Filtering 56
6.42 OSPF Database Filtering 56
6.43 OSPF Stub Router Advertisement 56
6.44 OSPF Interface Timers 57
6.45 OSPF Global Timers. 57
6.46 OSPF Resource Limiting. 57
6.47 Miscellaneous OSPF Features 58

BGP. 60
7.1 Establishing iBGP Peerings 60
7.2 Establishing EBGP Peerings. 60
7.3 BGP Update Source Modification. 60
7.4 Multihop EBGP Peerings. 60
7.5 Neighbor Disable-Connected-Check. 61
7.6 Authenticating BGP Peerings. 61
7.7 iBGP Route Reflection 61
7.8 Large Scale iBGP Route Reflection with Clusters. 62
7.9 iBGP Confederation 63
7.10 BGP Next-Hop Processing - Next-Hop-Self. 64
7.11 BGP Next-Hop Processing - Manual Modification. 64
7.12 iBGP Synchronization 64
7.13 BGP over GRE. 65
7.14 BGP Redistribute Internal 65
7.15 BGP Peer Groups. 66
7.16 BGP Network Statement. 66
7.17 BGP Auto-Summary 66
7.18 BGP Bestpath Selection - Weight 68
7.19 BGP Bestpath Selection - Local Preference 68
7.20 BGP Bestpath Selection - AS-Path Prepending. 68
7.21 BGP Bestpath Selection - Origin. 68
7.23 BGP Bestpath Selection - MED 68
7.24 BGP Bestpath Selection - Always Compare MED 68
7.25 BGP Bestpath Selection - AS-Path Ignore. 68
7.26 BGP Bestpath Selection - Router-IDs 69
7.27 BGP Bestpath Selection - DMZ Link Bandwidth 69
7.28 BGP Bestpath Selection - Maximum AS Limit. 69
7.29 BGP Backdoor 69
7.30 BGP Aggregation 69
7.31 BGP Aggregation - Summary Only 69

Copyright © 2010 Internetwork Expert

7.32 BGP Aggregation - Suppress Map 69
7.33 BGP Aggregation - Unsuppress Map 70
7.34 BGP Aggregation - AS-Set 70
7.35 BGP Aggregation - Attribute-Map 70
7.36 BGP Aggregation - Advertise Map 70
7.37 BGP Communities 71
7.38 BGP Communities - No-Advertise 71
7.39 BGP Communities - No-Export 71
7.40 BGP Communities - Local-AS 71
7.41 BGP Communities - Deleting 71
7.42 BGP Conditional Advertisement 72
7.43 BGP Conditional Route Injection 72
7.44 BGP Filtering with Prefix-Lists 72
7.45 BGP Filtering with Standard Access-Lists 73
7.46 BGP Filtering with Extended Access-Lists 73
7.47 BGP Regular Expressions 73
7.49 BGP Default Routing 74
7.50 BGP Local AS 74
7.51 BGP Local AS Replace-AS/Dual-AS 74
7.52 BGP Remove Private AS 74
7.53 BGP Dampening 74
7.54 BGP Dampening with Route-Map 75
7.55 BGP Timers Tuning 75
7.56 BGP Fast Fallover 75
7.57 BGP Outbound Route Filtering 75
7.58 BGP Soft Reconfiguration 75
7.59 BGP Next-Hop Trigger 75
7.60 BGP TTL Security 76
7.61 BGP AllowAS in 76

Multicast 78
8.1 PIM Dense Mode 79
8.2 Multicast RPF Failure 79
8.3 PIM Sparse Mode 79
8.4 PIM Sparse-Dense Mode 79
8.5 PIM Assert 80
8.6 PIM Accept RP 80
8.7 PIM DR Election 80
8.8 PIM Accept Register 80
8.9 Multicast Tunneling 80
8.10 PIM NBMA Mode 80
8.11 Auto-RP 80
8.12 Auto-RP - Multiple Candidate RPs 81
8.13 Auto-RP - Filtering Candidate RPs 81
8.14 Auto-RP Listener 81
8.15 Auto-RP and RP/MA Placement 81

Copyright © 2010 Internetwork Expert

8.16 Filtering Auto-RP Messages 81
8.17 Multicast Boundary 81
8.18 PIM Bootstrap Router 81
8.19 BSR - Multiple RP Candidates 82
8.20 Filtering BSR Messages. 82
8.21 Stub Multicast Routing & IGMP Helper 82
8.22 IGMP Filtering 82
8.23 IGMP Timers. 82
8.24 Multicast Helper Map 83
8.25 Multicast Rate Limiting. 83
8.26 Bidirectional PIM 83
8.27 Source Specific Multicast. 83
8.28 DVMRP Interoperability. 83
8.29 Multicast BGP Extension. 85
8.30 MSDP 85
8.31 Anycast RP 85
8.32 Catalyst IGMP Snooping. 85
8.33 Catalyst Multicast VLAN Registration 86
8.34 Catalyst IGMP Profiles. 86

IPv6 88
9.1 IPv6 Link-Local Addressing. 88
9.2 IPv6 Unique Local Addressing 88
9.3 IPv6 Global Aggregatable Addressing. 88
9.4 IPv6 EUI-64 Addressing. 88
9.5 IPv6 Auto-Configuration 89
9.6 RIPng 89
9.7 RIPng over NBMA. 89
9.8 RIPng Summarization 89
9.9 RIPng Prefix Filtering 90
9.10 RIPng Metric Manipulation. 90
9.11 RIPng Default Routing 90
9.12 EIGRPv6. 90
9.13 EIGRPv6 Summarization 90
9.14 EIGRPv6 Prefix Filtering 90
9.15 EIGRPv6 Metric Manipulation. 91
9.16 EIGRPv6 Default Routing 91
9.17 OSPFv3 91
9.18 OSPFv3 over NBMA. 91
9.19 OSPFv3 Virtual Links. 91
9.20 OSPFv3 Summarization. 91
9.21 IPv6 Redistribution. 92
9.22 IPv6 Filtering. 92
9.23 IPv6 NAT-PT. 92
9.24 IPv6 MP-BGP. 92
9.25 IPv6 PIM and MLD. 92

Copyright © 2010 Internetwork Expert

9.26 IPv6 PIM BSR 93
9.27 IPv6 Embedded RP 93
9.28 IPv6 SSM 93
9.29 IPv6 Tunneling 93
9.30 Automatic 6to4 Tunnels 94
9.31 ISATAP Tunnels 94

Quality of Service 96
10.1 Hold-Queue and Tx-Ring 96
10.2 Weighted Fair Queuing (WFQ) 96
10.3 Legacy RTP Reserved Queue 96
10.4 Legacy RTP Prioritization. 96
10.5 Legacy Custom Queueing. 97
10.6 Legacy Custom Queueing with Prioritization. 97
10.7 Legacy Priority Queueing. 97
10.8 Legacy Random Early Detection. 97
10.9 Legacy Flow-Based Random Early Detection. 98
10.10 Selective Packet Discard 98
10.11 Payload Compression on Serial Links 98
10.12 Generic TCP/UDP Header Compression. 98
10.13 MLP Link Fragmentation and Interleaving 98
10.14 Legacy Generic Traffic Shaping 99
10.15 Legacy CAR for Admission Control 99
10.16 Oversubscription with Legacy CAR and WFQ 99
10.17 Legacy CAR for Rate Limiting 99
10.18 Legacy CAR Access-Lists 99
10.19 Legacy GTS for Frame Relay 100
10.20 Legacy Frame Relay Traffic Shaping. 100
10.21 Legacy Adaptive FRTS. 100
10.22 Legacy FRTS with Per-VC WFQ 100
10.23 Legacy FRTS with Per-VC PQ 101
10.24 Legacy FRTS with Per-VC CQ 101
10.25 Legacy FRTS with Per-VC Fragmentation. 101
10.26 Legacy FRTS with Per-VC IP RTP Priority 101
10.27 Frame-Relay RTP/TCP Header Compression. 101
10.28 Frame-Relay Broadcast Queue 102
10.29 Frame-Relay DE Marking 102
10.30 Legacy FRTS PVC Interface Priority Queue. 102
10.31 Frame-Relay Priority to DLCI Mapping 102
10.32 Frame-Relay Traffic Policing & Congestion Mgmt 103
10.33 MQC Classification and Marking. 104
10.34 MQC Bandwidth Reservations and CBWFQ 104
10.35 MQC Bandwidth Percent 104
10.36 MQC LLQ and Remaining Bandwidth Reservations 105
10.37 MQC WRED 105
10.38 MQC Dynamic Flows and WRED 105

Copyright © 2010 Internetwork Expert

10.39 MQC WRED with ECN 105
10.40 MQC Class-Based Generic Traffic Shaping 105
10.41 MQC Class-Based GTS and CBWFQ 106
10.42 MQC Single-Rate Three-Color Policer 106
10.43 MQC Hierarchical Policers 106
10.44 MQC Two-Rate Three-Color Policer 107
10.45 MQC Peak Shaping 107
10.46 MQC Percent-Based Policing 107
10.47 MQC Header Compression 107
10.48 Using Class-Based GTS for FRTS 108
10.49 MQC Based Frame-Relay DE-Marking 108
10.50 Using MQC CBWFQ with Legacy FRTS 108
10.51 MQC Compatible FRF.12 Fragmentation 108
10.52 MQC Based Frame-Relay Traffic Shaping 109
10.53 Voice Adaptive Traffic Shaping 109
10.54 Voice Adaptive Fragmentation 109
10.55 MLPPP LFI over Frame Relay 110
10.56 QoS Pre-Classify 110
10.57 RSVP and WFQ 111
10.58 RSVP and SBM 111
10.59 RSVP and CBWFQ 111
10.60 RSVP and LLQ 111
10.61 RSVP and Per-VC WFQ 112
10.62 Catalyst QoS Port-Based Classification 112
10.63 Catalyst QoS Marking Pass-Through 112
10.64 Catalyst QoS ACL Based Classification & Marking 113
10.65 Catalyst 3550 Per-Port Per-VLAN Classification 113
10.66 Catalyst 3560 Per-VLAN Classification 113
10.67 Catalyst QoS Port-Based Policing and Marking 114
10.68 Catalyst 3560 Per-Port Per-VLAN Policing 114
10.69 Catalyst 3550 Per-Port Per-VLAN Policing 114
10.70 Catalyst QoS Aggregate Policers 115
10.71 Catalyst 3560 Ingress Queueing 115
10.72 Catalyst 3560 Ingress Queue Tuning 115
10.73 Catalyst 3550 Egress Queueing 116
10.74 Catalyst 3550 Regular Queues Tuning 116
10.75 Catalyst 3550 Gigabit Interface Queues Tuning 116
10.76 Catalyst 3550 Egress Policing 117
10.77 Catalyst 3560 SRR Shared Mode 117
10.78 Catalyst 3560 SRR Shaped Mode 117
10.79 Catalyst 3560 Egress Queues Tuning 118
10.80 Catalyst QoS DSCP Mutation 118
10.81 Advanced HTTP Classification with NBAR 118

Security 120
11.1 AAA Authentication Lists 120

Copyright © 2010 Internetwork Expert

11.2 AAA Exec Authorization 120
11.3 AAA Local Command Authorization 121
11.3 AAA Local Command Authorization 121
11.4 Traffic Filtering with Standard Access-Lists 121
11.5 Traffic Filtering with Extended Access-Lists 121
11.6 Traffic Filtering with Reflexive Access-Lists 122
11.7 Filtering Fragmented Packets 122
11.8 Access Control with Dynamic Access-Lists 122
11.9 Traffic Filtering with Time-Based Access-Lists 122
11.10 Traffic Filtering with Policy-Based Routing 123
11.11 Preventing Spoofing with uRPF 123
11.12 Using NBAR for Content-Based Filtering 123
11.13 TCP Intercept 123
11.14 TCP Intercept Watch Mode 124
11.15 Packet Logging with Access-Lists 124
11.16 Stateful Filtering with CBAC 124
11.17 Advanced CBAC Features 125
11.18 CBAC TCP/UDP Intercept Feature 125
11.19 VLAN Filters for IP Traffic 125
11.20 VLAN Filters for Non-IP Traffic 126
11.21 Port Security 126
11.22 HSRP and Port-Security 126
11.23 DHCP Snooping 127
11.24 DHCP Snooping and the Information Option 127
11.25 Dynamic ARP Inspection 127
11.26 IP Source Guard 127
11.27 Using Catalyst Ingress Access-Lists 128
11.28 Controlling Terminal Line Access 128
11.29 IOS Login Enhancements 128
11.30 Role Based CLI 128
11.31 IP Source Tracker 129
11.32 Router IP Traffic Export 129
11.33 Controlling the ICMP Messages Rate 129
11.34 Control Plane Policing 129
11.35 Control Plane Protection 130
11.36 IOS ACL Selective IP Option Drop 130
11.37 BGP Generic TTL Security Mechanism 130
11.38 Flexible Packet Matching 130
11.39 Zone Based Firewall 131
11.40 ZFW Rate Limiting 131
11.41 ZFW Application Inspection 131
11.42 Classic IOS Transparent Firewall 132
11.43 ZFW-Based IOS Transparent Firewall 132
11.44 IOS IPS 132

System Management 134

Copyright © 2010 Internetwork Expert

12.1 Exec Aliases 134
12.2 System Message Logging. 134
12.3 Syslog Logging 135
12.4 Logging Counting and Timestamps 135
12.5 Logging to Flash Memory. 135
12.6 Configuration Change Notification and Logging. 135
12.7 Configuration Archive & Rollback. 136
12.8 Logging with Access-Lists. 136
12.9 TCP Keepalives 136
12.10 Generating Exception Core Dumps 136
12.11 Conditional Debugging 136
12.12 Telnet Service Options 137
12.13 Tuning Packet Buffers 137
12.14 Terminal Line Settings. 137
12.15 SNMPv2c Server 138
12.16 SNMPv2c Access Control. 138
12.17 SNMP Traps and Informs 138
12.18 CPU and Memory Thresholds 138
12.19 SNMPv3 139
12.20 SNMP MAC Address Notifications. 139
12.21 SNMP Notifications of Syslog Messages. 139
12.22 CDP 140
12.23 RMON Alarms. 140
12.24 RMON Statistics Collection. 140
12.25 HTTP Server and Client. 140
12.26 FTP Server and Client 141
12.27 TFTP Server and Client. 141
12.28 Remote Shell 141
12.29 NTP 141
12.30 NTP Authentication. 142
12.31 NTP Access Control 142
12.32 Auto-Install over LAN Interfaces using DHCP 142
12.33 Auto-Install over Frame-Relay 142
12.34 Auto-Install over LAN Interfaces using RARP. 143
12.35 IOS Menus. 143
12.36 IOS Banners 144
12.37 KRON Command Schedule. 144
12.38 EEM Scripting: Interface Events 144
12.39 EEM Scripting: Syslog Events 144
12.40 EEM Scripting: CLI Events 145
12.41 EEM Scripting: Periodic Scheduling 145
12.42 EEM Scripting: Advanced Features. 145

IP Services 146
13.1 Proxy ARP 146
13.2 DHCP Server 146

Copyright © 2010 Internetwork Expert

13.3 DHCP Client. 147
13.4 DHCP Relay. 147
13.5 DHCP Host Pool 147
13.6 DHCP On-Demand Pool 147
13.7 DHCP Proxy. 147
13.8 DHCP Information Option 148
13.9 DHCP Authorized ARP 148
13.10 IP SLA 148
13.11 Object Tracking. 148
13.12 HSRP 149
13.13 VRRP 149
13.14 GLBP 149
13.15 Router Redundancy and Objects Tracking 149
13.16 IRDP 150
13.17 Router ICMP Settings 150
13.18 Basic NAT. 150
13.19 NAT Overload 150
13.20 NAT with Route Maps 150
13.21 Static NAT 151
13.22 Static PAT. 151
13.23 NAT and IP Aliasing. 151
13.24 Static Policy NAT 151
13.25 NAT with Overlapping Subnets 151
13.26 TCP Load Distribution with NAT 152
13.27 Stateful NAT with HSRP 152
13.28 Stateful NAT Primary/Backup 152
13.29 NAT Virtual Interface 153
13.30 NAT Default Interface 153
13.31 Reversible NAT. 153
13.32 Static Extendable NAT. 153
13.33 IP Precedence Accounting 154
13.34 IP Output Packets Accounting 154
13.35 IP Access Violation Accounting 154
13.36 MAC Address Accounting. 154
13.37 TCP Optimization. 154
13.38 IOS Small Services & Finger 155
13.39 Directed Broadcasts & UDP Forwarding. 155
13.40 DRP Server Agent 155
13.41 WCCPv1 Web-Cache 155
13.42 WCCPv2 Services 156
13.43 NBAR Protocol Discovery. 156
13.44 Netflow Ingress & Egress 156
13.45 Netflow Top Talkers. 157
13.46 Netflow Aggregation Cache 157
13.47 Netflow Random Sampling 157
13.48 Netflow Input Filters 157

Copyright © 2010 Internetwork Expert

13.49 IOS Authoritative DNS Server 158
13.50 IOS Caching DNS Server 158
13.51 IOS DNS Spoofing 158
13.52 IP Event Dampening 158

MPLS VPN 160
14.1 VRF Lite 160
14.2 MPLS LDP 160
14.3 MPLS Label Filtering 160
14.4 MP-BGP VPNv4 161
14.5 MP-BGP Prefix Filtering 161
14.6 PE-CE Routing with RIP 161
14.7 PE-CE Routing with OSPF 161
14.8 OSPF Sham-Link 162
14.9 PE-CE Routing with EIGRP 162
14.10 EIGRP Site-of-Origin 162
14.11 PE-CE Routing with BGP 162
14.12 BGP SoO Attribute 162
14.13 Internet Access 163
14.14 AToM 163
14.15 L2TPV3 163
14.16 MPLS VPN Performance Tuning 163

Bridging & Switching Solutions 164
1.1 Layer 2 Access Switchports 164
1.2 Layer 2 Dynamic Switchports 170
1.3 ISL Trunking 174
1.4 802.1q Trunking 176
1.5 802.1q Native VLAN 178
1.6 Disabling DTP Negotiation 180
1.7 Router-On-A-Stick 184
1.8 VTP 186
1.9 VTP Transparent 192
1.10 VTP Pruning 194
1.11 VTP Prune-Eligible List 198
1.12 Layer 2 EtherChannel 202
1.13 Layer 2 EtherChannel with PAgP 210
1.14 Layer 2 EtherChannel with LACP 216
1.15 Layer 3 EtherChannel 222
1.16 802.1q Tunneling 226
1.17 EtherChannel over 802.1q Tunneling 232
1.18 STP Root Bridge Election 238
1.19 STP Load Balancing with Port Cost 248
1.20 STP Load Balancing with Port Priority 250
1.21 Tuning STP Convergence Timers 256
1.22 STP PortFast 258
1.23 STP PortFast Default 260

Copyright © 2010 Internetwork Expert

1.24 STP UplinkFast 262
1.25 STP BackboneFast 264
1.26 STP BPDU Guard 266
1.27 STP BPDU Guard Default 268
1.28 STP BPDU Filter 270
1.29 STP BPDU Filter Default 274
1.30 STP Root Guard 276
1.31 STP Loop Guard 278
1.32 Unidirectional Link Detection 282
1.33 MST Root Bridge Election 286
1.34 MST Load Balancing with Port Cost 296
1.35 MST Load Balancing with Port Priority 300
1.36 MST and Rapid Spanning Tree 304
1.37 Protected Ports 306
1.38 Storm Control 308
1.39 MAC-Address Table Static Entries & Aging 310
1.40 SPAN 314
1.41 RSPAN 316
1.42 Voice VLAN 320
1.43 IP Phone Trust and CoS Extend 324
1.44 Smartport Macros 326
1.45 Flex Links 330
1.46 Fallback Bridging 336
1.47 Private VLANs 340
1.48 PPP 346
1.49 PPP AAA Authentication 354
1.50 PPPoE 362

Frame Relay Solutions 370
2.1 Inverse-ARP 370
2.2 Static Mappings 374
2.3 Point-to-Point Subinterfaces 376
2.4 Multipoint Subinterfaces & Inverse-ARP 378
2.5 Multipoint Subinterfaces & Static Mappings 382
2.6 Hub-and-Spoke & Static Mappings 386
2.7 Hub-and-Spoke & Inverse-ARP 390
2.8 Hub-and-Spoke & Point-to-Point Subinterfaces 394
2.9 Disabling Inverse-ARP 398
2.10 Back-to-Back Frame Relay 402
2.11 Frame Relay End-to-End Keepalives 404
2.12 Frame Relay Broadcast Queue 406
2.13 Frame Relay TCP & RTP Header Compression 406
2.14 PPP over Frame Relay 408
2.15 Bridging over Frame Relay 412
2.16 Frame-Relay Switching 416

IP Routing Solutions 422

Copyright © 2010 Internetwork Expert

3.1 Routing to Multipoint Broadcast Interfaces. 422
3.2 Routing to NBMA Interfaces. 426
3.3 Longest Match Routing 430
3.4 Floating Static Routes 432
3.5 Backup Interface 434
3.6 Reliable Static Routing with Enhanced Object Tracking. 438
3.7 Policy Routing 444
3.8 Reliable Policy Routing 448
3.9 Local Policy Routing. 454
3.10 GRE Tunneling 456
3.11 GRE Tunneling and Recursive Routing 460
3.12 Reliable Backup Interface with GRE 462
3.13 On-Demand Routing (ODR). 468
3.14 OER Components Setup. 472
3.15 OER Profile Phase. 480
3.16 OER Measure Phase 489
3.17 OER Apply Policy Phase. 502
3.18 OER Control & Verify Phase 518

RIP Solutions. 528
4.1 Basic RIP Configuration 528
4.2 RIPv2 Authentication. 536
4.3 RIPv2 Split Horizon 540
4.4 RIPv2 Auto-Summary. 542
4.5 RIP Send and Receive Versions 544
4.6 RIPv2 Manual Summarization. 550
4.7 RIPv2 Convergence Timers 552
4.8 RIPv2 Offset List 554
4.9 RIPv2 Filtering with Passive Interface 556
4.10 RIPv2 Filtering with Prefix-Lists 558
4.11 RIPv2 Filtering with Standard Access-Lists. 566
4.12 RIPv2 Filtering with Extended Access-Lists. 566
4.13 RIPv2 Filtering with Offset Lists 570
4.14 RIPv2 Filtering with Administrative Distance. 572
4.15 RIPv2 Filtering with Per Neighbor AD 574
4.16 RIPv2 Default Routing 576
4.17 RIPv2 Conditional Default Routing. 582
4.18 RIPv2 Reliable Conditional Default Routing. 584
4.19 RIPv2 Unicast Updates. 588
4.20 RIPv2 Broadcast Updates. 590
4.21 RIPv2 Triggered Updates. 592
4.22 RIPv2 Source Validation 594

EIGRP Solutions 598
5.1 EIGRP Network Statement. 598
5.2 EIGRP Auto-Summary 604
5.3 EIGRP Split Horizon. 606

Copyright © 2010 Internetwork Expert

5.4 EIGRP MD5 Authentication. 608
5.5 EIGRP Key Chain Rotation 610
5.6 EIGRP Unicast Updates. 612
5.7 EIGRP Default Network. 614
5.8 EIGRP Summarization 616
5.9 EIGRP Summarization with Default Routing 618
5.10 EIGRP Summarization with Leak Map. 620
5.11 EIGRP Floating Summarization 622
5.12 EIGRP Poisoned Floating Summarization 626
5.13 EIGRP Metric Weights 628
5.14 EIGRP Traffic Engineering with Metric. 632
5.15 EIGRP Unequal Cost Load Balancing 636
5.16 EIGRP Convergence Timers. 644
5.17 EIGRP Stub Routing 650
5.18 EIGRP Stub Routing with Leak Map 654
5.19 EIGRP Filtering with Passive Interface 658
5.20 EIGRP Filtering with Prefix-Lists 660
5.21 EIGRP Filtering with Standard Access-Lists 664
5.22 EIGRP Filtering with Extended Access-Lists. 666
5.23 EIGRP Filtering with Offset Lists. 670
5.24 EIGRP Filtering with Administrative Distance 674
5.25 EIGRP Filtering with Per Neighbor AD. 676
5.26 EIGRP Filtering with Route Maps 680
5.27 EIGRP Bandwidth Pacing 686
5.28 EIGRP Default Metric. 688
5.29 EIGRP Neighbor Logging. 690
5.30 EIGRP Router-ID 690
5.31 EIGRP Maximum Hops. 692

OSPF Solutions 694
6.1 OSPF over Broadcast Media. 694
6.2 OSPF over Non-Broadcast Media. 704
6.3 OSPF DR/BDR Election Manipulation. 714
6.4 OSPF Network Point-to-Point. 726
6.5 OSPF Network Point-to-Multipoint 730
6.6 OSPF Network Point-to-Multipoint Non-Broadcast. 738
6.7 OSPF Network Loopback. 742
6.8 OSPF Path Selection with Auto-Cost 746
6.9 OSPF Path Selection with Cost. 750
6.10 OSPF Path Selection with Bandwidth 760
6.11 OSPF Path Selection with Per-Neighbor Cost 766
6.12 Repairing Discontiguous OSPF Areas with Virtual-Links 770
6.13 OSPF Path Selection with Non-Backbone Transit Areas 782
6.14 OSPF Path Selection with Virtual-Links 792
6.15 OSPF Demand Circuit 798
6.16 OSPF Flooding Reduction 800

Copyright © 2010 Internetwork Expert

6.17 OSPF Clear Text Authentication 802
6.18 OSPF MD5 Authentication 806
6.19 OSPF Null Authentication 810
6.20 OSPF MD5 Authentication with Multiple Keys 812
6.21 OSPF Internal Summarization 814
6.22 OSPF Path Selection with Summarization 816
6.23 OSPF External Summarization 820
6.24 OSPF Stub Areas 832
6.25 OSPF Totally Stubby Areas 840
6.26 OSPF Not-So-Stubby Areas 844
6.27 OSPF Not-So-Stubby Areas and Default Routing 850
6.28 OSPF Not-So-Totally-Stubby Areas 852
6.29 OSPF Stub Areas with Multiple Exit Points 854
6.30 OSPF NSSA Type-7 to Type-5 Translator Election 860
6.31 OSPF NSSA Redistribution Filtering 868
6.32 OSPF LSA Type-3 Filtering 872
6.33 OSPF Forwarding Address Suppression 876
6.34 OSPF Default Routing 884
6.35 OSPF Conditional Default Routing 888
6.36 OSPF Reliable Conditional Default Routing 890
6.37 OSPF Filtering with Distribute-Lists 894
6.38 OSPF Summarization and Discard Routes 896
6.39 OSPF Filtering with Administrative Distance 900
6.40 OSPF Filtering with Route-Maps 902
6.41 OSPF NSSA ABR External Prefix Filtering 904
6.42 OSPF Database Filtering 906
6.43 OSPF Stub Router Advertisement 908
6.44 OSPF Interface Timers 912
6.45 OSPF Global Timers 914
6.46 OSPF Resource Limiting 916
6.47 Miscellaneous OSPF Features 918

BGP Solutions 920
7.1 Establishing iBGP Peerings 920
7.2 Establishing EBGP Peerings 930
7.3 BGP Update Source Modification 934
7.4 Multihop EBGP Peerings 940
7.5 Neighbor Disable-Connected-Check 942
7.6 Authenticating BGP Peerings 948
7.7 iBGP Route Reflection 950
7.8 Large Scale iBGP Route Reflection with Clusters 956
7.9 iBGP Confederation 968
7.10 BGP Next-Hop Processing - Next-Hop-Self 974
7.11 BGP Next-Hop Processing - Manual Modification 980
7.12 iBGP Synchronization 984
7.13 BGP over GRE 992

Copyright © 2010 Internetwork Expert

7.14 BGP Redistribute Internal 998
7.15 BGP Peer Groups 1004
7.16 BGP Network Statement 1010
7.17 BGP Auto-Summary 1014
7.18 BGP Bestpath Selection - Weight 1022
7.19 BGP Bestpath Selection - Local Preference 1028
7.20 BGP Bestpath Selection - AS-Path Prepending 1032
7.21 BGP Bestpath Selection - Origin 1036
7.23 BGP Bestpath Selection - MED 1040
7.24 BGP Bestpath Selection - Always Compare MED 1046
7.25 BGP Bestpath Selection - AS-Path Ignore 1052
7.26 BGP Bestpath Selection - Router-IDs 1056
7.27 BGP Bestpath Selection - DMZ Link Bandwidth 1060
7.28 BGP Bestpath Selection - Maximum AS Limit 1066
7.29 BGP Backdoor 1070
7.30 BGP Aggregation 1074
7.31 BGP Aggregation - Summary Only 1080
7.32 BGP Aggregation - Suppress Map 1082
7.33 BGP Aggregation - Unsuppress Map 1086
7.34 BGP Aggregation - AS-Set 1090
7.35 BGP Aggregation - Attribute-Map 1094
7.36 BGP Aggregation - Advertise Map 1098
7.37 BGP Communities 1102
7.38 BGP Communities - No-Advertise 1108
7.39 BGP Communities - No-Export 1110
7.40 BGP Communities - Local-AS 1114
7.41 BGP Communities - Deleting 1118
7.42 BGP Conditional Advertisement 1122
7.43 BGP Conditional Route Injection 1128
7.44 BGP Filtering with Prefix-Lists 1136
7.45 BGP Filtering with Standard Access-Lists 1140
7.46 BGP Filtering with Extended Access-Lists 1144
7.47 BGP Regular Expressions 1148
7.49 BGP Default Routing 1158
7.50 BGP Local AS 1160
7.51 BGP Local AS Replace-AS/Dual-AS 1166
7.52 BGP Remove Private AS 1172
7.53 BGP Dampening 1176
7.54 BGP Dampening with Route-Map 1180
7.55 BGP Timers Tuning 1184
7.56 BGP Fast Fallover 1188
7.57 BGP Outbound Route Filtering 1192
7.58 BGP Soft Reconfiguration 1196
7.59 BGP Next-Hop Trigger 1200
7.60 BGP TTL Security 1202
7.61 BGP AllowAS in 1204

Copyright © 2010 Internetwork Expert

Multicast Solutions 1208
8.1 PIM Dense Mode 1210
8.2 Multicast RPF Failure 1218
8.3 PIM Sparse Mode 1224
8.4 PIM Sparse-Dense Mode 1234
8.5 PIM Assert 1242
8.6 PIM Accept RP 1248
8.7 PIM DR Election 1250
8.8 PIM Accept Register 1252
8.9 Multicast Tunneling 1256
8.10 PIM NBMA Mode 1260
8.11 Auto-RP 1266
8.12 Auto-RP - Multiple Candidate RPs 1270
8.13 Auto-RP - Filtering Candidate RPs 1276
8.14 Auto-RP Listener 1278
8.15 Auto-RP and RP/MA Placement 1282
8.16 Filtering Auto-RP Messages 1288
8.17 Multicast Boundary 1290
8.18 PIM Bootstrap Router 1294
8.19 BSR - Multiple RP Candidates 1298
8.20 Filtering BSR Messages 1302
8.21 Stub Multicast Routing & IGMP Helper 1304
8.22 IGMP Filtering 1310
8.23 IGMP Timers 1314
8.24 Multicast Helper Map 1320
8.25 Multicast Rate Limiting 1330
8.26 Bidirectional PIM 1334
8.27 Source Specific Multicast 1340
8.27 Source Specific Multicast 1340
8.28 DVMRP Interoperability 1344
8.29 Multicast BGP Extension 1350
8.30 MSDP 1356
8.31 Anycast RP 1366
8.32 Catalyst IGMP Snooping 1372
8.33 Catalyst Multicast VLAN Registration 1378
8.34 Catalyst IGMP Profiles 1384
IPv6 Solutions 1386
9.1 IPv6 Link-Local Addressing 1386
9.2 IPv6 Unique Local Addressing 1390
9.3 IPv6 Global Aggregatable Addressing 1392
9.4 IPv6 EUI-64 Addressing 1396
9.5 IPv6 Auto-Configuration 1398
9.6 RIPng 1402
9.7 RIPng over NBMA 1406
9.8 RIPng Summarization 1408

Copyright © 2010 Internetwork Expert

9.9 RIPng Prefix Filtering 1410
9.10 RIPng Metric Manipulation 1412
9.11 RIPng Default Routing 1414
9.12 EIGRPv6 1416
9.13 EIGRPv6 Summarization 1422
9.14 EIGRPv6 Prefix Filtering 1424
9.15 EIGRPv6 Metric Manipulation 1426
9.16 EIGRPv6 Default Routing 1430
9.17 OSPFv3 1432
9.18 OSPFv3 over NBMA 1436
9.19 OSPFv3 Virtual Links 1438
9.20 OSPFv3 Summarization 1440
9.21 IPv6 Redistribution 1444
9.22 IPv6 Filtering 1448
9.23 IPv6 NAT-PT 1450
9.24 IPv6 MP-BGP 1452
9.25 IPv6 PIM and MLD 1456
9.26 IPv6 PIM BSR 1460
9.27 IPv6 Embedded RP 1466
9.28 IPv6 SSM 1470
9.29 IPv6 Tunneling 1474
9.30 Automatic 6to4 Tunneling 1478
9.31 ISATAP Tunneling 1482

Quality of Service Solutions 1486
10.1 Hold-Queue and Tx-Ring 1486
10.2 Weighted Fair Queuing (WFQ) 1490
10.3 Legacy RTP Reserved Queue 1500
10.4 Legacy RTP Prioritization 1506
10.5 Legacy Custom Queueing 1510
10.6 Legacy Custom Queueing with Prioritization 1520
10.7 Legacy Priority Queueing 1524
10.8 Legacy Random Early Detection 1532
10.9 Legacy Flow-Based Random Early Detection 1540
10.10 Selective Packet Discard 1542
10.11 Payload Compression on Serial Links 1546
10.12 Generic TCP/UDP Header Compression 1552
10.13 MLP Link Fragmentation and Interleaving 1556
10.14 Legacy Generic Traffic Shaping 1564
10.15 Legacy CAR for Admission Control 1574
10.16 Oversubscription with Legacy CAR and WFQ 1580
10.17 Legacy CAR for Rate Limiting 1590
10.18 Legacy CAR Access-Lists 1594
10.19 Legacy GTS for Frame Relay 1598
10.20 Legacy Frame Relay Traffic Shaping 1602
10.21 Legacy Adaptive FRTS 1606

Copyright © 2010 Internetwork Expert

10.22 Legacy FRTS with Per-VC WFQ 1610
10.23 Legacy FRTS with Per-VC PQ 1612
10.24 Legacy FRTS with Per-VC CQ 1616
10.25 Legacy FRTS with Per-VC Fragmentation 1620
10.26 Legacy FRTS with Per-VC IP RTP Priority 1628
10.27 Frame-Relay RTP/TCP Header Compression 1634
10.28 Frame-Relay Broadcast Queue 1638
10.29 Frame-Relay DE Marking 1640
10.30 Legacy FRTS PVC Interface Priority Queue 1642
10.31 Frame-Relay Priority to DLCI Mapping 1644
10.32 Frame-Relay Traffic Policing & Congestion Mgmt 1648
10.33 MQC Classification and Marking 1658
10.34 MQC Bandwidth Reservations and CBWFQ 1664
10.35 MQC Bandwidth Percent 1675
10.36 MQC LLQ and Remaining Bandwidth Reservations 1681
10.37 MQC WRED 1689
10.38 MQC Dynamic Flows and WRED 1693
10.39 MQC WRED with ECN 1697
10.40 MQC Class-Based Generic Traffic Shaping 1701
10.41 MQC Class-Based GTS and CBWFQ 1705
10.42 MQC Single-Rate Three-Color Policer 1713
10.43 MQC Hierarchical Policers 1721
10.44 MQC Two-Rate Three-Color Policer 1729
10.45 MQC Peak Shaping 1737
10.46 MQC Percent-Based Policing 1743
10.47 MQC Header Compression 1745
10.48 Using Class-Based GTS for FRTS 1749
10.49 MQC Based Frame-Relay DE-Marking 1753
10.50 Using MQC CBWFQ with Legacy FRTS 1755
10.51 MQC Compatible FRF.12 Fragmentation 1759
10.52 MQC Based Frame-Relay Traffic Shaping 1761
10.53 Voice Adaptive Traffic Shaping 1765
10.54 Voice Adaptive Fragmentation 1771
10.55 MLPPP LFI over Frame Relay 1775
10.56 QoS Pre-Classify 1783
10.57 RSVP and WFQ 1787
10.58 RSVP and SBM 1805
10.59 RSVP and CBWFQ 1809
10.60 RSVP and LLQ 1819
10.61 RSVP and Per-VC WFQ 1825
10.62 Catalyst QoS Port-Based Classification 1831
10.63 Catalyst QoS Marking Pass-Through 1847
10.64 Catalyst QoS ACL Based Classification & Marking 1853
10.65 Catalyst 3550 Per-Port Per-VLAN Classification 1863
10.66 Catalyst 3560 Per-VLAN Classification 1869
10.67 Catalyst QoS Port-Based Policing and Marking 1874

Copyright © 2010 Internetwork Expert

10.68 Catalyst 3560 Per-Port Per-VLAN Policing 1884
10.68 Catalyst 3560 Per-Port Per-VLAN Policing 1884
10.69 Catalyst 3550 Per-Port Per-VLAN Policing 1888
10.70 Catalyst QoS Aggregate Policers 1894
10.71 Catalyst 3560 Ingress Queueing 1902
10.72 Catalyst 3560 Ingress Queue Tuning 1906
10.73 Catalyst 3550 Egress Queueing 1910
10.74 Catalyst 3550 Regular Queues Tuning 1916
10.75 Catalyst 3550 Gigabit Interface Queues Tuning 1918
10.76 Catalyst 3550 Egress Policing 1923
10.77 Catalyst 3560 SRR Shared Mode 1927
10.78 Catalyst 3560 SRR Shaped Mode 1937
10.79 Catalyst 3560 Egress Queues Tuning 1943
10.80 Catalyst QoS DSCP Mutation 1949
10.81 Advanced HTTP Classification with NBAR 1953

Security Solutions 1961
11.1 AAA Authentication Lists 1961
11.2 AAA Exec Authorization 1967
11.3 AAA Local Command Authorization 1973
11.4 Traffic Filtering using Standard Access-Lists 1979
11.5 Traffic Filtering using Extended Access-Lists 1985
11.6 Traffic Filtering using Reflexive Access-Lists 1993
11.7 Filtering Fragmented Packets 1997
11.8 Filtering Packets with Dynamic Access-Lists 2001
11.9 Filtering Traffic with Time-Based Access Lists 2007
11.10 Traffic Filtering with Policy Based Routing 2009
11.11 Preventing Packet Spoofing with uRPF 2011
11.12 Using NBAR for Content-Based Filtering 2015
11.13 TCP Intercept 2019
11.14 TCP Intercept Watch Mode 2025
11.15 Packet Logging with Access-Lists 2029
11.16 Stateful Filtering with CBAC 2033
11.17 Advanced CBAC Features 2039
11.18 CBAC TCP/UDP Intercept Feature 2043
11.19 VLAN Filtering for IP Traffic 2047
11.20 VLAN Filters for Non-IP Traffic 2049
11.21 Port Security 2053
11.22 HSRP and Port-Security 2059
11.23 DHCP Snooping 2063
11.24 DHCP Snooping and the Information Option 2069
11.25 Dynamic ARP Inspection 2073
11.26 IP Source Guard 2079
11.27 Using Catalyst Ingress Access-Lists 2083
11.28 Controlling Terminal Line Access 2085
11.29 IOS Login Enhancements 2087

Copyright © 2010 Internetwork Expert

11.30 Role Based CLI. 2091
11.31 IP Source Tracker. 2095
11.32 Router IP Traffic Export 2097
11.33 Controlling the ICMP Messages Rate. 2101
11.34 Control Plane Policing 2103
11.35 Control Plane Protection (CPPr). 2109
11.36 IOS ACL Selective IP Option Drop 2121
11.37 BGP Generic TTL Security Mechanism 2123
11.38 Flexible Packet Matching. 2125
11.39 Zone Based Firewall. 2135
11.40 ZFW Rate Limiting. 2155
11.41 ZFW Application Inspection 2161
11.42 Classic IOS Transparent Firewall 2169
11.43 ZFW-Based IOS Transparent Firewall. 2178
11.44 IOS IPS. 2182

System Management Solutions 2191
12.1 Exec Aliases 2191
12.2 System Message Logging. 2195
12.3 Syslog Logging 2199
12.4 Logging Counting and Timestamps 2201
12.5 Logging to Flash Memory. 2205
12.6 Lonfiguration Change Notification and Logging 2207
12.7 Configuration Archive & Rollback. 2209
12.8 Logging with Access-Lists. 2213
12.9 TCP Keepalives 2215
12.10 Generating Exception Core Dumps 2217
12.11 Conditional Debugging 2219
12.12 Telnet Service Options 2221
12.13 Tuning Packet Buffers 2225
12.14 Terminal Line Settings. 2227
12.15 SNMPv2 Server. 2233
12.16 SNMPv2c Access Control. 2235
12.17 SNMP Traps and Informs 2237
12.18 CPU and Memory Thresholds 2239
12.19 SNMPv3 2241
12.20 SNMP MAC Address Notifications. 2247
12.21 SNMP Notifications of Syslog Messages. 2249
12.22 CDP 2251
12.23 RMON Alarms. 2253
12.24 RMON Statistics Collection. 2257
12.25 HTTP Server and Client. 2259
12.26 FTP Server and Client 2263
12.27 TFTP Server and Client. 2265
12.28 Remote Shell 2267
12.29 NTP 2271

Copyright © 2010 Internetwork Expert

12.30 NTP Authentication 2275
12.31 NTP Access Control 2277
12.32 Auto-Install over LAN Interfaces using DHCP 2281
12.33 Auto-Install over Frame-Relay 2287
12.34 Auto-Install over LAN Interfaces using RARP 2291
12.35 IOS Menus 2295
12.36 IOS Banners 2299
12.37 KRON Command Schedule 2303
12.38 EEM Scripting: Interface Events 2305
12.39 EEM Scripting: Syslog Events 2311
12.40 EEM Scripting: CLI Events 2315
12.41 EEM Scripting: Periodic Scheduling 2319
12.42 EEM Scripting: Advanced Features 2321

IP Services Solutions 2325
13.1 Proxy ARP 2325
13.2 DHCP Server 2329
13.3 DHCP Client 2331
13.4 DHCP Relay 2335
13.5 DHCP Host Pools 2337
13.6 DHCP on-Demand Pool 2341
13.7 DHCP Proxy 2343
13.8 DHCP Information Option 2349
13.9 DHCP Authorized ARP 2355
13.10 IP SLA 2357
13.11 Object Tracking 2359
13.12 HSRP 2362
13.13 VRRP 2364
13.14 GLBP 2366
13.15 Router Redundancy and Object Tracking 2372
13.16 IRDP 2376
13.17 Router ICMP Settings 2378
13.18 Basic NAT 2380
13.19 NAT Overload 2384
13.20 NAT with Route Maps 2386
13.21 Static NAT 2390
13.22 Static PAT 2394
13.23 Static NAT and IP Aliasing 2396
13.24 Static Policy NAT 2398
13.25 NAT with Overlapping Subnets 2402
13.26 TCP Load Distribution with NAT 2406
13.27 Stateful NAT with HSRP 2408
13.28 Stateful NAT with Primary/Backup 2414
13.29 NAT Virtual Interface 2416
13.30 NAT Default Interface 2418
13.31 Reversible NAT 2422

Copyright © 2010 Internetwork Expert

13.32 Static Extendable NAT. 2426
13.33 IP Precedence Accounting 2430
13.34 IP Output Packet Accounting. 2432
13.35 IP Access Violation Accounting 2436
13.36 MAC Address Accounting. 2438
13.37 TCP Optimization. 2440
13.38 IOS Small Services & Finger 2444
13.39 Directed Broadcasts & UDP Forwarding. 2446
13.40 DRP Server Agent 2450
13.41 WCCPv1 Web-Cache 2452
13.42 WCCPv2 Services 2456
13.43 NBAR Protocol Discovery. 2458
13.44 Netflow Ingress & Egress 2462
13.45 Netflow Top Talkers. 2466
13.46 Netflow Aggregation Cache 2468
13.47 Netflow Random Sampling 2472
13.48 Netflow Input Filters. 2476
13.49 IOS Authoritative DNS Server 2480
13.50 IOS Caching DNS Server 2484
13.51 IOS DNS Spoofing. 2486
13.52 IP Event Dampening. 2490

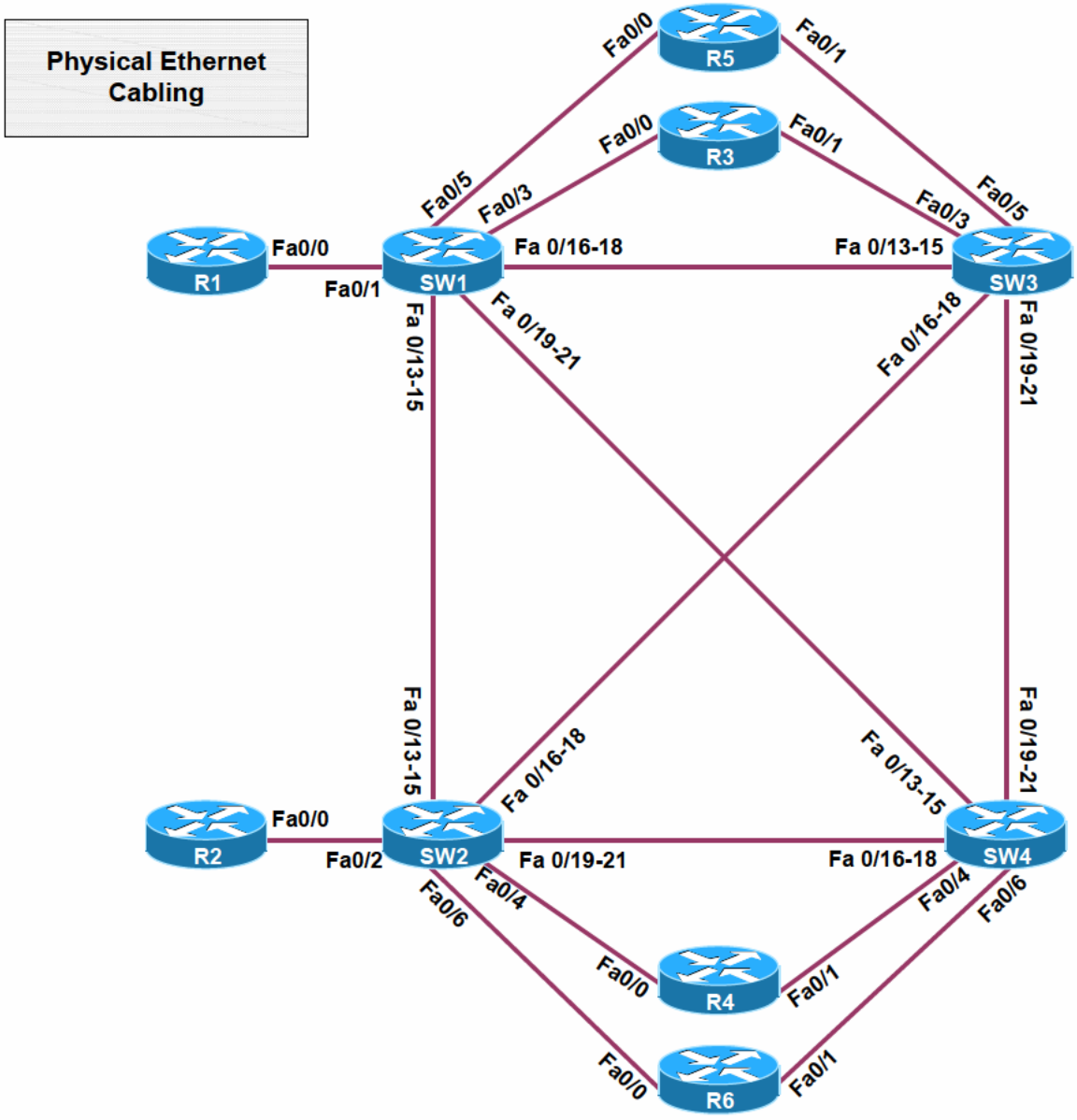
MPLS VPN Solutions. 2492
14.1 VRF Lite. 2492
14.2 MPLS LDP 2498
14.3 MPLS Label Filtering. 2504
14.4 MP-BGP VPNv4. 2506
14.5 MP-BGP Prefix Filtering. 2516
14.6 PE-CE Routing with RIP 2520
14.7 PE-CE Routing with OSPF. 2526
14.8 OSPF Sham-Link. 2534
14.9 PE-CE Routing with EIGRP 2542
14.10 EIGRP Site-of-Origin 2550
14.11 PE-CE Routing with BGP 2558
14.12 BGP SoO Attribute. 2562
14.13 Internet Access. 2566
14.14 AToM. 2572
14.15 L2TPV3. 2576
14.16 MPLS VPN Performance Tuning 2582

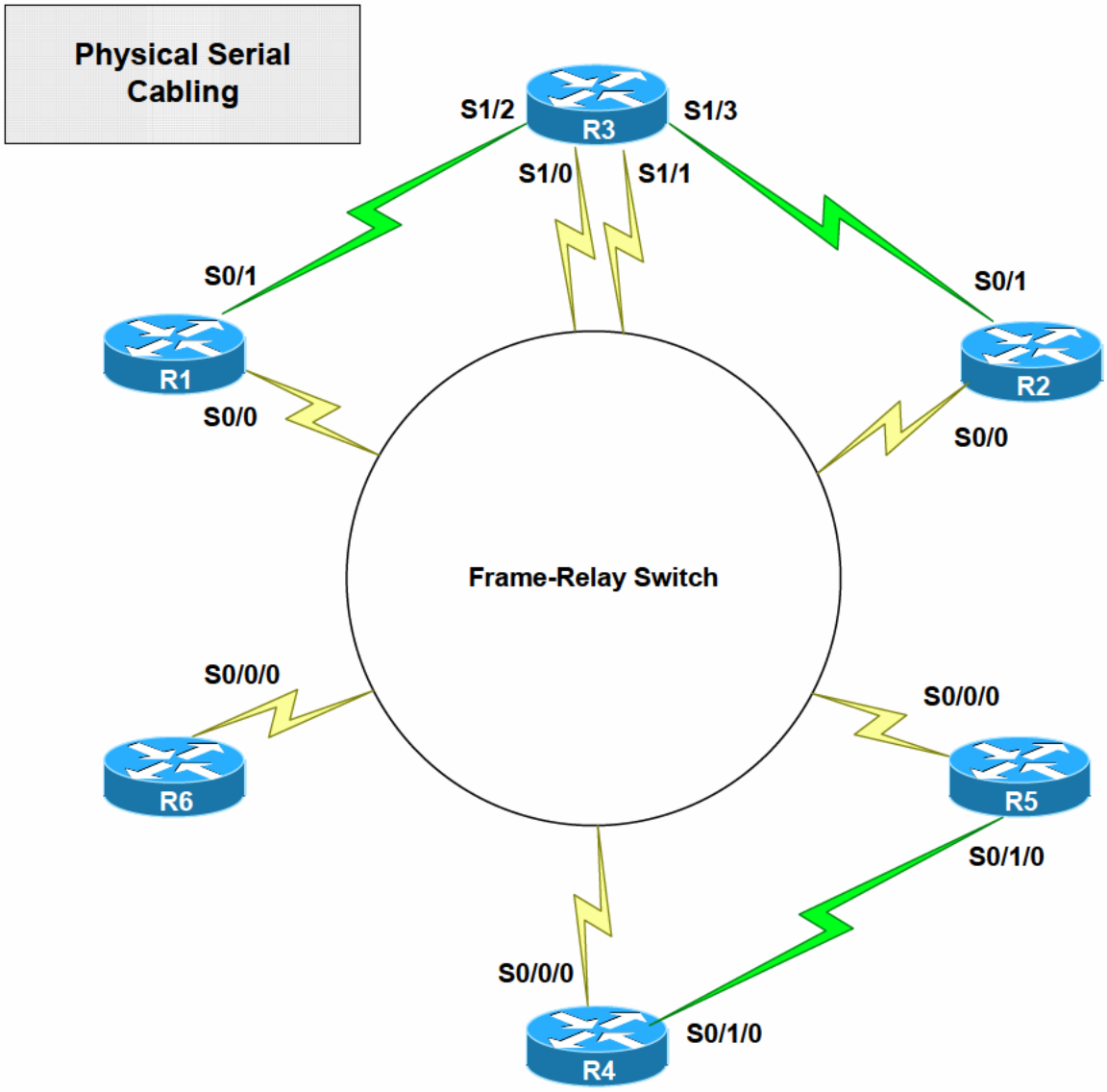
Copyright © 2010 Internetwork Expert

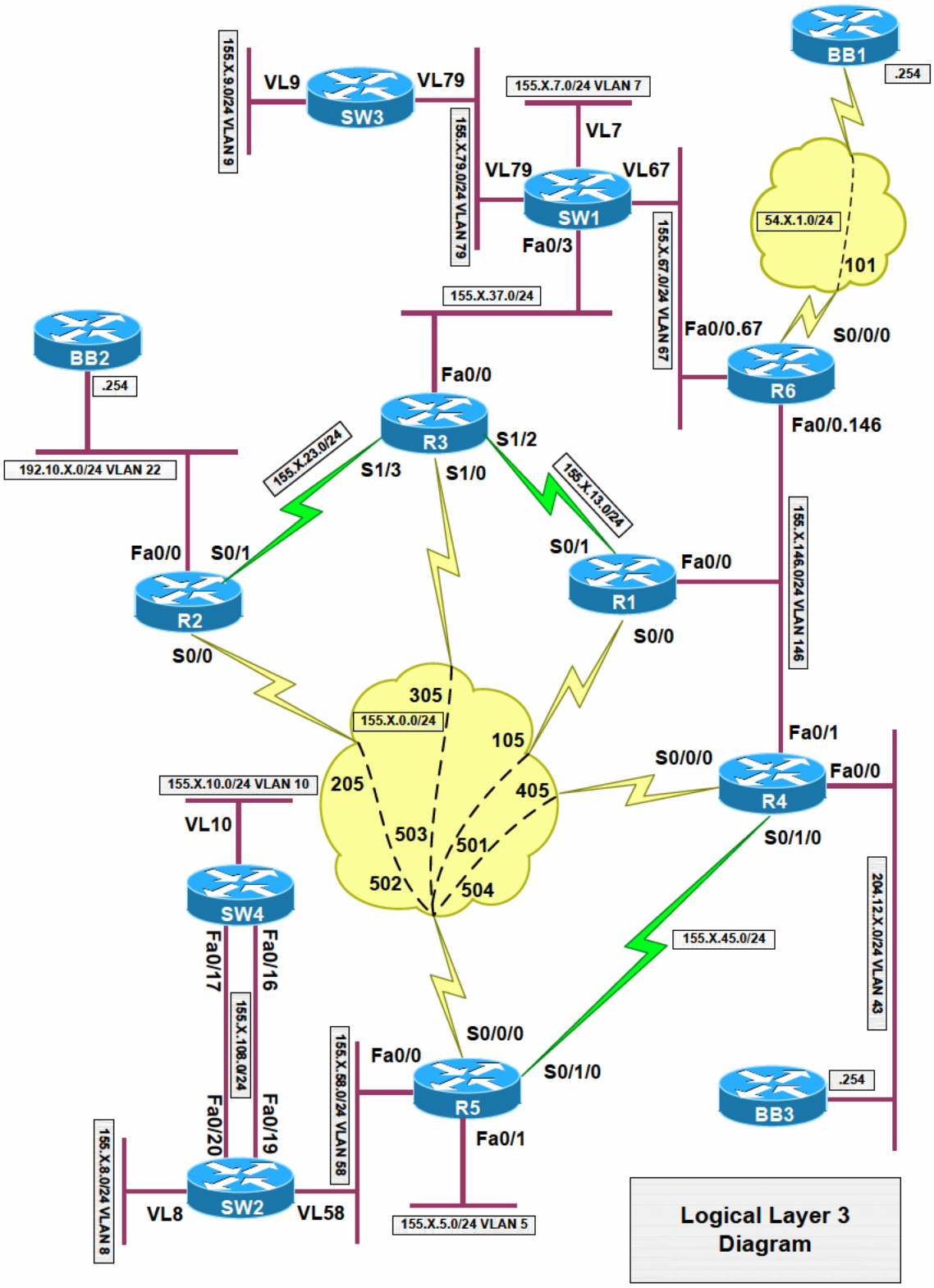
Copyright © 2010 Internetwork Expert

Workbook Diagrams

There are three main diagram supplied with this workbook – two physical cabling diagrams and the logical Layer 3 addressing diagram. Those two are supposed to be used together to give you the complete understanding of the network topology. There are no separate diagrams per section generally, only for those that have some pre-configurations, such as parts of BGP and Multicast sections.




Physical Serial Cabling
R3
S1/2
S1/3
S1/0
S1/1
R1
S0/1
S0/0
R2
S0/1
S0/0
Frame-Relay Switch
R6
S0/0/0
R5
S0/0/0
S0/1/0
R4
S0/0/0
S0/1/0



Copyright © 2010 Internetwork Expert

You are assumes these three main diagrams as foundation for every section in this workbook. We highly recommend you re-drawing the Logical Layer 3 diagram and extending it as appropriate for every section, e.g. adding routing protocol domains and additional addressing, if used. Keep in mind that some sections, such as those centered around Layer 2 technologies, may not make use of Layer 3 diagram at all, as they concentrate mainly on bridging and switching topics.

How would you use these diagrams? Suppoese you are required to apply some policy to the switch port connection to R6's VLAN146 interface. Using Layer 3 logical diagram you locate this VLAN and find out that R6's port Fa 0/1 belongs to it. Next, using the Ethernet Physical Cabling Diagram you find out that R6's Fa 0/1 is connected to SW4 port Fa 0/6. This is the physical port you are looking for to apply the switchport policy.

Copyright © 2010 Internetwork Expert

Bridging & Switching

> **Note**
>
> Load the *Basic IP Addressing* initial configurations prior to starting.

1.1 Layer 2 Access Switchports

- Using the diagram for reference configure access VLAN assignments on SW1, SW2, SW3, and SW4 to obtain basic connectivity between the devices with Ethernet segments with the exception of R6.
- Do not use VTP to accomplish this.

1.2 Layer 2 Dynamic Switchports

- Configure all inter-switch links on SW2, SW3, and SW4 to be in dynamic auto state.
- Configure all inter-switch links on SW1 to be in dynamic desirable state.
- Using the CAM table verify that all layer 2 traffic between devices in the same VLAN, but not attached to the same switch, is transiting SW1.

1.3 ISL Trunking

- Statically set the trunking encapsulation of SW1's inter-switch links to ISL.
- Verify that SW2, SW3, & SW4 are negotiating ISL as the trunking encapsulation to SW1, and that SW1 is not negotiating ISL to SW2, SW3, and SW4.

1.4 802.1q Trunking

- Change the trunking encapsulation on SW1's inter-switch links from static ISL to static 802.1q.
- Verify that SW2, SW3, & SW4 are negotiating 802.1q as the trunking encapsulation to SW1, and that SW1 is not negotiating 802.1q to SW2, SW3, and SW4.

1.5 802.1q Native VLAN

- Modify the native VLAN on the 802.1q trunks of SW1 so that traffic between devices in VLAN 146 is not tagged when sent over the trunk links.

Copyright © 2010 Internetwork Expert

1.6 Disabling DTP Negotiation

- Disable Dynamic Trunking Protocol on the trunk links of SW1.
- Verify that trunking is still occurring between SW1 & SW2, SW1 & SW3, and SW1 & SW4 without the use of DTP.

1.7 Router-On-A-Stick

- Configure the link between SW2 and R6 as an 802.1q trunk link.
- Using the subinterfaces listed in the diagram configure R6 to route traffic for both VLANs 67 and 146 on its Ethernet link.
- Verify that R6 has reachability to devices both on VLAN 67 and 146.

> **Note**
>
> Erase and reload SW1, SW2, SW3, & SW4, and load the *Basic IP Addressing* initial configurations before continuing.

1.8 VTP

- Configure all inter-switch links on SW2, SW3, and SW4 to be in dynamic auto state.
- Configure all inter-switch links on SW1 to be in dynamic desirable state.
- Configure SW2 as a VTP server in the domain CCIE.
- Configure SW1, SW3, and SW4 as VTP clients in the domain CCIE.
- Configure necessary VLAN definitions on SW2 using the diagram for reference.
- Configure access VLAN assignments on SW1, SW2, SW3, and SW4 to obtain basic connectivity between the devices with Ethernet segments.
- Configure router-on-a-stick between SW2 and R6 per the diagram so R6 has reachability to devices on VLANs 67 and 146.

1.9 VTP Transparent

- Configure SW1 in VTP transparent mode and remove all previous VLAN definitions on it.
- Configure SW1 with only the VLAN definitions necessary to obtain basic connectivity between the devices with Ethernet segments.

Copyright © 2010 Internetwork Expert

1.10 VTP Pruning

- Configure SW1 in VTP client mode.
- Enable VTP pruning in the layer 2 network so that inter-switch broadcast replication is minimized.
- Verify this configuration is functional through the **`show interface trunk`** output.

1.11 VTP Prune-Eligible List

- Edit the prune-eligible list to ensure that traffic for VLAN 7 is carried on all active trunk links in the layer 2 network.
- Verify this configuration is functional through the **`show interface trunk`** output.

1.12 Layer 2 EtherChannel

- Remove all previous configurations on the links connecting SW1, SW2, SW3, and SW4.
- Configure all inter-switch links on SW2, SW3, and SW4 to be in dynamic auto state.
- Configure all inter-switch links on SW1 to be in dynamic desirable state.
- Configure Layer 2 EtherChannels on all inter-switch links between SW1 & SW2, SW1 & SW3, and SW1 & SW4.
- Use Port-Channel numbers 12, 13, and 14 respectively.
- These links should not use dynamic EtherChannel negotiation.

1.13 Layer 2 EtherChannel with PAgP

- Modify the previous EtherChannel configuration to use PAgP for dynamic negotiation.
- SW1 should initiate negotiation and the other devices should respond.

1.14 Layer 2 EtherChannel with LACP

- Modify the previous EtherChannel configuration to use LACP for dynamic negotiation.
- SW1 should initiate negotiation and the other devices should respond.

Copyright © 2010 Internetwork Expert

1.15 Layer 3 EtherChannel

- Configure links Fa0/16 & Fa0/17 on SW4 and links Fa0/19 & Fa0/20 on SW2 to be bound together as a Layer 3 EtherChannel.
- Use Port-Channel number 24 and the subnet 155.X.108.0/24 per the diagram.
- Ensure IP reachability is obtained between these devices over the segment.

> **Note**
>
> Erase and reload SW1, SW2, SW3, & SW4 before continuing.

1.16 802.1q Tunneling

- Configure 802.1q trunk links between SW1 & SW2's interfaces Fa0/13, SW2's interface Fa0/16 & SW3's interface Fa0/16, and SW3's interface Fa0/19 & SW4's interface Fa0/19.
- Disable all other inter-switch links.
- Configure two Ethernet subinterfaces on R1 with the IP addresses 14.0.0.1/24 and 41.0.0.1/24 using VLANs 14 and 41 respectively.
- Configure two Ethernet subinterfaces on R4's second Ethernet interface1 with the IP addresses 14.0.0.4/24 and 41.0.0.4/24 using VLANs 14 and 41 respectively.
- Using VLAN 100 configure an 802.1q tunnel between SW1 and SW4 to connect R1 and R4.
- R1 and R4 should appear to be directly connected when viewing the **`show cdp neighbor`** output.

Copyright © 2010 Internetwork Expert

1.17 EtherChannel over 802.1q Tunneling

- Remove the previous trunking and tunneling configuration.
- Configure an 802.1q trunk link between SW2 and SW3.
- Configure interfaces Fa0/13, Fa0/14, and Fa0/15 on SW1 as a layer 2 EtherChannel using PAgP for negotiation.
- Configure interfaces Fa0/19, Fa0/20, and Fa0/21 on SW4 as a layer 2 EtherChannel using PAgP for negotiation.
- Disable all other inter-switch links on SW1 and SW4.
- Configure SW2 and SW3 to tunnel the EtherChannel link between SW1 and SW4 using VLANs 100, 200, and 300.
- Tunnel Spanning-Tree Protocol along with CDP over these links so that SW1 and SW4 appear to be directly connected when viewing the **`show cdp neighbor`** output.
- SW1 and SW4 should form an 802.1q trunk link over this EtherChannel.
- To verify this configure SW1 and SW4's links to R1 and R4 in VLAN 146 per the diagram and ensure connectivity between R1 and R4.

> **Note**
>
> Erase and reload SW1, SW2, SW3, & SW4, and load the *Basic IP Addressing* initial configurations before continuing.

1.18 STP Root Bridge Election

- Configure the inter-switch links between SW1 & SW2, SW1 & SW3, SW2 & SW4, and SW3 & SW4 as 802.1q trunk links.
- Disable all other inter-switch links.
- Configure SW4 as a VTP server using the domain name CCIE with SW1, SW2, and SW3 as its clients.
- Configure VLAN assignments per the diagram.
- Configure SW1 as the STP Root Bridge for all active VLANs.
- If SW1 goes down SW4 should take over as the STP Root Bridge for all active VLANs.

Copyright © 2010 Internetwork Expert

1.19 STP Load Balancing with Port Cost

- Using Spanning-Tree cost modify the layer 2 transit network so that traffic for all active VLANs from SW2 to SW1 uses the last link between SW2 and SW4.
- If this link goes down traffic should fall over to the second link between SW2 and SW4.

1.20 STP Load Balancing with Port Priority

- Using Spanning-Tree priority modify the layer 2 transit network so that traffic for all active VLANs from SW4 to SW1 uses the last link between SW3 and SW4.
- If this link goes down traffic should fall over to the second link between SW3 and SW4.

1.21 Tuning STP Convergence Timers

- Configure the switches so that they broadcast Spanning-Tree hello packets every three seconds.
- When a new port becomes active it should wait twenty seconds before transitioning to the forwarding state.
- If the switches do not hear a configuration message within ten seconds they should attempt reconfiguration.
- This configuration should impact all currently active VLANs and any additional VLANs created in the future.

1.22 STP PortFast

- Configure Spanning-Tree PortFast on the switches so that ports connected to the internal and external routers do not have to wait for the Spanning-Tree listening and learning phases to begin forwarding.
- Do not use any global Spanning-Tree commands to accomplish this.

1.23 STP PortFast Default

- Remove the previous PortFast configuration.
- Configure Spanning-Tree PortFast on the switches so that ports connected to the internal and external routers do not have to wait for the Spanning-Tree listening and learning phases to begin forwarding.
- Do not use any interface level Spanning-Tree commands to accomplish this.

Copyright © 2010 Internetwork Expert

1.24 STP UplinkFast

- Configure SW2, SW3, and SW4 with Spanning-Tree UplinkFast such that if their root port is lost they immediately reconverge to an alternate connection to their upstream bridge.
- Verify this by shutting down the root port of SW2.

1.25 STP BackboneFast

- Configure Spanning-Tree BackboneFast such that if the links between SW3 and SW4 go down SW2 immediately expires its maxage timer and begins Spanning-Tree reconvergence.

1.26 STP BPDU Guard

- Configure Spanning-Tree BPDU Guard on the switches so that ports connected to the internal and external routers are disabled if a Spanning-Tree BPDU is detected.
- Once disabled the switches should attempt to re-enable the ports after two minutes.
- Do not use the global `portfast` command to accomplish this.

1.27 STP BPDU Guard Default

- Remove the previous BPDU Guard configuration.
- Configure Spanning-Tree PortFast on the switches so that ports connected to the internal and external routers do not have to wait for the Spanning-Tree listening and learning phases to begin forwarding.
- Configure Spanning-Tree BPDU Guard so that if a Spanning-Tree BPDU is detected on any of these ports they are disabled.
- Do not use any interface level Spanning-Tree commands to accomplish this.

1.28 STP BPDU Filter

- Remove the previous BPDU Guard configuration.
- Configure the switches so that ports connected to the internal and external routers do not send Spanning-Tree packets sent out them.
- Do not use any global Spanning-Tree commands to accomplish this.

Copyright © 2010 Internetwork Expert

1.29 STP BPDU Filter Default

- Remove the previous BPDU Filter configuration.
- Configure Spanning-Tree PortFast on the switches so that ports connected to the internal and external routers do not have to wait for the Spanning-Tree listening and learning phases to begin forwarding.
- Configure Spanning-Tree BPDU Filter on the switches so that the PortFast enabled ports are reverted out of PortFast state if a Spanning-Tree packet is received in them.
- Do not use any interface level Spanning-Tree commands to accomplish this.

1.30 STP Root Guard

- Configure SW1 so that the links to either SW2 or SW3 are disabled if either SW2, SW3, or SW4 is elected the Spanning-Tree Root Bridge for any VLAN.

1.31 STP Loop Guard

- Configure Spanning-Tree Loop Guard to prevent unidirectional links from forming on any of the inter-switch links in the layer 2 network.

1.32 Unidirectional Link Detection

- Remove the previous Loop Guard configuration.
- Configure UDLD to prevent unidirectional links from forming on any of the inter-switch links in the layer 2 network.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Erase and reload SW1, SW2, SW3, & SW4, and load the *Basic IP Addressing* initial configurations before continuing.

1.33 MST Root Bridge Election

- Configure the inter-switch links between SW1 & SW2, SW1 & SW3, SW2 & SW4, and SW3 & SW4 as 802.1q trunk links.
- Disable all other inter-switch links.
- Configure SW4 as a VTP server using the domain name CCIE with SW1, SW2, and SW3 as its clients.
- Configure VLAN assignments per the diagram.
- Configure Multiple Spanning-Tree on the switches.
- Instance 1 should service VLANs 1 - 100.
- Instance 2 should service VLANs 101 - 200.
- Instance 3 should service all other VLANs.
- Configure SW1 as the STP Root Bridge for instance 1.
- Configure SW4 as the STP Root Bridge for instance 2.
- If SW1 goes down SW2 should take over as the STP Root Bridge for instance 1.
- If SW4 goes down SW3 should take over as the STP Root Bridge for instance 2.

1.34 MST Load Balancing with Port Cost

- Using Spanning-Tree cost modify the layer 2 transit network so that traffic for MST instance 1 from SW2 to SW1 uses the last link between SW2 and SW4.
- If this link goes down traffic should fall over to the second link between SW2 and SW4.

1.35 MST Load Balancing with Port Priority

- Remove the previous STP cost modifications.
- Set the cost for MST instance 1 on SW3's links to SW1 to be 100,000.
- Using Spanning-Tree priority modify the layer 2 transit network so that traffic for MST instance 1 from SW4 to SW1 uses the last link between SW3 and SW4.
- If this link goes down traffic should fall over to the second link between SW3 and SW4.

Copyright © 2010 Internetwork Expert

1.36 MST and Rapid Spanning Tree

- Configure Rapid Spanning-Tree on the switches so that ports connected to the internal and external routers immediately begin forwarding when enabled.

1.37 Protected Ports

- Create a new SVI for VLAN22 on SW2 and assign it the IP address 192.10.X.8/24, where X is your rack number.
- Configure port protection on SW2 so that R2 and BB2 cannot directly communicate with each other, but can communicate with SW2's VLAN22 interface.

1.38 Storm Control

- Configure SW1 to limit unicast traffic received from R1 to 100 pps.
- Configure SW1 to limit broadcast traffic received from R6 to 10Mbps.
- Configure SW1 to limit broadcast traffic received from R4 to 1Mbps using a relative percentage of the interface bandwidth.

1.39 MAC-Address Table Static Entries & Aging

- Ensure reachability on VLAN 146 between R1, R4, and R6.
- Configure a static CAM entry on SW4 so that frames destined to the MAC address of R4's interface connected to VLAN 146 are dropped; once complete R1 and R6 should have reachability to each other, but not R4.
- Configure static CAM entry for that MAC address of R6's connection to VLAN 146 to ensure that this address is not allowed to roam.

1.40 SPAN

- Configure SW1 so that all traffic transiting VLAN 146 is redirected to a host located on port Fa0/24.
- Configure SW4 so that all traffic coming from and going to R4's connection to VLAN 146 is redirected to a host located on port Fa0/24; Inbound traffic from the Linux host should be placed into VLAN 146.

Copyright © 2010 Internetwork Expert

1.41 RSPAN

- Disable the trunk links between SW1 and SW2.
- Create VLAN 500 as an RSPAN VLAN on all switches in the topology.
- Configure SW2 so that traffic received from and sent to R4's connection to VLAN 43 is redirected to the RSPAN VLAN.
- Configure SW1 to receive traffic from the RSPAN VLAN and redirect it to a host connected to port Fa0/24.
- Inbound traffic on the link connected to this host should be placed in VLAN 146.

1.42 Voice VLAN

- Ports Fa0/2, Fa0/4, and Fa0/6 on SW1 will be connected to Cisco IP phones in the near future.
- Configure port Fa0/2 with an access VLAN assignment of 146 and a voice VLAN assignment of 600.
- Enable Spanning-Tree portfast on this link and ensure that CDP is enabled.
- Configure port Fa0/4 as an 802.1q trunk link.
- Configure SW1 so that only VLANs 146 and 600 are permitted on this switchport, so that STP BPDUs received on the port are filtered out, and so that the interface runs in STP portfast mode.
- Configure VLAN 146 as the native VLAN for this port and so that VLAN 600 is advertised as the voice VLAN via CDP.
- Configure port Fa0/6 with an access VLAN assignment of 146, and for voice VLAN frames to use dot1p tagging.

1.43 IP Phone Trust and CoS Extend

- Enable MLS QoS globally on SW1.
- Configure SW1 to trust the CoS of frames received on the ports connected to the IP phones.
- This trust should only occur if the Cisco IP phone is present and advertises itself via CDP.
- SW1 should enforce a CoS value of 1 to any appliance connected to the second port of the IP phone.

Copyright © 2010 Internetwork Expert

1.44 Smartport Macros

- Configure a macro on SW1 named VLAN_146 that when applied to an interface will set it to be an access switchport, apply VLAN 146 as the access vlan, and filter Spanning-Tree BPDUs.
- Apply this macro to ports Fa0/7 and Fa0/8 on the switch.

> **Note**
>
> Erase and reload all devices to a blank configuration before continuing.

1.45 Flex Links

- Configure links Fa0/16 between SW2 and SW3 as an 802.1q trunk.
- Configure link Fa0/16 on SW1 and Fa0/13 on SW3 as an 802.1q trunk.
- Configure links Fa0/13 & Fa0/14 between SW1 and SW2 as an 802.1q trunked EtherChannel.
- Disable all other inter-switch links.
- Configure R1's Ethernet interface with the IP address 10.0.0.1/24, R2's Ethernet interface with the IP address 10.0.0.2/24, and R3's second Ethernet interface with the IP address 10.0.0.3/24.
- Configure flex links on SW1 so that traffic from R1 to R3 uses the EtherChannel to SW2.
- If the EtherChannel goes down traffic should immediately switch over to use the link between SW1 and SW3.
- If the EtherChannel and all its members comes back up traffic should forward back over this link after 20 seconds.

1.46 Fallback Bridging

- Configure R4's second Ethernet interface with the IP address 104.0.0.4/24, and with the IPv6 address 2001::4/24.
- Configure R6's second Ethernet interface with the IP address 106.0.0.6/24, and with the IPv6 address 2001::6/24.
- Configure interface VLAN104 on SW4 with the IP address 104.0.0.10/24, and configure interface Fa0/4 in VLAN 104.
- Configure interface Fa0/6 on SW4 with the IP address 106.0.0.10/24.
- Enable RIPv2 on all of these links.
- Configure fallback bridging on SW4 to bridge the IPv6 subnet of R4 and R6 together.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Erase and reload all devices to a blank configuration before continuing.

1.47 Private VLANs

- Configure the first Ethernet interfaces of R1, R2, R3, R4, R5, and R6 with IP addresses 100.0.0.Y/24, where Y is the device number.
- Configure the first inter-switch link between SW1 and SW2 as a trunk.
- Configure the primary VLAN 100 to service private VLANs 1000, 2000, and 3000.
- VLANs 1000 and 2000 should be community VLANs, while VLAN 3000 should be an isolated VLAN.
- Assign VLAN 1000 to the links connecting to R2 & R3, VLAN 2000 to the links connecting to R4 & R5, and VLAN 3000 to R6.
- The link connecting to R1 should be a promiscuous port.
- Ensure that R1 can reach all devices, R2 can reach R3, and R4 can reach R5.
- No other connectivity should be allowed within this topology.

1.48 PPP

- Enable PPP encapsulation for the Serial link connecting R4 and R5 and use the IP subnet 155.X.45.0/24 for this link
- R4 should attempt to authenticate R5 using PAP and than CHAP. R5 should refuse PAP authentication and use CHAP.
- Make sure R4 uses an alternate CHAP hostname R4CHAP.
- Use the name R5PPP and the password of CISCO to accomplish this.
- R5 should authenticate R4 using PAP only. R4 should use the name R4PPP and the password of CISCO.

Copyright © 2010 Internetwork Expert

1.49 PPP AAA Authentication

- Reconfigure R4 so that it attempts to authenticate R5 against the RADIUS server at the IP address 155.X.146.100.
- Reconfigure R5 so that it attempts to authenticate R4 against the TACACS+ server at the IP address 155.X.146.200.
- In case if the remote AAA servers fail both R4 and R5 should use the local users database for authentication.
- Use the password value of CISCO to authenticate with the remote servers.
- Do not use global configuration commands to define a RADIUS server in R4.

1.50 PPPoE

- Configure R3 as a PPPoE client and R5 as the PPPoE server.
- R5 should block any host attempting to initiate more than 10 sessions in a minute for 5 minutes.
- Use R3's interface Fa0/1 and R5's sub-interface Fa0/1.54 to accomplish this.
- You are allowed to create an additional VLAN 54 for this task.
- R5 should allocate an IP address to R3 from the local DHCP pool for the subnet 155.X.35.0.
- Make sure R3 is assigned the IP address 155.X.35.3 from the pool.
- R5 should authenticate R3 using CHAP and the password value of CISCO.

Copyright © 2010 Internetwork Expert

Frame Relay

> **Note**
>
> Erase and reload all devices to a blank configuration prior to starting.

2.1 Inverse-ARP

- Configure the IP address of R6's Serial interface using the information in the diagram.
- Enable Frame Relay on this link.
- Ensure R6 learns an IP to DLCI mapping dynamically for PVC 101 connected to BB1, and that these devices have unicast, multicast, and broadcast reachability to each other.

2.2 Static Mappings

- Configure the IP addresses of R1 and R5's Serial interfaces connected to the Frame Relay cloud using the information in the diagram.
- Enable Frame Relay on this link and configure static IP to DLCI mappings so that R1 and R5 have unicast, multicast, and broadcast reachability to each other.

2.3 Point-to-Point Subinterfaces

- Enable Frame Relay on R2's link connecting to the Frame Relay cloud.
- Create a point-to-point subinterface numbered 205 on R2 with the IP address 155.X.25.2/24 and assign DLCI 205 to it.
- Create a point-to-point subinterface numbered 502 on R5 with the IP address 155.X.25.5/24 and assign DLCI 502 to it.
- Ensure that these devices have unicast, multicast, and broadcast reachability to each other on this segment.

Copyright © 2010 Internetwork Expert

2.4 Multipoint Subinterfaces & Inverse-ARP

- Enable Frame Relay on R3 and R4's connections to the Frame Relay cloud.
- Create a multipoint subinterface numbered 100 on R3 with the IP address 155.X.100.3/24 and assign DLCIs 304 and 305 to it.
- Create a multipoint subinterface numbered 100 on R4 with the IP address 155.X.100.4/24 and assign DLCIs 403 and 405 to it.
- Create a multipoint subinterface numbered 100 on R5 with the IP address 155.X.100.5/24 and assign DLCIs 503 and 504 to it.
- Ensure that R3, R4, and R5 all learn dynamic IP to DLCI mappings for each other on this segment and have unicast, multicast, and broadcast reachability to each other.

2.5 Multipoint Subinterfaces & Static Mappings

- Modify the previous configuration between R3, R4, and R5 to use static IP to DLCI mappings as opposed to dynamic mappings.
- Ensure that these devices have unicast, multicast, and broadcast reachability to each other on this segment.

> **Note**
>
> Erase and reload all devices to a blank configuration before continuing.

2.6 Hub-and-Spoke & Static Mappings

- Enable Frame Relay on the Serial interfaces of R1, R3, and R5 connected to the Frame Relay switch.
- Assign IP addresses to these interfaces per the diagram.
- Configure static IP to DLCI mappings on these devices to gain reachability to each other using only the DLCI assignments shown in the diagram.
- Ensure that R1 & R5 and R3 & R5 have unicast, multicast, and broadcast reachability to each other, and that unicast traffic between R1 & R3 transits through R5.

Copyright © 2010 Internetwork Expert

2.7 Hub-and-Spoke & Inverse-ARP

- Reset the Frame Relay interfaces of R1, R3, and R5 to its default configuration.
- Enable Frame Relay on the Serial interfaces of R1, R3, and R5 connected to the Frame Relay switch.
- Assign IP addresses to these interfaces per the diagram.
- Configure the network in such a way that R1 & R5 and R3 & R5 resolve each other's IP addresses dynamically through Inverse-ARP, while R1 and R3 resolve each other's addresses statically.
- Ensure that R1 & R5 and R3 & R5 have unicast, multicast, and broadcast reachability to each other, that unicast traffic between R1 & R3 transits through R5.

2.8 Hub-and-Spoke & Point-to-Point Subinterfaces

- Reset the Frame Relay interfaces of R1, R3, and R5 to its default configuration.
- Enable Frame Relay on the Serial interfaces of R1, R3, and R5 connected to the Frame Relay switch.
- Configure a point-to-point subinterface numbered 105 on R1 and assign it the IP address 155.X.0.1/24 & DLCI 105.
- Configure a point-to-point subinterface numbered 305 on R3 and assign it the IP address 155.X.0.3/24 & DLCI 305.
- Assign the IP address 155.X.0.5 to R5's main Serial interface and configure static IP to DLCI mappings for R1 and R3.
- Ensure that R1 & R5 and R3 & R5 have unicast, multicast, and broadcast reachability to each other, that unicast traffic between R1 & R3 transits through R5.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Erase and reload all devices to a blank configuration before continuing.

2.9 Disabling Inverse-ARP

- Enable Frame Relay on R1, R2, R3, R4, and R5's interfaces connected to the Frame Relay cloud.
- Configure R1's interface with the IP address 155.X.100.1/24.
- Configure R2's interface with the IP address 155.X.100.2/24.
- Configure R3's interface with the IP address 155.X.200.3/24.
- Configure R4's interface with the IP address 155.X.200.4/24.
- Create a multipoint subinterface numbered 100 on R5 with the IP address 155.X.100.5/24 and the DLCIs 501 & 502.
- Create a multipoint subinterface numbered 200 on R5 with the IP address 155.X.200.5/24 and the DLCIs 503 & 504.
- Configure R1 and R2 so that they will not send Inverse-ARP requests for IP out any DLCIs learned from the Frame Relay cloud other than those in the diagram.
- Configure R3 and R4 to disable Inverse-ARP on DLCIs learned from the Frame Relay cloud that are not listed in the diagram by assigning them to the subinterface 999.
- Ensure that R5 has unicast, multicast, and broadcast reachability to R1, R2, R3, and R4.
- R1 and R2 should be able to reach R5, but not each other.
- R3 and R4 should be able to reach R5, but not each other.
- R1, R2, R3, and R4 should not be able to reach each other.

2.10 Back-to-Back Frame Relay

- Configure IP addresses on the directly connected Serial link between R4 and R5 using the information in the diagram.
- Configure Frame Relay PVC 514 on this link without using LMI.
- Configure static IP to DLCI mappings on R4 and R5 to obtain reachability on this segment.

Copyright © 2010 Internetwork Expert

2.11 Frame Relay End-to-End Keepalives

- Create a Frame Relay map-class called FREEK on R4 and R5.
- Configure R4 to respond to Frame Relay keepalives, and R5 to request them.
- Set R5 to poll every 5 seconds with an event window of 10 and an error threshold of 8.
- R4 should apply this configuration to its main Serial0/0 interface, while R5 should apply this just to the DLCI connecting to R4.

2.12 Frame Relay Broadcast Queue

- Tune the Frame Relay broadcast queue size between R4 and R5 to 100 packets.

2.13 Frame Relay TCP & RTP Header Compression

- Enable RTP header compression for DLCIs 503 and 305 on R5 and R3 respectively.
- Enable TCP header compression for DLCIs 501 and 105 on R5 and R1 respectively.

> **Note**
>
> Erase and reload all devices to a blank configuration before continuing.

2.14 PPP over Frame Relay

- Create Virtual-Template interfaces on R1 and R5 using the IP addresses 155.X.0.1/24 and 155.X.0.5/24 respectively.
- Enable Frame Relay on R1 and R5's Serial interfaces attached to the Frame Relay cloud, and attach the Virtual-Templates to the DLCIs connecting these devices.
- Ensure that R1 and R5 have unicast, multicast, and broadcast reachability to each other over this segment.

Copyright © 2010 Internetwork Expert

2.15 Bridging over Frame Relay

- Remove the previous PPPoFR configuration on R5.
- Configure ports Fa0/2 and Fa0/24 on SW2 in VLAN 100.
- Configure SW1's interface Fa0/5 with the IP address 192.10.X.7/24.
- Enable Frame Relay on the Serial interfaces connecting to the Frame Relay cloud on R2 and R5.
- Disable IP routing on R2 and R5 and remove any IP addresses from their interfaces.
- Configure an IEEE STP bridge group numbered 1 on R2 & R5, and apply this to their Frame Relay connections and their FastEthernet0/0 interfaces.
- Ensure that SW1 and BB2 have unicast, multicast, and broadcast reachability to each other.

2.16 Frame-Relay Switching

- Configure R3 to provision a switched Frame-Relay link between R1 and R2.
- Use the Serial interfaces connecting R1 and R2 to R3 to accomplish this.
- Use the DLCI numbers 132 and 231 on the links connecting R1 to R3 and R2 to R3 respectively.
- Do not use the **`frame-relay route`** command to accomplish this.
- Do not use Inverse-ARP on R2 to establish IP connectivity.
- Use the IP subnet 155.X.12.0/24 for IP addressing.

Copyright © 2010 Internetwork Expert

IP Routing

> **Note**
>
> Load the *Full Layer 2* initial configurations prior to starting.

3.1 Routing to Multipoint Broadcast Interfaces

- Configure R1 with a static route for R4's Loopback0 network pointing to the IP address of R4.
- Configure R1 with a static route for R6's Loopback0 network pointing to R1's Ethernet interface connecting to VLAN 146.
- Ensure that R1 can ping the Loopback0 interfaces of R4 and R6.
- Verify that R6 is running Proxy ARP by checking the ARP table of R1.
- Disable Proxy ARP on R6's connections to VLAN 146.
- Modify R1's ARP table so that it still has IP reachability to the Loopback0 interface of R6.

3.2 Routing to NBMA Interfaces

- Configure R1 and R2 with default routes pointing out their point-to-point Frame Relay interfaces connecting to R5.
- Configure R5 with a static route for R1's Loopback0 network pointing to the IP address of R1.
- Configure R5 with a static route for R2's Loopback0 network pointing out its main Frame Relay interface connecting to R2.
- Ensure that R1, R2, and R5 can all ping each other's Loopback0 interfaces.

Copyright © 2010 Internetwork Expert

3.3 Longest Match Routing

- Configure R4 and R5 with routes to each other's Loopback0 interfaces via the point-to-point Serial interface between them.
- Configure R4 and R5 to route the rest of the 150.X.0.0/16 network via the Frame Relay connection between them.
- Ensure that traffic between R4 and R5's Loopback0 networks transits the point-to-point link between them, but is rerouted out the Frame Relay connection if the point-to-point link is down.

3.4 Floating Static Routes

- Remove the previous static routes on R4 and R5.
- Configure R4 and R5 with identical static routes for each others' Loopback0 networks out both their Frame Relay connections and the point-to-point connection between them.
- Using administrative distance ensure that traffic between R4 and R5's Loopback0 networks transits the point-to-point link between them, but is rerouted out the Frame Relay connection if the point-to-point link is down.

3.5 Backup Interface

- Modify the administrative distance of R4's static routes to R5's Loopback0 interface to be identical out the Frame Relay and point-to-point interfaces.
- Configure the backup interface feature on R4 to route traffic for R5's Loopback0 out the Frame Relay network unless the PVC to R5 is down.
- If the PVC to R5 is down this traffic should be rerouted out the point-to-point link.
- Ensure the backup link is activate 3 seconds after the main link fails, and deactivated once the main link is active for 60 seconds.

Copyright © 2010 Internetwork Expert

3.6 Reliable Static Routing with Enhanced Object Tracking

- Remove all previous routing configurations.
- Configure static routes on R5 so that it has reachability to the Loopback0 networks of R1 and R4 via the Frame Relay network.
- Configure R1 with a static route to R4's Loopback0 network via the Frame Relay connection to R5.
- Configure R4 with a static route to R1's Loopback0 network via the Ethernet connection to VLAN 146.
- Configure a second static route on R4 for R1's Loopback0 network via the Frame Relay connection to R5 with an administrative distance of 2.
- Configure an IP SLA instance on R4 to ping R1's Ethernet connection to VLAN 146 every 5 seconds.
- Create an enhanced object to track this SLA instance.
- Tie this enhanced object to the static route R4 has for R1's Loopback0 out the connection to VLAN 146.
- Ensure that R4 maintains IP reachability to R1's Loopback0 in the case that R1's Ethernet interface is down.

3.7 Policy Routing

- Remove all previous routing configuration.
- Configure default routes on R4 and R6 pointing to the VLAN 146 connection of R1.
- Configure a default route on R3 pointing to the point-to-point link connecting to R1.
- Configure a default route on R5 pointing to R1 out the Frame Relay network.
- Configure a static route on R3 for R5's Loopback0 network and a static route on R5 for R3's Loopback0 network out the Frame Relay network.
- Create two extended access-lists on R1, one named FROM_R4 and the other named FROM_R6.
- Access-list FROM_R4 should match all IP packets coming from R4, while access-list FROM_R6 should match all IP packets coming from R6.
- Configure policy-routing on R1 so that traffic from R4 is routed out the serial link between R1 and R3, and traffic from R6 is routed out the Frame Relay link to R5.
- Use traceroute on R4 and R6 for R3 and R5's Loopback0 networks to verify that this configuration is functional.

Copyright © 2010 Internetwork Expert

3.8 Reliable Policy Routing

- Remove the previous static and policy routing configurations.
- Remove the point-to-point Frame Relay subinterface on R1 and replace this with a static mapping to R5 on R1's main Serial0/0 interface.
- Configure a default route on R4 pointing to the VLAN 146 connection of R1.
- Configure a default route on R3 pointing to the point-to-point link connecting to R1.
- Configure a default route on R5 pointing to R1 out the Frame Relay network.
- Configure a route for R4's Loopback0 network on R5 via their point-to-point Serial link.
- Configure a route for R5's Loopback0 network on R4 via their point-to-point Serial link.
- Configure R1 and R5 to run CDP over the Frame Relay network with each other.
- Configure an IP SLA instance on R1 that pings R4's connection to VLAN 146 every five seconds.
- Configure policy routing on R1 so that traffic from R3 going to R4's Loopback0 network is sent to R5 over the Frame Relay link, while traffic to R5's Loopback0 network is sent to R4 over VLAN 146.
- Use traceroute to verify that this configuration is functional.
- Modify the R1's policy routing so that if R1 loses R5 as a CDP neighbor traffic from R3 to R4's Loopback0 network is rerouted directly to R4.
- Modify the R1's policy routing so that if R1 loses ICMP reachability to R4 traffic from R3 to R5's Loopback0 network is rerouted directly to R5.
- Use traceroute to verify that this configuration is functional.

Copyright © 2010 Internetwork Expert

3.9 Local Policy Routing

- Create two access-lists named TO_R4 and TO_R5 on R1.
- Access-list TO_R4 should match all IP packets going to the Loopback0 network of R4, while access-list TO_R5 should match all packets going to the Loopback0 network of R5.
- Configure local policy-routing on R1 so that locally generated traffic matched by the list TO_R5 is routed out the Ethernet link to R4, and traffic matched by the access-list TO_R4 is routed out the Frame Relay link to R5.
- Use traceroute on R1 for R4 and R5's Loopback0 networks to verify that this configuration is functional.

3.10 GRE Tunneling

- Remove all previous static and policy routing configurations.
- Enable RIPv2 on all links in the 150.X.0.0 and 155.X.0.0 address space on all internal devices and disable auto-summary.
- Create a GRE tunnel between SW3 and SW4 using the IP addresses 10.34.0.Y/24.
- The tunnel should be sourced from and destined to these devices Loopback0 networks.
- Create new Loopback1 interfaces on SW3 and SW4 with the IP addresses 172.16.0.Y/32.
- Configure static routes so that traffic from SW3 to SW4's new Loopback1 interface is routed over the tunnel, and vice-versa.
- Use traceroute on SW3 and SW4 to verify that this configuration is functional.

3.11 GRE Tunneling and Recursive Routing

- Enable RIPv2 on the tunnel interface between SW3 and SW4.
- Configure distribute-list filtering on SW3 and SW4 so that a recursive routing error does not occur for routing lookups out the tunnel interface between them.

Copyright © 2010 Internetwork Expert

3.12 Reliable Backup Interface with GRE

- Remove all previous routing configurations.
- Configure static routes on R5 for R4's Loopback0 interface via both the Frame Relay and point-to-point connections between them.
- Configure static routes on R4 for R5's Loopback0 interface via both the Frame Relay and point-to-point connections between them.
- The static routes on R4 and R5 via the Frame Relay circuit should have a higher administrative distance than those out the point-to-point link.
- Create a GRE tunnel between R4 and R5 using the addresses 10.0.0.Y/24 that is sourced from and destined to the Frame Relay connection between them, and enable GRE keepalives.
- Configure the backup interface feature on R5 such that if the tunnel between R4 and R5 goes down the point-to-point Serial link between them is activated.
- To verify this configuration ensure that traffic between the Loopback0 interfaces of R4 and R5 is routed out the Frame Relay link between them, but if R4's Frame Relay interface is shut down this traffic is rerouted out the point-to-point link.

3.13 On-Demand Routing (ODR)

- Remove all previous routing configurations.
- Disable all physical interfaces on R1, R2, R3, R4, and R5 with the exception of their connections to the Frame Relay cloud.
- Ensure that CDP is enabled on the Frame Relay network between these devices.
- Enable ODR on R5.
- Ensure that all devices on this segment have IP reachability to each others' Loopback0 networks.

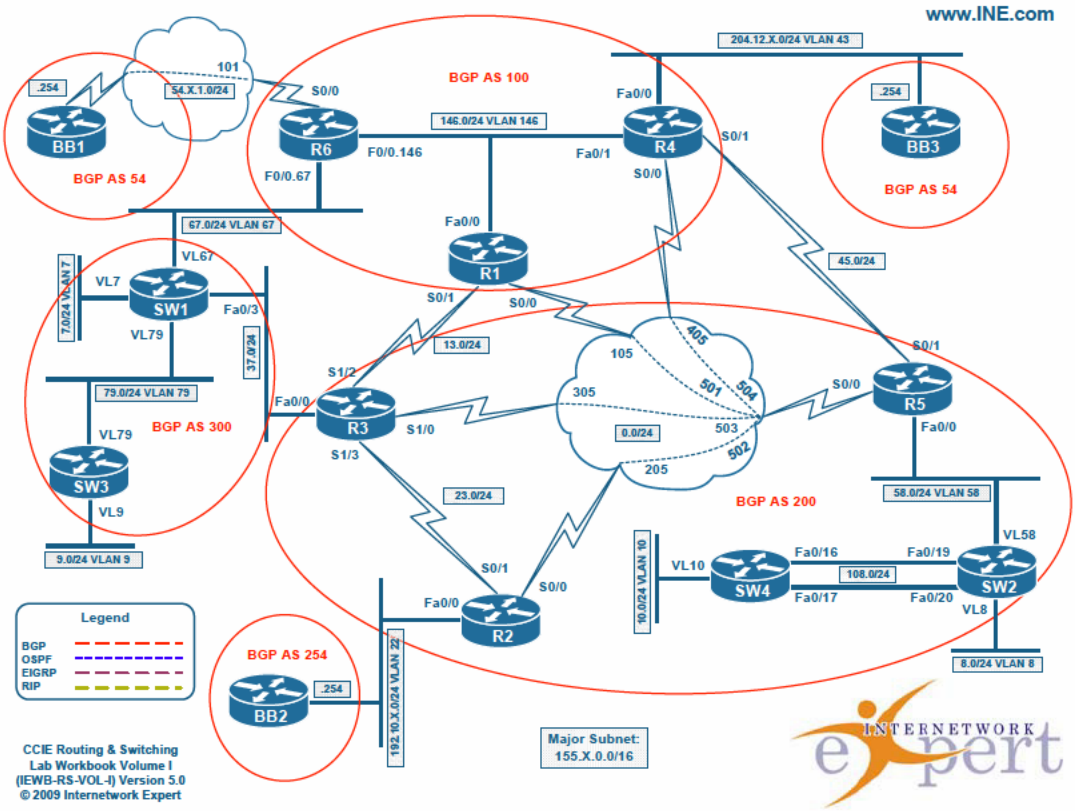
Copyright © 2010 Internetwork Expert

Note

Load the *OER* initial configurations prior to starting. The scenario presents AS 200 with 3 exits points to reach AS 100:

a) via a link between R4 and R5, with a link bandwidth of 128Kbps
b) via a link between R1 and R3, with a link bandwidth of 128Kbps
c) via a link between R3 and SW1, with a simulated bandwidth of 256Kbps (via traffic-shaping)

AS 200 will be used as the network implementing Cisco OER to optimally utilize all exits points to AS 100 (two direct connection and one across AS 300).



3.14 OER Components Setup

- Configure R5 as the OER MC and R2, R3, and R5 as the Border Routers.
- Authenticate all OER communications using a password value of "CISCO".
- Set up external and internal OER interfaces on all routers per the network topology presented.
- Enable OER event logging and change the keepalive interval to 5 seconds.

3.15 OER Profile Phase

- Configure the OER deployment to learn network prefixes based on maximum throughput and delay.
- The address information should be summarized based on the BGP table contents.
- Run the periodic learning procedure every 5 minutes for a duration of 3 minutes, and do not store more that 10 prefixes in the MTC.
- Only store the traffic flows corresponding to ICMP data exchange, WWW transactions, and VoIP bearer traffic (UDP port range 16384-32767).
- Manually configure a prefix traffic class of 112.0.0.0/24.

3.16 OER Measure Phase

- Use the combined monitoring mode and the configured manual TCP connect probes for the Loopback0 IP addresses of R1, R4, and R6.
- Configure the Master Controller to use active monitoring only for the prefix 150.X.4.0/24.
- Configure the Border Routers so that any interface utilization changes are detected as quickly as possible.

Copyright © 2010 Internetwork Expert

3.17 OER Apply Policy Phase

- Configure the network's performance policy to look for the best exit for the traffic classes.
- OER should generate an OOP event once any link utilization exceeds 80% or when the range between link utilization is above 10%.
- Any acceptable exit point should have a relative loss of no higher than 3% and delay no higher than 200ms.
- Best exit should be selected based on the minimum external utilization, and then based on the lowest delay for the prefix class.
- Reverse the above policy logic for the prefix 150.X.6.0/24 and change the acceptable delay to 100ms.
- All exit points with performance metrics within 15% of the best exit should be considered equivalent.
- Ensure periodic re-optimization occurs every minimal time interval.
- After an OOP event, the traffic class should retain the new exit for no less than 10 minutes.

3.18 OER Control & Verify Phase

- Enable OER to control routes in the network domain.
- Use the tag value of 1000 for injected static routes and a BGP Local Preference value of 6000.
- Ensure that any injected static route will be redistributed into the IGP used by AS 200.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

RIP

Note

Load the *RIP* initial configurations prior to starting. Note that R4's connection to VLAN 146 and the Serial link between R2 and R3 are disabled.

4.1 Basic RIP Configuration

- Configure RIPv2 on all interfaces of all devices in the internal network.
- Disable auto-summary.
- R4 and R6 should be learning RIP routes from BB3 and BB1 respectively.
- Test reachability to all networks and note any problems within the topology.

4.2 RIPv2 Authentication

- Configure RIPv2 authentication on the Ethernet link between R2 and BB2.
- Use the MD5 key number 1 with a password of CISCO.
- R2 should be learning RIP routes from BB2.
- Configure clear-text RIP authentication on the segment between R1 and R6 using the password CCIE.

4.3 RIPv2 Split Horizon

- Disable split-horizon on R5's connection to the Frame Relay cloud.
- Test reachability to all networks and note any changes within the topology.

4.4 RIPv2 Auto-Summary

- Enable auto-summary under the RIP process of R4.
- Note any changes in the network advertisements that R4 is sending.

Copyright © 2010 Internetwork Expert

4.5 RIP Send and Receive Versions

- Remove the `version 2` commands under the RIP processes of SW2 and SW4.
- Configure SW2 to send and receive only RIPv2 updates on VLAN 58.
- Configure SW2 to send and receive only RIPv1 updates on the link to SW4.
- Note any changes in reachability or routing information in the network.

4.6 RIPv2 Manual Summarization

- Configure R4 to send two summary routes for the RIP networks learned from BB3 to R5.
- Ensure that these summaries do not overlap any address space that R4 does not have a longer match route to.

4.7 RIPv2 Convergence Timers

- Change the RIP timers throughout the topology to make convergence three times faster than the default.
- Ensure that this configuration does not affect the links to the BB routers.

4.8 RIPv2 Offset List

- Configure an offset-list on R6 so that all traffic going to BB1 uses the Ethernet link to SW1.
- If this link is down traffic should be rerouted over the link to R1.

4.9 RIPv2 Filtering with Passive Interface

- Configure the passive interface feature on SW2 so that it learns RIP updates from SW4 but does not advertise any information back to SW4.

4.10 RIPv2 Filtering with Prefix-Lists

- Configure a prefix-list on R5 so that it does not advertise the two RIP summaries of the BB3 networks to SW2, and so that all other networks are allowed to be advertised.
- Configure a prefix-list on R5 so that it does not install any updates received from R4 on the Frame Relay network. This configuration should not affect the updates received from other neighbors on this segment.
- Ensure that route feedback does not occur for the summaries that R4 is generating to R5.

Copyright © 2010 Internetwork Expert

4.11 RIPv2 Filtering with Standard Access-Lists

- Configure a one line standard access-list on R6 to filter out all routes coming from BB1 that have an even number in the third octet.

4.12 RIPv2 Filtering with Extended Access-Lists

- Remove the previous prefix-list filters on R5.
- Configure an extended access-list filter on R5 so that the routes to VLANs 7 and 9 are only received from R1, while the routes to R1's Loopback and VLAN 146 are only received from R3.
- This filter should not affect any other updates on this segment.

4.13 RIPv2 Filtering with Offset Lists

- Configure an offset-list on SW1 so that SW3 does not install a route to VLAN 5.
- This filter should not affect any other updates on this segment.

4.14 RIPv2 Filtering with Administrative Distance

- Configure administrative distance filtering on R5 so that devices within the network cannot reach R4's Loopback0 network.
- This filter should not affect any other networks in the topology.

4.15 RIPv2 Filtering with Per Neighbor AD

- Configure administrative distance filtering on SW1 so that traffic destined for R3's Loopback 0 network is sent towards R6.
- This configuration should not affect any other networks in the topology.

4.16 RIPv2 Default Routing

- Configure R6 to advertise a default route only to R1 via RIP.
- R6 should not send this default route directly to SW1.
- Do not use any access-list or prefix-lists on R6 to accomplish this.

4.17 RIPv2 Conditional Default Routing

- Remove the previous default route advertisement on R6.
- Configure R4 to originate a default route into the RIP domain.
- If the link to BB3 goes down R4 should withdraw its default advertisement.

Copyright © 2010 Internetwork Expert

4.18 RIPv2 Reliable Conditional Default Routing

- Configure the IP SLA feature on R4 to track ICMP reachability to BB3.
- Modify the previous default route origination on R4 so that if an ICMP echo-reply is not received from BB3 R4 withdraws its default advertisement.

4.19 RIPv2 Unicast Updates

- Configure R5 and SW2 so that RIPv2 updates sent over VLAN 58 use unicasts instead of multicasts.

4.20 RIPv2 Broadcast Updates

- Configure R1 and R6 so that RIPv2 updates sent over VLAN 146 use broadcasts instead of multicasts.

4.21 RIPv2 Triggered Updates

- Configure R4 and R5 so that RIPv2 updates are only exchanged over the low speed point-to-point Serial link between them when there is a change in the RIP topology.

4.22 RIPv2 Source Validation

- Configure R1 and R3 to use PPP on the Serial link between them.
- Remove R1's IP address on this segment.
- Configure R3 to assign R1 the IP address 155.1.13.1/32 via IPCP.
- Ensure that RIPv2 updates sent across this link can be installed in the routing tables of these two devices.

Copyright © 2010 Internetwork Expert

EIGRP

> **Note**
>
> Load the *Initial EIGRP* initial configurations prior to starting. Note that R4's link to VLAN 146 and the point-to-point link between R2 and R3 are disabled.

5.1 EIGRP Network Statement

- Configure EIGRP AS 100 on all devices in the internal topology.
- Enable EIGRP on all interfaces in the 150.X.0.0 and 155.X.0.0 networks on all devices.
- Any new interfaces added should not automatically have EIGRP enabled on them regardless of their IP addresses.
- Do not disable auto-summary.
- Note any reachability problems throughout the network.

5.2 EIGRP Auto-Summary

- Disable auto-summary on all devices running EIGRP.
- Note any changes in reachability throughout the network.

5.3 EIGRP Split Horizon

- Disable split-horizon for EIGRP on R5's connection to the Frame Relay network.
- Note any changes in reachability throughout the network.

5.4 EIGRP MD5 Authentication

- Configure EIGRP 10 on the link between R6 and BB1.
- Authenticate this adjacency with the MD5 key 1 using the password CISCO.
- Use a key-chain named MD5_KEYS.

Copyright © 2010 Internetwork Expert

5.5 EIGRP Key Chain Rotation

- Authenticate the EIGRP adjacencies on the Frame Relay network between R1, R2, R3, R4, and R5 using key 10 and the password CISCO10.
- Key 10 should be sent until 12:05 AM on Jan 1st 2030, and should be accepted for 10 minutes past this time.
- Configure a new key 20 with the password CISCO20 that is sent starting 12:00 AM on Jan 1st 2030, and is accepted any time after this time.
- Use a key-chain named KEY_ROTATION.

5.6 EIGRP Unicast Updates

- Configure R5 and SW2 so that they exchange EIGRP packets only as unicasts on their connection to VLAN 58.

5.7 EIGRP Default Network

- Redistribute between EIGRP AS 10 and EIGRP AS 100 on R6.
- Configure R6 to advertise the network 200.0.0.0/24 as the default network to all devices in EIGRP AS 100.

5.8 EIGRP Summarization

- Redistribute between RIP and EIGRP AS 100 on R4.
- Use the metric of R4's connection to VLAN 43 to translate RIP metrics into EIGRP metrics.
- Configure R4 to summarize the 30.0.0.0 subnets to R5 out the Frame Relay link, and the 31.0.0.0 subnets out the point-to-point link.
- Do not overlap any address space that R4 does not have a longer match to.
- If R4's point-to-point link is down traffic for the 30.0.0.0 subnets should be rerouted out the Frame Relay link.
- If R4's Frame Relay link is down traffic for the 31.0.0.0 subnets should be rerouted out the point-to-point link.

Copyright © 2010 Internetwork Expert

5.9 EIGRP Summarization with Default Routing

- Remove R6's default network advertisement.
- Remove R4's previous summarization.
- Configure summarization on R4's connections to R5 so that it only advertises a default route out to R5 via EIGRP.

5.10 EIGRP Summarization with Leak Map

- Configure a leak-map on R4 so that traffic going to R4's Loopback 0 network is routed out the point-to-point link between R4 and R5.
- If this link is down traffic should still be rerouted out the Frame Relay connection between these devices.

5.11 EIGRP Floating Summarization

- Shutdown the point-to-point link between R4 and R5.
- Configure R5 to summarize the Loopback 0 networks of R4 and R5 out to SW2; this route should not overlap any additional networks.
- Configure an equal longest match static route on R5 so that SW2 has reachability to both Loopback 0 networks of R4 and R5.

5.12 EIGRP Poisoned Floating Summarization

- Remove the previously configured static route on R5.
- Modify the administrative distance of the summary that R5 is generating to SW2 so that a route to Null0 is not installed.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Erase and reload all devices and load the *Basic EIGRP Routing* initial configurations before continuing.

5.13 EIGRP Metric Weights

- Configure all devices in EIGRP AS 100 so that only delay is used in the composite metric calculation.

5.14 EIGRP Traffic Engineering with Metric

- Configure a metric manipulation on SW1 so that traffic from SW3 to the Loopback 0 network of R6 transits the link between R3 and R1.

5.15 EIGRP Unequal Cost Load Balancing

- Configure unequal cost load balancing so that traffic from R6 going to VLAN 9 is load balanced between R1 and SW1.
- The traffic share should be configured in such a way that the link to SW1 is used five times as much as the link to R1.
- Verify this by configuring per-packet load balancing on R6.

5.16 EIGRP Convergence Timers

- Configure R1 through R6 so that EIGRP hellos are sent every one second; these devices should inform their neighbors to declare them down if subsequent hellos are not received within three seconds.
- Configure SW1 through SW4 so that EIGRP hellos are sent every ten seconds; these devices should inform their neighbors to declare them down if subsequent hellos are not received within thirty seconds.
- Additionally configure AS 100 so that lost routes are considered Stuck In Active if a query response has not been heard within one minute.

5.17 EIGRP Stub Routing

- Configure the EIGRP stub feature in such a way that SW2 does not receive EIGRP query messages.
- Ensure that all devices in AS 100 still have IP reachability to VLAN 8.

Copyright © 2010 Internetwork Expert

5.18 EIGRP Stub Routing with Leak Map

- Configure the EIGRP stub feature in such a way that R5 does not receive EIGRP query messages.
- R5 should continue to advertise all learned routes with the exception of SW2's Loopback 0 network.

5.19 EIGRP Filtering with Passive Interface

- Configure the passive-interface feature on R5, SW2, and SW4 so that EIGRP hello packets are not sent out the LAN segments without routers attached.
- Configure the passive-interface default feature on SW1 and SW3 so that EIGRP hello packets are not sent out the LAN segments without routers attached; ensure that full reachability is maintained after this change is made.

5.20 EIGRP Filtering with Prefix-Lists

- Configure a prefix-list on R4 so that it does not advertise the 30.0.0.0 and 31.0.0.0 subnets learned from BB3 out the point-to-point link to R5; use the most efficient list to accomplish this that will not deny any other networks than those subnets R4 is learning.
- Configure a prefix-list on R1 so that it does not install any updates received from R4 on the VLAN 146 segment.

5.21 EIGRP Filtering with Standard Access-Lists

- Configure a one line standard access-list on R6 to filter out all routes coming from BB1 that have an odd number in the third octet.

5.22 EIGRP Filtering with Extended Access-Lists

- Shutdown R5's point-to-point link to R4.
- Configure an extended access-list filter on R5 so that traffic for the Loopback 0 networks of R4 and R6 is sent to R2.
- Traffic for the Loopback 0 networks of R1 and R2 should be sent to R3.
- Traffic for the Loopback 0 networks of SW1 and SW3 should be sent to R1.
- This filter should not affect any other updates on this segment.

Copyright © 2010 Internetwork Expert

5.23 EIGRP Filtering with Offset Lists

- Configure an offset-list on SW1 so traffic destined for R3's Loopback0 network is sent to R6.
- If the link to R6 is down traffic should be rerouted directly to R3.

5.24 EIGRP Filtering with Administrative Distance

- Configure administrative distance filtering on R6 so that it does not install the route to R4's Loopback 0 network.

5.25 EIGRP Filtering with Per Neighbor AD

- Configure administrative distance filtering on R3 so that traffic destined for SW1's Loopback 0 network is sent towards R1.

5.26 EIGRP Filtering with Route Maps

- Configure R4 to redistribute the VLAN 43 subnet into EIGRP with the tag value of 4.
- Configure a route-map filter on R2 that matches this tag value and denies the route from being installed in the routing table.
- Configure a route-map filter on R3 that denies EIGRP routes with a metric in the range of 500,000 – 750,000 from entering the routing table.
- These filters should not impact any other networks advertised by R4 or learned by R2 and R3.

5.27 EIGRP Bandwidth Pacing

- Configure R2 and R3 so that EIGRP can not use more than 154Kbps of bandwidth on the point-to-point link between them, assuming that the link speed is 1544Kbps.

5.28 EIGRP Default Metric

- Configure a static route on R2 for the prefix 222.22.2.2/32 that is reachable via BB2.
- Advertise this prefix into EIGRP as external routes using a default metric of 100Mbps, 100 microseconds of delay, maximum reliability, minimum load, and an MTU of 1500 bytes.

Copyright © 2010 Internetwork Expert

5.29 EIGRP Neighbor Logging

- Configure SW3 so that it does not log EIGRP neighbor adjacency events.
- Additionally EIGRP warning logs should not be generated more often than every 20 seconds.

5.30 EIGRP Router-ID

- Modify the EIGRP Router-ID on SW2 so that external EIGRP routes generated by R2 are ignored.

5.31 EIGRP Maximum Hops

- Configure all devices in EIGRP AS 100 so that routes with a hop count of greater than 10 are considered invalid.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

OSPF

> **Note**
>
> Load the *Initial OSPF* initial configurations prior to starting.

6.1 OSPF over Broadcast Media

- Enable OSPF on all devices using Process-ID 1.
- On R1 configure OSPF area 1 on the link to VLAN 146.
- On R2 configure OSPF area 51 on the link to VLAN 22.
- On R3 configure OSPF area 2 on the link to SW1.
- On R4 configure OSPF area 1 on the link to VLAN 146.
- On R5 configure OSPF area 3 on the links to VLANs 5 and 58.
- On R6 configure OSPF area 1 on the link to VLAN 146, and area 2 on the link to VLAN 67.
- Do not use the **`network`** statement under the OSPF process on R1, R2, R3, R4, R5, or R6.
- On SW1 configure OSPF area 2 on all interfaces with an IP address assigned using one single **`network`** statement.
- On SW2 configure OSPF area 3 on interfaces with IP addresses in the range of 155.X.0.0 – 155.X.127.255 using one single **`network`** statement.
- On SW3 configure OSPF area 2 on interfaces with IP addresses in the range of 155.X.9.0 – 155.X.9.255 and 155.X.79.0 – 155.X.79.255 using two **`network`** statements.
- On SW4 configure OSPF area 3 only on interfaces with the exact IP addresses 155.X.10.10 and 155.X.108.10.
- Note any reachability problems throughout the OSPF topology.

Copyright © 2010 Internetwork Expert

6.2 OSPF over Non-Broadcast Media

- Configure OSPF area 0 on R1, R2, R3, R4, and R5's connections to the Frame Relay network.
- Without modifying the OSPF network type on any of these devices ensure full reachability is obtained throughout the OSPF domain.

6.3 OSPF DR/BDR Election Manipulation

- Configure the network so that R6 is elected the OSPF Designated Router for VLANs 67 and 146.
- If R6 goes down R1 should take over the DR status for VLAN 146.
- When R6 comes back up it should become the BDR.
- Modify the DR/BDR election on the Frame Relay network to ensure that if R5's connection to the Frame Relay network goes down and comes back up, full reachability is still maintained.

6.4 OSPF Network Point-to-Point

- Configure OSPF area 0 on the point-to-point Serial link between R4 & R5.
- Configure OSPF area 4 on the point-to-point Serial link between R1 & R3.
- Configure OSPF area 5 on the point-to-point Serial link between R2 & R3.

6.5 OSPF Network Point-to-Multipoint

- Remove all **`frame-relay map`** statements on R1, R2, R3, and R4 with the exception of the mapping to R5, and note any changes in reachability throughout the OSPF domain.
- Modify the Frame Relay network's OSPF configuration to use network type point-to-multipoint, and note any changes in the OSPF database, the routing table, and in reachability throughout the network.

6.6 OSPF Network Point-to-Multipoint Non-Broadcast

- Remove the **`broadcast`** keyword from the **`frame-relay map`** statements of R1, R2, R3, R4, and R5, and note any changes in reachability throughout the OSPF domain.
- Modify the Frame Relay network's OSPF configuration to use network type point-to-multipoint non-broadcast, and note any changes in the OSPF database, the routing table, and in reachability throughout the network.

Copyright © 2010 Internetwork Expert

6.7 OSPF Network Loopback

- Advertise the Loopback0 interfaces of R1, R2, R3, and R4 into OSPF area 0.
- Advertise the Loopback0 interface of R6 into OSPF area 1.
- Advertise the Loopback0 interfaces of SW1, and SW3 into OSPF area 2.
- Advertise the Loopback0 interfaces of R5, SW2, and SW4 into OSPF area 3.
- Modify the network type of these interfaces so that the links are not advertised as host routes.

6.8 OSPF Path Selection with Auto-Cost

- Modify the global OSPF cost calculation of all devices so that a TenGigabitEthernet interface has a cost of 3, and an OC-3 link has a cost of 193.
-

6.9 OSPF Path Selection with Cost

- Using the interface level **`ip ospf cost`** command modify the OSPF domain so that traffic from SW4 going to the Loopback0 network of R6 uses the point-to-point Serial link between R4 and R5.

6.10 OSPF Path Selection with Bandwidth

- Using the interface level **`bandwidth`** keyword modify the OSPF domain so that traffic from R6 to the Loopback0 network of SW2 is first sent to R1.

6.11 OSPF Path Selection with Per-Neighbor Cost

- The Frame Relay PVC between R1 & R5 has been provisioned with 512Kbps of bandwidth by the service provider, while the PVC between R4 & R5 has been provisioned at 768Kbps. Configure per-neighbor cost values on R5 to reflect the underlying bandwidth of the Frame Relay circuits connecting it to R1 and R4.

6.12 Repairing Discontiguous OSPF Areas with Virtual-Links

- Configure the network so that full reachability is maintained if R3's connection to SW1 goes down.
- Do not apply any configuration onto R4 to solve this task.

Copyright © 2010 Internetwork Expert

6.13 OSPF Path Selection with Non-Backbone Transit Areas

- Configure R4's Frame Relay and point-to-point Serial links connecting to R5 to have an OSPF cost of 1.
- Modify the SPF calculation in the OSPF domain so that R6 cannot use R4 to reach area 0 by transiting area 1.
- Verify this by ensuring that traffic from R6 going to the Loopback0 network of SW2 is still first sent to R1, and then directly to R5.

6.14 OSPF Path Selection with Virtual-Links

- Configure the OSPF domain so that traffic from SW3 going to OSPF area 51 transits the point-to-point Serial link between R3 and R2.

6.15 OSPF Demand Circuit

- The point-to-point between R4 and R5 is a low-speed Serial link. Configure the OSPF demand circuit feature on this link to reduce periodic OSPF hello transmission and paranoid update flooding.

6.16 OSPF Flooding Reduction

- Configure R5, SW2, and SW4 so that links in OSPF area 3 do not participate in periodic paranoid update LSA flooding.

6.17 OSPF Clear Text Authentication

- Configure clear-text OSPF authentication for all adjacencies in area 2 using the password CLEARKEY.
- SW1 should enable authentication on all interfaces in area 2 with one single command.
- R3, R6, and SW3 should only enable authentication on their links connecting to SW1.

6.18 OSPF MD5 Authentication

- Configure MD5 based OSPF authentication for all adjacencies in area 0, including the Virtual-Links, using the password MD5KEY.
- R1 should enable MD5 authentication on all interfaces in area 0 with a single command.
- All other devices in area 0 should enable MD5 authentication on a per interface basis.

Copyright © 2010 Internetwork Expert

6.19 OSPF Null Authentication

- Modify SW1 so that if a new router is added to area 2 on VLAN 7 it does not require authentication.

6.20 OSPF MD5 Authentication with Multiple Keys

- Enable MD5 authentication in OSPF area 1.
- R1 should authenticate R6 using the password R1R6KEY.
- R4 should authenticate R6 using the password R4R6KEY.

6.21 OSPF Internal Summarization

- Configure R5 to advertise a summary route for the VLAN 8 and 10 prefixes as they are sent into area 0.
- This summary should be as specific as possible while still encompassing all addresses in both subnets.

6.22 OSPF Path Selection with Summarization

- Configure R3 to originate the summary route 155.X.6.0/23 in order to force traffic destined to VLAN 7 to transit R6.
- If R6's connection to VLAN 146 is down, traffic for VLAN 7 should transit R3.

6.23 OSPF External Summarization

- Redistribute between RIPv2 and OSPF on R4.
- Redistribute between EIGRP and OSPF on R6.
- R4 should advertise a single summary route into OSPF for the prefixes learned from BB3 that has a cost of 50.
- R6 should advertise a single summary route into OSPF for the prefixes learned from BB1 that has a cost of 100, and includes the cost needed to reach R6.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Revert all devices to the *Full OSPF Routing* initial configurations before continuing.

6.24 OSPF Stub Areas

- Configure OSPF area 3 so that R5 filters External routes out as they are sent from area 0 to area 3.
- Devices in area 3 should still have reachability to routes external to the OSPF domain.

6.25 OSPF Totally Stubby Areas

- Configure OSPF area 3 so that R5 filters Inter-Area and External routes as they are sent from area 0 to area 3.
- Devices in area 3 should still have reachability to routes external to the OSPF domain.

6.26 OSPF Not-So-Stubby Areas

- Shutdown R6's Loopback0 interface and connection to VLAN 146.
- Configure OSPF area 2 so that R3 filters External routes out as they are sent from area 0 to area 2.
- R6 should still be allowed to redistribute EIGRP into OSPF.
- Note any reachability problems throughout the domain.

6.27 OSPF Not-So-Stubby Areas and Default Routing

- Configure R3 to advertise a default route into area 2.
- This default route should have a cost of 500.

6.28 OSPF Not-So-Totally-Stubby Areas

- Remove the previous default routing configuration on R3.
- Modify the area 2 configuration so that R3 filters Inter-Area and External routes as they are sent from area 0 to area 2.
- R6 should still be allowed to redistribute EIGRP into OSPF.

Copyright © 2010 Internetwork Expert

6.29 OSPF Stub Areas with Multiple Exit Points

- Re-enable R6's Loopback0 interface and connection to VLAN 146.
- Modify area 2 in such a way that traffic from devices in area 2 going to area 3 uses R6 as the exit point, while traffic from area 2 going to the RIP routes learned from BB3 uses R3.
- If R3 loses its connection to area 2, traffic for External destinations should be rerouted to R6.
- If R6 loses its connection to area 2, traffic for Inter-Area destinations should be rerouted to R3.
- Do not modify the cost of any links in area 2 to accomplish this.

6.30 OSPF NSSA Type-7 to Type-5 Translator Election

- Create a new Loopback9 interface on SW3 with the IP address 9.9.9.9/32, and redistribute it into OSPF.
- Modify the OSPF domain so that only R3 advertises this route into area 0, but traffic from R5 going to this destination transits R6.
- If R6 is down this traffic should be rerouted to R3.

6.31 OSPF NSSA Redistribution Filtering

- Create new Loopback interfaces on R5 and SW4 with IP addresses 5.5.5.5/32 and 10.10.10.10/32 respectively, and redistribute them into the OSPF domain.
- Configure area 3 as an NSSA so R5 blocks all LSA types 3, 4, and 5 and replaces them with a default route.
- Modify area 3 so that R5's redistributed Loopback is injected into area 0 as LSA Type-5, but is not injected into area 3 as LSA Type-7.
-

6.32 OSPF LSA Type-3 Filtering

- Configure LSA Type-3 Filtering on R5 so that devices in area 0 do not have reachability information about the layer 3 EtherChannel link between SW2 and SW4 or SW4's Loopback0 interface.
- Configure LSA Type-3 Filtering on R2 so that devices in area 51 do not have reachability information about the Loopback0 interface of R5; this should not affect devices in area 5.

Copyright © 2010 Internetwork Expert

6.33 OSPF Forwarding Address Suppression

- Modify R5's NSSA configuration so that devices outside of area 3 maintain connectivity to SW4's redistributed Loopback 10.10.10.10/32.

> **Note**
>
> Revert all devices to the *Full OSPF Routing* initial configurations before continuing.

6.34 OSPF Default Routing

- Configure R6 with a static default route pointing to BB1.
- Configure R6 to originate an external type-2 default route with a metric of 60 into OSPF as long as the static default route is installed in the routing table.
- Do not use a route-map to accomplish this.
- Configure R4 to originate an external type-1 default route with a metric of 40 regardless of whether it has a default route already installed in the routing table.

6.35 OSPF Conditional Default Routing

- Remove the previous default route advertisement on R6.
- Configure R6 to originate a default route into OSPF, but only if the local Serial interface connecting to BB1 is up.

6.36 OSPF Reliable Conditional Default Routing

- Remove the previous default route advertisement on R4.
- Configure an IP SLA instance on R4 to check ICMP reachability to BB3 every five seconds.
- R4 should advertise a default route into OSPF, but only if the SLA monitoring reports its status as OK.

Copyright © 2010 Internetwork Expert

6.37 OSPF Filtering with Distribute-Lists

- Configure distribute-list filtering on R5, SW2, and SW4 so that these devices do not install routes to the Loopback0 networks of R1 and R2.

6.38 OSPF Summarization and Discard Routes

- Configure R5 to advertise the summary 150.X.0.0/22 into area 3.
- Ensure that R5, SW2, and SW4 can still reach the Loopback0 networks of R1 and R2, even with the previously configured distribute-list filter applied.

6.39 OSPF Filtering with Administrative Distance

- Configure administrative distance filtering on R5 so that traffic going towards the VLAN 67 network is sent towards R3.

6.40 OSPF Filtering with Route-Maps

- Configure route-map filtering on R6 so that traffic going towards the Loopback0 network of R3 is sent towards R1.

6.41 OSPF NSSA ABR External Prefix Filtering

- Disable R6's link to VLAN 146 and Loopback0.
- Configure area 2 as an NSSA.
- Configure summarization on R3 so that devices outside of area 2 do not have a route to the network 200.0.0.0/24.
- This filter should not affect any other prefixes.

6.42 OSPF Database Filtering

- Configure SW1 so that SW3 cannot learn any OSPF routes from SW1, but SW1 can still learn OSPF routes from SW3.
- Configure R5 so that R2 cannot learn from R5, but R5 can still learn OSPF routes from R2.

6.43 OSPF Stub Router Advertisement

- Configure R4 to advertise the maximum metric value to all neighbors inside its Type-1 Router LSA.

Copyright © 2010 Internetwork Expert

6.44 OSPF Interface Timers

- Configure R5 to send OSPF hello packets out its link to the Frame Relay network every five seconds, and wait for seven seconds before declaring a neighbor down.
- Configure R4 to send OSPF hello packets every 250ms on the point-to-point link to R5.

6.45 OSPF Global Timers

- Modify R4 and R5's OSPF timers as follows:
    - Configure SPF throttling to start new re-calculation at least 100ms after a new LSA arrives. The second SPF calculation should occur in no less than 1s after the first one, and the maximum wait time should be no more than 10 seconds.
    - Configure LSA pacing to wait at least 50ms between consecutive link-state updates. LSA retransmissions should be paced at least 75ms apart.
    - Configure LSA throttling to generate subsequent LSAs after 10ms, to wait at least 4 seconds to generate the next LSA, and no more than 6 seconds between generation of the same LSA.
    - Configure LSA arrival throttling to wait 2 seconds between reception of the same LSA from a neighbor.
    - of the re-calculation at least 100ms after a new LSA arrives. The second SPF calculation should occur in no less than 1s after the first one, and the maximum wait time should be no more than 10 seconds.
- Configure R4 and R5 to assume that LSA transmission takes 2 seconds on the point-to-point link between them. LSA retransmission should occur if an acknowledgement is not received within 10 seconds over this link.

6.46 OSPF Resource Limiting

- Configure R4 so that no more than 5000 LSAs can exist in the database.
- No more than 500 of these routes should be originated through redistribution.
- R4's OSPF process should be allowed to use up to 20% CPU utilization before allowing interrupts from higher priority processes.

Copyright © 2010 Internetwork Expert

6.47 Miscellaneous OSPF Features

- Configure SW4 so that it goes not generate a log message upon receipt of a Type-6 LSA advertisement.
- Configure SW4 so that it does not take the MTU value into account when establishing adjacencies on its Port-Channel1 interface.
- Configure SW4 to reflect the following output:

```
Rack1SW4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address
Interface
Rack1SW2          1   FULL/BDR        00:00:38    155.1.108.8
Port-channel1
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

BGP

> **Note**
>
> Load the *Initial BGP* initial configurations prior to starting.

7.1 Establishing iBGP Peerings

- Configure BGP on all internal devices using AS 100.
- Create a full mesh of iBGP peerings between these devices without using their Loopback interfaces.
- Advertise the Loopback 0 interfaces of these devices into BGP.
- Ensure full reachability to these Loopback 0 interfaces from all internal devices.

7.2 Establishing EBGP Peerings

- BB1 and BB3 are in AS 54.
- Configure EBGP peerings between R4 & BB3 and R6 & BB1 using their directly connected links.
- Advertise the external links between R4 & BB3 and R6 & BB1 into IGP.
- Ensure full reachability to all prefixes learned from AS 54 from all internal devices when sourcing traffic from the Loopback 0 interfaces.

7.3 BGP Update Source Modification

- Advertise the Loopback 0 interfaces of R4 and R5 into IGP.
- Modify the BGP peering between these devices so that if either the Frame Relay or Point-to-Point Serial link between them goes down, the BGP peering is not affected.

7.4 Multihop EBGP Peerings

- Create a new Loopback 1 interface on SW4 with the IP address 204.12.X.10/32, and advertise it into IGP.
- Configure an EBGP peering between SW4 and BB3 using this new interface as the source of the peering.

Copyright © 2010 Internetwork Expert

7.5 Neighbor Disable-Connected-Check

- Remove the all previous BGP configurations.
- Configure R1 & R4 in AS 100, and R5 in AS 200.
- Configure an iBGP peering between R1 and R4.
- Configure an EBGP peering between R4 and BB3, which is in AS 54.
- Configure an EBGP peering between R1 and R5.
- Configure an EBGP peering between R4 and R5 in such a way that the peering remains up as long as R4 has a connection to either the Frame Relay network or the Point-to-Point link to R5, but is torn down if both of these links are down.

7.6 Authenticating BGP Peerings

- Remove the previous BGP configuration on R2.
- Configure R2 in BGP AS 200, and configure an EBGP peering with BB2.
- BB2 is in AS 254.
- Authenticate this peering with the password "CISCO".

7.7 iBGP Route Reflection

- Remove the previous BGP configuration on all devices.
- Remove the advertisement of R4 and R5's Loopback 0 networks into IGP.
- Configure BGP on all internal devices using AS 100.
- Configure iBGP peerings from R1 to all other devices in the internal network.
- Configure EBGP peerings between R4 & BB3 and R6 & BB1 using their directly connected links; BB1 and BB3 are in AS 54.
- Advertise the Loopback0 interfaces of these devices into BGP.
- Ensure full reachability to these Loopback 0 interfaces from all internal devices, and to all prefixes learned from AS 54 from all internal devices when sourcing traffic from the Loopback 0 interfaces.

Copyright © 2010 Internetwork Expert

7.8 Large Scale iBGP Route Reflection with Clusters

- Remove the previous BGP configuration on all devices.
- Configure R2 in BGP AS 200, and configure an EBGP peering with BB2.
- BB2 is in AS 254, and is authenticating this peering with the password "CISCO".
- Configure BGP on all other internal devices using AS 100.
- Configure a BGP cluster between R1, R4, and R6 as follows:
  - R1 should be the route-reflector, and peer with R4 and R6.
  - R4 and R6 should peer with BB3 and BB1 respectively, who are in AS 54.
  - Use the cluster-id 150.X.1.1.
- Configure a BGP cluster between R3, SW1, and SW3 as follows:
  - R3 should be the route-reflector, and peer with SW1 and SW3.
  - Use the cluster-id 150.X.3.3.
- Configure a BGP cluster between R5, SW2, and SW4 as follows:
  - R5 should be the route-reflector, and peer with SW2 and SW4.
  - Use the cluster-id 150.X.5.5.
- R1, R3, and R5 should all peer with each other in a full-mesh, but should not propagate updates between clusters.
- Configure EBGP peerings between R2 & R3 and R2 & R5.
- Advertise the Loopback0 interfaces of all devices into BGP.
- Ensure full reachability to these Loopback0 interfaces from all internal devices, and to all prefixes learned from AS 54 and 254 from all internal devices when sourcing traffic from the Loopback0 interfaces.

Copyright © 2010 Internetwork Expert

7.9 iBGP Confederation

- Remove the BGP configuration of all devices in AS 100.
- Configure a BGP Confederation Sub-AS between R1, R4, and R6 as follows:
  - Use the Sub-AS number 65146.
  - Use the Public AS number 100.
  - Configure full-mesh peerings between R1, R4, and R6.
  - R4 and R6 should peer with BB3 and BB1 respectively.
- Configure a BGP Confederation Sub-AS between R3, SW1, and SW3 as follows:
  - Use the Sub-AS number 65379.
  - Use the Public AS number 100.
  - Configure full-mesh peerings between R3, SW1, and SW3.
- Configure a BGP Confederation Sub-AS between R5, SW2, and SW4 as follows:
  - Use the Sub-AS number 65508.
  - Use the Public AS number 100.
  - R5 should be a route-reflector, and peer with SW2 and SW4.
- R1, R3, and R5 should all peer with each other in a full-mesh.
- Configure EBGP peerings between R2 & R3 and R2 & R5.
- Advertise the Loopback 0 interfaces of all devices into BGP.
- Ensure full reachability to these Loopback 0 interfaces from all internal devices, and to all prefixes learned from AS 54 and 254 from all internal devices when sourcing traffic from the Loopback 0 interfaces.

Copyright © 2010 Internetwork Expert

7.10 BGP Next-Hop Processing - Next-Hop-Self

- Remove the previous BGP configuration on all devices.
- Configure R2 in BGP AS 200, and configure an EBGP peering with BB2.
- BB2 is in AS 254, and is authenticating this peering with the password "CISCO".
- Configure BGP on all other internal devices using AS 100.
- Configure iBGP peerings from R1 to all other devices in AS 100.
- Configure EBGP peerings between R4 & BB3 and R6 & BB1 using their directly connected links; BB1 and BB3 are in AS 54.
- Configure EBGP peerings between R2 & R3 and R2 & R5.
- Advertise the Loopback 0 interfaces of all devices into BGP.
- Remove the advertisement of the links between R4 & BB3 and R6 & BB1 into IGP.
- Use the **`next-hop-self`** command where necessary to ensure full connectivity to the prefixes coming from AS 54.
- Ensure full reachability to the Loopback 0 interfaces from all internal devices, and to all prefixes learned from AS 54 and AS 254 from all internal devices when sourcing traffic from the Loopback 0 interfaces.

7.11 BGP Next-Hop Processing - Manual Modification

- Remove the previously configured next-hop-self statements.
- Configure an outbound route-map on R4, and an inbound route-map on R1 in order to resolve any next-hop reachability issues for the routes learned from AS 54.

7.12 iBGP Synchronization

- Enable iBGP Synchronization on all devices in AS 100.
- Ensure full reachability to the Loopback 0 interfaces from all internal devices, and to all prefixes learned from AS 54 and AS 254 from all internal devices when sourcing traffic from the Loopback 0 interfaces.
- Temporarily shut down the peering between R2 and R5.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Revert all devices to the *Initial BGP* initial configurations prior to continuing.

7.13 BGP over GRE

- Configure R4 in AS 100, with an EBGP peering to BB3 in AS 54.
- Configure R2 in AS 200, with an EBGP peering to BB2 in AS 254 using the password CISCO.
- Configure an EBGP peering between R2 and R4.
- Advertise the Loopback0 networks of R2 and R4 into BGP.
- Ensure that R2 can reach prefixes learned from AS 54, and R4 can reach prefixes learned from AS 254 when sourcing traffic from their Loopback0 networks.
- Do not redistribute between BGP and IGP to accomplish this.

7.14 BGP Redistribute Internal

- Remove all BGP configuration on all devices.
- Configure R1, R3, R4, and R6 in AS 100.
- R4 and R6 should peer with BB3 and BB1 respectively, who are in AS 54.
- R1 should peer with R3, R4, and R6 as a route reflector.
- Configure R4 and R6 to advertise the network 155.X.0.0/16 to AS 54.
- Configure BGP to IGP redistribution on R3 so that all internal devices have reachability to the prefixes learned from AS 54.

Copyright © 2010 Internetwork Expert

7.15 BGP Peer Groups

- Remove all BGP the configuration on the devices in AS 100.
- Configure BGP on all internal devices, with the exception of R2 and all switches, using AS 100.
- Configure iBGP peerings from R1 to all other devices in AS 100 using the peer group named IBGP_PEERS, be sure to use the Loopback 0 of these devices to form the peerings.
- Configure EBGP peerings between R4 & BB3 and R6 & BB1 using their directly connected links; BB1 and BB3 are in AS 54.
- Advertise the Loopback 0 interfaces of all devices into both IGP and BGP.
- Ensure full reachability to the Loopback0 interfaces from all internal devices, and to all prefixes learned from AS 54 and AS 254 from all internal devices when sourcing traffic from the Loopback0 interfaces.

7.16 BGP Network Statement

- Remove the **`next-hop-self`** statement used in the previous task to peer with BB1 and BB3
- Ensure full reachability from your internal network to all routes learned from AS 54 but do not advertise any prefixes into IGP

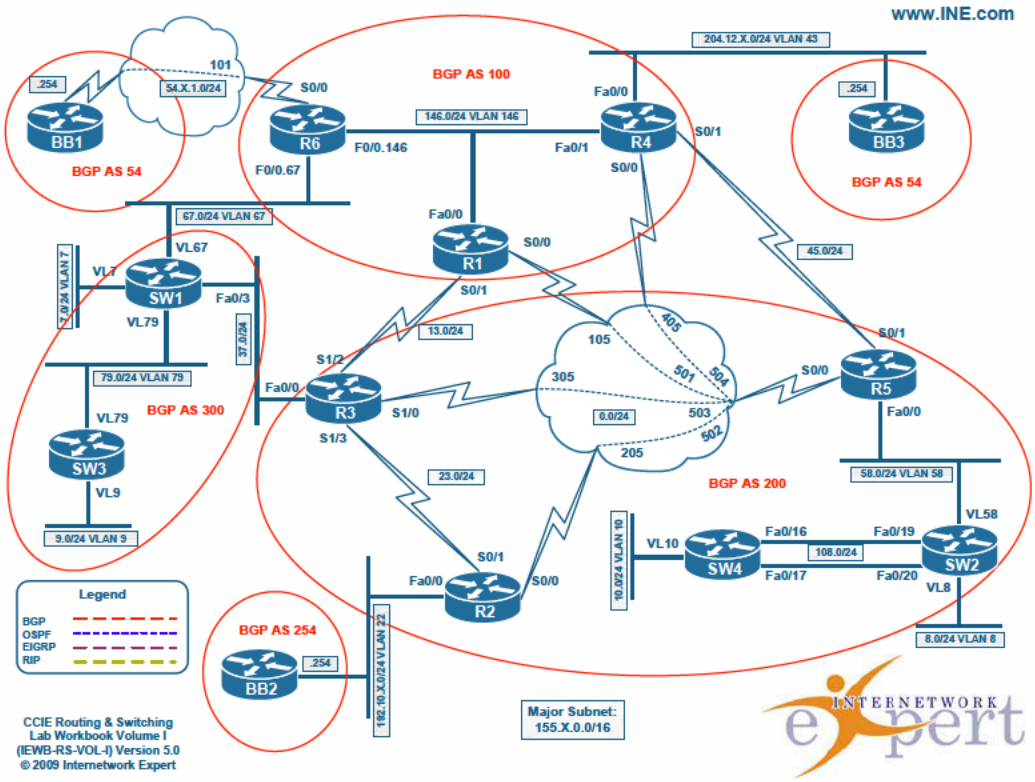
7.17 BGP Auto-Summary

- Configure R4 and R6 to originate *classful* auto-summaries for all of your internally assigned address space.
- BB1 and BB3 should not see any of the subnet advertisements that make up this summary.
- Ensure full reachability from your internal network to all routes learned from AS 54.
- Do not use the **`aggregate-address`** command to accomplish this and use different methods to originate routes at R4 and R6.

Copyright © 2010 Internetwork Expert

Note

Load the the *Basic BGP Routing* initial configurations prior to continuing. Use the diagram below as your reference for the tasks that follows.



7.18 BGP Bestpath Selection - Weight

- Using the most influential attribute, configure SW1 so that traffic from AS 300 going to AS 54 prefixes exits towards R3, and that traffic from AS 300 going to AS 254 networks exits towards R6.

7.19 BGP Bestpath Selection - Local Preference

- Remove the BGP configuration for the previous task from SW1.
- Use the local-preference in R6 so that traffic from AS 100 going to AS 254 transits through AS 300.

7.20 BGP Bestpath Selection - AS-Path Prepending

- Remove the BGP configuration for the previous task from R6.
- Using AS-Path Prepending, configure AS 200 so that traffic from AS 100 going to AS 254 enters via the link to AS 300.

7.21 BGP Bestpath Selection - Origin

- Remove the BGP configuration made for the previous task.
- Using the Origin attribute configure AS 200 so that traffic from AS 100 going to AS 254 enters via the link between R4 and R5.

7.23 BGP Bestpath Selection - MED

- Remove the BGP configuration made for the previous task.
- Using MED configure AS 100 so that traffic from AS 200 going to AS 54 enters via the link between R4 and R5.

7.24 BGP Bestpath Selection - Always Compare MED

- Remove the BGP configuration made for the previous task.
- Create a new Loopback1 interface on both R6 and SW3 with the IP address 1.2.3.4/32 and advertise it into BGP on both R6 and SW3.
- Using just the MED attribute configure the network so that traffic from AS 200 going to this prefix is always received by SW3.

7.25 BGP Bestpath Selection - AS-Path Ignore

- Remove the BGP configuration made for the previous task.
- Ensure that traffic from AS 200 to AS 54 prefixes takes path across AS 300.
- Do not use AS-PATH prepending to accomplish this.

Copyright © 2010 Internetwork Expert

7.26 BGP Bestpath Selection - Router-IDs

- Modify the BGP router-ids in AS 100 as necessary so that traffic from R1 to AS 54 exits via R6.

7.27 BGP Bestpath Selection - DMZ Link Bandwidth

- Remove the Route-ID configurated in the previous section.
- Modify the configuration of AS 100 routers so that R1 load-balances to the paths in AS 54 proportional to the bandwidth of the links connecting R4 and R6 to AS 54 routers.

7.28 BGP Bestpath Selection - Maximum AS Limit

- Remove the previous section's configurations.
- Configure the routers in AS 100 to accept only the prefixes originated from directly connected ASes.
- Do not use filtering based on AP-PATH access-lists to accomplish this.

7.29 BGP Backdoor

- Remove the BGP configuration applied in the previous task.
- Shutdown the BGP peering link between AS 100 and AS 300
- Create a new Loopback1 interface in SW1 with the IP address 150.1.77.77/24 and advertise it into BGP.
- Configure R1 and R4 so that they prefer reaching the new subnet via EIGRP as opposed to eBGP.

7.30 BGP Aggregation

- Remove the BGP configuration made for the previous task.
- Configure R2 with four new Loopback interfaces with the IP addresses 10.0.0.1/24, 10.0.1.1/24, 10.0.2.1/24, and 10.0.3.1/24.
- Advertise an aggregate route for these networks that does not overlap any address space.

7.31 BGP Aggregation - Summary Only

- Modify the configuration for the previous task so that no other devices but R2 could see the specific prefixes that make up the summary.

7.32 BGP Aggregation - Suppress Map

- Modify the solution for the previous task so that R2 advertises 10.0.2.0/24 prefix in addition to the summary route.

Copyright © 2010 Internetwork Expert

7.33 BGP Aggregation - Unsuppress Map

- Remove the summarization configured in R2.
- Using the summary-only feature, configure R3 and R5 to originate an aggregate route for these networks that does not overlap any address space.
- Using the unsuppress-map feature configure the network in such a way that traffic from AS 100 and 54 going to the prefix 10.0.1.0/24 always transits AS 300 unless the link between R3 and SW1 is down.
- Traffic from these ASes going to other subnets of the aggregate should use the direct path through the network.

7.34 BGP Aggregation - AS-Set

- Configure R1 to aggregate the subnets 112.0.0.0/24-119.0.0.0/24 into one prefix using the optimal prefix length.
- Ensure that the new summary prefix is not accepted by AS 54 peers.
- Do not use any filtering technique to accomplish this.

7.35 BGP Aggregation - Attribute-Map

- Configure R4 to mark the prefix 112.0.0.0/24 received from BB3 with the community value of “no-export”.
- Ensure this community value propagates across AS 100.
- Configure R1 so that the summary prefix 112.0.0.0/5 is still advertised to AS 300 and AS 200.

7.36 BGP Aggregation - Advertise Map

- Configure R2 with two new Loopback interfaces with the IP addresses 222.22.0.1/24 and 222.22.1.1/24 and advertise them into BGP.
- Configure SW3 with a new Loopback interface with the IP address 222.22.3.1/24 and advertise it into BGP.
- Configure R4 and R6 to advertise the aggregate 222.22.0.0/22 into BGP. Include as much of the original AS-Path information as possible while still allowing devices in AS 300 to install the aggregate in the BGP table.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Load the the *Basic BGP Routing* initial configurations prior to continuing.

7.37 BGP Communities

- Configure AS200 to set local-preference attribute to 200 for eBGP prefixes tagged with community value 200:200
- Configure R1 in AS100 to signal AS200 to prefer path to the prefixes originated in AS 60 via R3.

7.38 BGP Communities - No-Advertise

- Configure R2 so that it does not advertise prefixes received from AS 254 to any peer.
- Do not use any sort of prefix filtering to accomplish this.

7.39 BGP Communities - No-Export

- Modify R2'2 configuration so that AS254 prefixes are constrained to stay within AS 200 only.

7.40 BGP Communities - Local-AS

- Re-configure R1 and R4 in the same BGP sub-confederation, using the AS# 65014. R6 should be in the sub-confederation 65006.
- Advertise R4's Loopback0 network in the BGP, but make sure that inside AS 100 only R1 receives it.

7.41 BGP Communities - Deleting

- Configure R2 to tag prefixes received from AS 254 with the community values "254:100", "200:254" and "200:123".
- Configure AS 300 to add the community value 300:200 to the list of communities and send them to AS 100.
- Configure AS 300 to remove any communities attached by AS 200, i.e. community starting with "200:" when sending prefixes to AS 100.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Load the the *Basic BGP Routing* initial configurations prior to continuing.

7.42 BGP Conditional Advertisement

- Configure R3 in such a way that AS 300 uses AS 100 to get to all prefixes learned from AS 254.
- If the link between R1 and R3 goes down traffic from AS 300 to AS 254 should be rerouted directly to AS 200.

7.43 BGP Conditional Route Injection

- Configure R2 with four new Loopback interfaces with the IP addresses 10.0.0.1/24, 10.0.1.1/24, 10.0.2.1/24 & 10.0.3.1/24 and advertise them into BGP.
- Configure R2 to originate an aggregate route for these networks that does not overlap any address space. Ensure no other devices in the BGP network see the individual subnet routes of this aggregate.
- Configure BGP Conditional Route Injection on R4 and R6 in such a way that traffic from AS 54 going to the subnet 10.0.1.0/24 enters via R4, while traffic to the subnet 10.0.2.0/24 enters via R6.
- Do not allow the more specific routes to be advertised to R1 from either R6 or R4, and prevent the formation of a routing loop on SW1.

7.44 BGP Filtering with Prefix-Lists

- Configure a prefix-list on R2 so that it does not accept the prefix 222.22.2.0/24 from BB2; this prefix-list should be applied directly to the neighbor.
- Configure a prefix-list on R4 so that it does not accept any prefixes with a subnet mask greater than /22 from BB3; this prefix-list should be applied through a route-map to the neighbor.

Copyright © 2010 Internetwork Expert

7.45 BGP Filtering with Standard Access-Lists

- Remove the filters applied in the previous task.
- Configure a standard access-list on R2 so that it does not accept any prefix with the address 222.22.2.0 from BB2; this access-list should be applied directly to the neighbor.
- Configure a standard access-list on R4 so that it does not accept any prefixes with an even number in the first octet; this access-list should be applied through a route-map to the neighbor.

7.46 BGP Filtering with Extended Access-Lists

- Modify the filtering configuration in R4 as follows.
- Configure an extended access-list on R4 so that it does not accept any prefixes with even 3rd octet and with a subnet mask greater than /22 from BB3.
- This list should apply directly to the neighbor.

7.47 BGP Regular Expressions

- Create a new Loopback 1 on each device R1 – R6 with IP addresses in the format Y.Y.Y.Y/32, where Y is your device number, and advertise them into BGP.
- Configure an AS-Path access-list on SW1 so that AS 300 cannot be used as transit for AS 100 to reach AS 200 or vice-versa; this access-list should be applied directly to its neighbors.
- Configure a local-preference modification on R5 such that traffic from AS 200 going to routes originated in AS 54 is always sent to R4, while traffic to routes that transit AS 54 but were not originated in AS 54 is always sent to R3.
- Additionally configure R3 so that routes learned from AS 254 are not advertised to R1

7.48 BGP Filtering with Maximum Prefix

- Configure R6 so that the peering session to BB1 is torn down if R6 learns more than 20 BGP prefixes from that neighbor.
- Once 16 prefixes are received from BB1, R6 should begin generating warning messages.
- Once down the peering should attempt to be restarted after three minutes.
- Configure SW1 so that if more than 20 prefixes are learned from R3 warning messages should be generated, but the peering session should NOT be terminated.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Load the the *Basic BGP Routing* initial configurations prior to continuing.

7.49 BGP Default Routing

- Configure R2 to originate a default route to R3 and R5 via BGP.
- This default route should be withdrawn if R2's link to BB2 goes down.

7.50 BGP Local AS

- AS 100 is planning transition to the AS number 146. Configure R4 and R6 to use the new AS number while R1 should still use the old AS 100.
- Ensure all BGP peering relationships are still maintained but do not modify the configurations of any routersother than R1, R4 and R6.

7.51 BGP Local AS Replace-AS/Dual-AS

- Re-configure R1 to reside in AS 146 as well and act as a routing reflector for R4 and R6.
- All external Autonomous Systems should be unaware of the new AS numbers used by these routers.
- R5 should peer with R4 using the AS number 146.

7.52 BGP Remove Private AS

- Reconfigure SW1 and SW3 in the private AS 65089 and adjust the peering settings accordingly.
- Create and advertise Loopback 1 subnet in SW1 with the IP address 7.7.7.7/24.
- Configure AS 100 and AS 200 speakers to strip the private AS number when advertising the prefixes to AS 254 and AS 54.

7.53 BGP Dampening

- Create a Looback 1 interface in R1 with the IP address 1.1.1.1/24 and advertise it into BGP.
- Configure AS 200 routers to suppress advertisement of oscillating networks.
- Once a prefix flaps two times in a row, the advertisement should resume in 5 minutes.

Copyright © 2010 Internetwork Expert

7.54 BGP Dampening with Route-Map

- Ensure that dampening process applies only to AS 100 originated routes, and does not affect any other prefixes.

> **Note**
>
> Load the the *Basic BGP Routing* initial configurations prior to continuing.

7.55 BGP Timers Tuning

- Configure R2's BGP process to process conditional route advertisement every 20 seconds.
- R2 should not batch routing updates to BB2 and advertise them immediately.
- Configure R2 so that session deactivation happens within 15 seconds of no session activity.

7.56 BGP Fast Fallover

- Disable the BGP feature in R3 that allows for eBGP peering session deactivation when a physical interface goes down.
- Configure all R3's BGP peering session for fast peering deactivation.

7.57 BGP Outbound Route Filtering

- R1 and R4 should filter out the prefixes 112.0.0.0/8 and 114.0.0.0/8 from being advertised to R3 and R5 respectively.
- The filtering configuration should be applied to routers R3 and R5.

7.58 BGP Soft Reconfiguration

- Configure R4 to accept all prefixes from BB3 irrespective of the configured inbound filters and store them all locally.

7.59 BGP Next-Hop Trigger

- Configure R3 to respond to BGP prefixes next-hop changes within 30 seconds of IGP prefix change.

Copyright © 2010 Internetwork Expert

7.60 BGP TTL Security

- Configure R3 to accept TCP packets from eBGP peers only if they are no more than one hop away.

7.61 BGP AllowAS in

- Configure R2 and SW2 to advertise networks 2.2.2.0/24 and 8.8.8.0/24 into BGP.
- Configure AS 200 border routers so that in case AS 200 is partitioned, the remaining segments could transit AS 100 to recover connectivity.

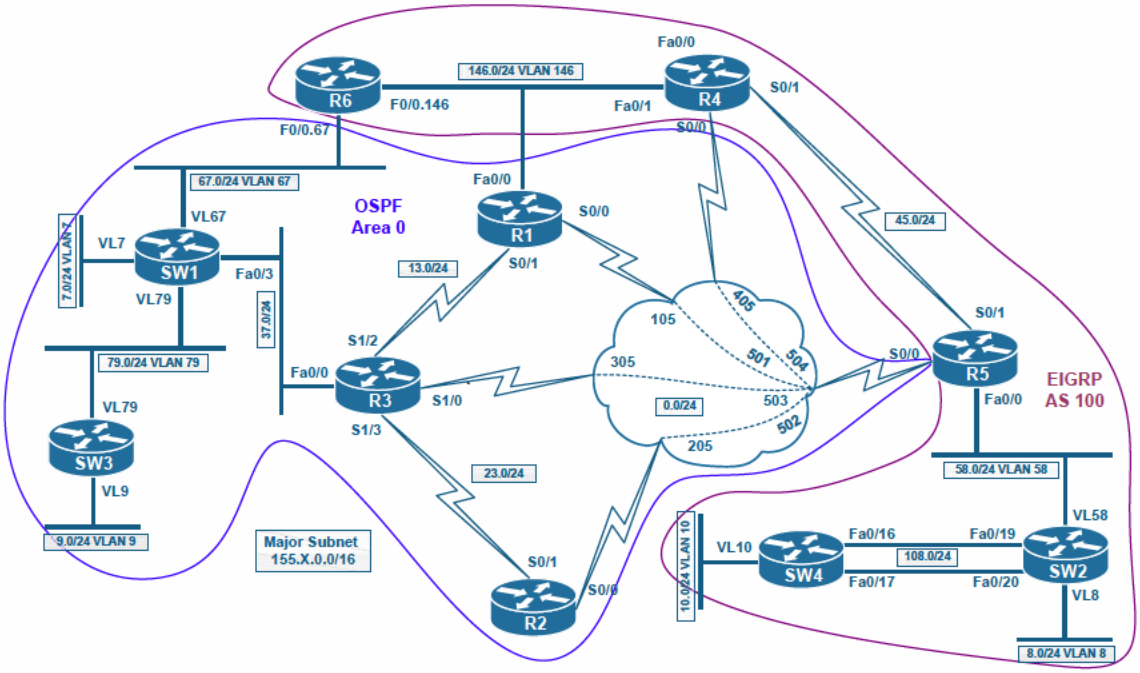
Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Multicast

Note

Load the *Multicast* initial configurations on all devices prior to starting. Refer to the diagram below when working with the following tasks.



8.1 PIM Dense Mode

- Enable dense-mode multicast delivery on the path between R6 and SW4.
- Do not enable PIM on the Frame-Relay cloud.
- To test this configuration, configure SW4's VLAN 10 interface to join the multicast group 224.10.10.10.
- Make sure R6 can ping this multicast group.

8.2 Multicast RPF Failure

- Disable PIM on the Serial link between R4 and R5 and activate it on the Frame-Relay connection between R4 and R5.
- Ensure that R6 can still ping the group 224.10.10.10.
- Set R5's periodic RPF checks interval to 6 seconds.
- To reduce the load on the router's CPU configure R5 to perform triggered RPF checks 10 ms after a topology change but don't allow the delay to ever be longer than 200ms.

8.3 PIM Sparse Mode

- Remove the PIM Dense Mode configuration and replace it with Sparse Mode on all interfaces between R6 and SW4.
- Do NOT run PIM on the Frame Relay link between R4 and R5.
- Remove the static default mroute from R5.
- Use R5's Loopback 0 interface as the static Rendezvous Point address.
- Configure SW4's VLAN 10 interface to join the group 224.10.10.10, and ensure that R6 can send multicast packets to this segment.
- Perform SPT switchover only when the traffic flow exceeds 128Kbps.

8.4 PIM Sparse-Dense Mode

- Remove the previous PIM Sparse Mode configuration and replace it with Sparse-Dense-Mode on all interfaces.
- Do not enable PIM on the Frame Relay link between R4 and R5.
- Modify the static RP assignments so that R5 is only used as the RP for groups in the range 224.0.0.0/8.
- Configure SW4's VLAN 10 interface to join the groups 224.10.10.10 and 239.0.0.1, and ensure that R6 can send multicast packets to this segment for both groups.

Copyright © 2010 Internetwork Expert

8.5 PIM Assert

- Enable multicast routing on R1 and configure PIM sparse-dense mode on the Fast Ethernet interface of R1.
- Configure PIM sparse-dense mode on the connection between R1 and R5.
- Use the same RP assignment that was configured on the other routers.
- Ensure R1 is always elected via PIM to flood traffic onto the shared VLAN 146 segment.

8.6 PIM Accept RP

- Ensure that SW2 and R5 will only accept (*,G) joins toward R5's Loopback 0 for groups 224.10.10.10 and 224.110.110.110

8.7 PIM DR Election

- Ensure that R1 is responsible for multicast source registration with the RP for devices on the VLAN 146 segment.

8.8 PIM Accept Register

- Configure R5 so that the only source allowed on the VLAN 146 segment is R6.
- Your configuration should not affect sources on any other segments.

8.9 Multicast Tunneling

- Without configuring multicast routing on the transit nodes, make sure that receivers on SW3 may receive multicast feeds from R1.
- Do not configure any additional IGP adjacencies to accomplish this task.

8.10 PIM NBMA Mode

- Configure R3 to join the multicast group 224.110.110.110 and make sure R6 can ping it.

8.11 Auto-RP

- Remove all static RP settings and previous Accept-RP/Register configurations.
- Using a Cisco proprietary technology, configure R5 so that it advertises itself as the RP for all multicast groups.

Copyright © 2010 Internetwork Expert

8.12 Auto-RP - Multiple Candidate RPs

- Configure SW2 and SW4 as RPs for group ranges 224.0.0.0-231.255.255.255 and 232.0.0.0-239.255.255.255 respectively.
- In case one RP fails, the other one should provide backup for the other's groups.
- The group 224.110.110.110 should be always switched in dense-mode.

8.13 Auto-RP - Filtering Candidate RPs

- Configure R5 to filter announcements for the group 224.110.110.110 received from SW4.

8.14 Auto-RP Listener

- Switch from PIM SM/DM on all interfaces to PIM SM, except for the loopback 0, and Tunnel 0 interfaces.
- Ensure all devices still hear Auto-RP announcements.

8.15 Auto-RP and RP/MA Placement

- Make R1 another MA that preempts the existing MA on R5.
- Provide a solution for R3 to hear Auto-RP discovery messages from R1.

8.16 Filtering Auto-RP Messages

- Configure R1 so that SW4 cannot hear Auto-RP discovery messages.

8.17 Multicast Boundary

- Configure R5's connection to VLAN 58 so that traffic to the group range 232.0.0.0/5 cannot reach SW2.
- Filter the Auto-RP messages to remove the information about this group range.

8.18 PIM Bootstrap Router

- Remove the Auto-RP configuration on SW2, SW4 and R1.
- Remove the multicast boundary configuration on R5.
- Configure R5 to advertise itself as the RP for all multicast groups using the standards-based protocol.

Copyright © 2010 Internetwork Expert

8.19 BSR - Multiple RP Candidates

- Configure SW2 and SW4 to advertise themselves as RP candidates for all multicast groups.
- R5 should distribute this information and instruct all routers to load-balance multicast groups between the two RPs.
- Use the maximum possible hash mask length to evenly distribute the load across the RPs.

8.20 Filtering BSR Messages

- Configure your network so that R6 does not learn any RP information,

8.21 Stub Multicast Routing & IGMP Helper

- Remove the tunnel connecting R1 and R3. Configure R3 as the stub multicast router with SW1 emulating a host on the stub segment.

8.22 IGMP Filtering

- Only permit R3 to accept IGMP joins for groups in the range 239.1.1.0/24 via its connection to SW1.
- Limit the number of concurrent IGMP states on the interface to 10.

8.23 IGMP Timers

- Configure the designated IGMP querier on the VLAN 146 segment so that failed multicast traffic receivers are detected and removed within 60 seconds.
- Every active receiver should respond to general IGMP queries within 4 seconds.
- Designated querier failures should be detected two times faster than by default.
- Since there is just one receiver on R3's connection to SW1, configure the router to remove multicast group state immediately after a corresponding IGMP leave report has been received.

Copyright © 2010 Internetwork Expert

8.24 Multicast Helper Map

- When R6 sends broadcast UDP packets on port 5000, those packets should be transported across the network and broadcast on the VLAN 37 segment.
- Use the group 239.1.1.100 to accomplish this task and use DNS broadcasts for testing.
- You may use static mroutes if needed to accomplish this task.

8.25 Multicast Rate Limiting

- Configure R3 to limit the aggregate rate of the multicast traffic leaving the VLAN 37 interface to 1Mbps.
- Any source sending to the group IP 239.1.1.7 should be limited to 128Kbps.
- Multicast feeds from R6 to the IP address 239.1.1.100 should be limited to 256Kbps.

8.26 Bidirectional PIM

- The group range 238.0.0.0/8 is used for network video-conferencing with many participants.
- Configure the network so that this group uses a single shared tree rooted on R5 for multicast traffic delivery.

8.27 Source Specific Multicast

- Using the default multicast group range enable PIM SSM functionality on your network.
- R6 should join the multicast feed (150.X.10.10,232.6.6.6).
- Ensure that the “ip pim bsr-border” commands have been removed from R1 and R4 VLAN 146 interfaces.

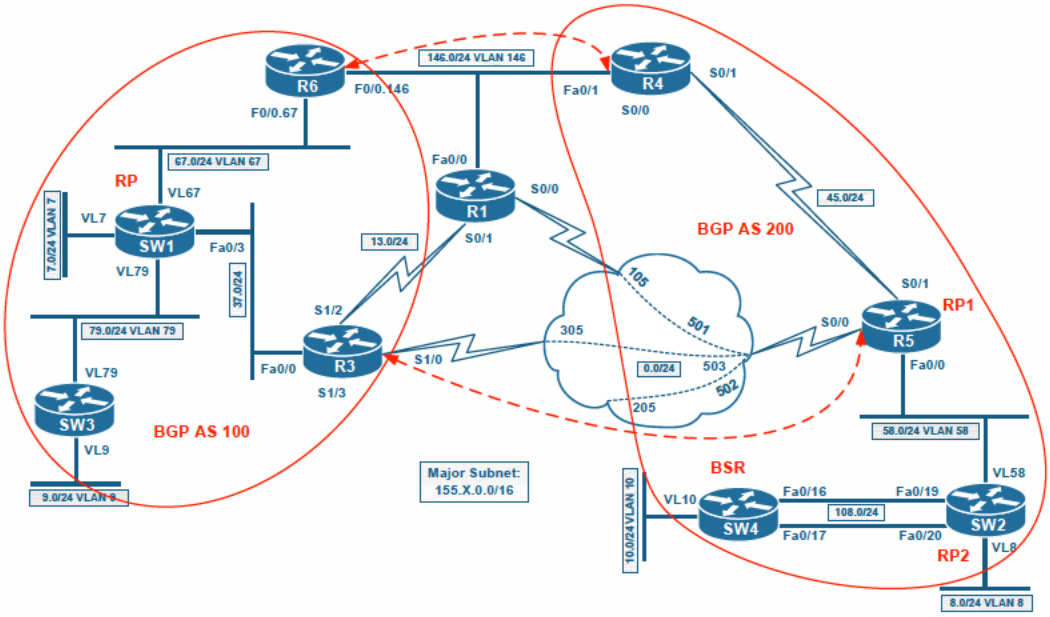
8.28 DVMRP Interoperability

- Configure R4 for this task. Enable multicast routing and configure PIM dense mode on the VLAN 146 interface and Loopack 0 interfaces. Create a DVMRP tunnel sourced off the Loopback 0 interface with a (non-existent) destination 204.12.X.100 (where X is your rack number). Advertise the VLAN 146 subnets to the DVMRP backbone with an offset value of 3. Configure the use of DVMRP routes for RPF check by PIM on VLAN 146 interface.

Copyright © 2010 Internetwork Expert

Note

At this point, erase all devices running configs and load the *Inter Domain Multicast* initial configuration files. Refer to the diagram below, when working with the tasks that follow.



8.29 Multicast BGP Extension

- Enable multicast exchange between AS 100 and AS 200 on both peering links.
- Multicast traffic should prefer to be routed across the Frame-Relay link, using the Fast Ethernet link as backup.

8.30 MSDP

- Change PIM dense-mode on all links where it's configured to PIM sparse-mode.
- Configure R5 as the RP for AS 200 and SW1 as the RP for AS 100. Use the BSR method to distribute RP information, and configure BSR border on the link between R3 and SW1.
- Create an MSDP peering session between SW1 and R5 sourcing it off the Loopback 0 interfaces.

8.31 Anycast RP

- Instead of relying on the BSR protocol to distribute the load between the RPs in AS 200, implement a solution that hides both RPs behind the same RP IP address 150.1.100.100.
- Every RP should inform the other one of the active sources registered with them.

> **Note**
>
> At this point, erase all devices running configs and load the *Basic IP Addressing* initial configuration files.

8.32 Catalyst IGMP Snooping

- Configure R6's Fa0/0 interface with the ip address 155.1.146.6/24
- Join the multicast group 239.1.1.100 on the VLAN 146 interfaces of R4 and R6.
- Configure the switches to assist in optimal traffic flooding across the switched topology for VLAN 146.
- Ensure a switch stops flooding out of a given port as soon as it receives an IGMPv2 report.

Copyright © 2010 Internetwork Expert

8.33 Catalyst Multicast VLAN Registration

- Configure SW1 so that multicast traffic sent by R1 on VLAN 146 is received by R5 on VLAN 58.
- Use the configuration that allows R1 to dynamically tracking sources in other VLANs.

8.34 Catalyst IGMP Profiles

- Configure SW4 to permit only the IGMP reports for groups in ranges 232.0.0.0/8 and 239.0.0.0/8 from R4.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

IPv6

> **Note**
>
> Load the *IPv6* initial configurations, which provide basic IPv4 connectivity across the topology with RIP as the IGP.

9.1 IPv6 Link-Local Addressing

- Configure link-local IPv6 addresses on the Frame-Relay cloud between R1, R2, R3, R4, and R5.
- Use the IPv6 addressing format FE80::Y, where Y is the router number.

9.2 IPv6 Unique Local Addressing

- Enable IPv6 processing on the links between R6 & SW1 and R3 & SW1.
- Use the ULA IPv6 prefixes FC00:X:0:67::Y/64 and FC00:X:0:37::Y/64 for the links between R1 & R6 and R3 & SW1 respectively.
- Here X is your rack number in decimal notation, (e.g. 10 for Rack10, not the hex 0A) and Y is your router number (7 for SW1).

9.3 IPv6 Global Aggregatable Addressing

- Configure globally aggregatable IPv6 addresses in the format 2001:X:0:1234::Y/64 on the Frame-Relay interfaces of R1, R2, R3, R4, and R5.

  Here X is your rack number in decimal notation, (e.g. 10 for Rack 10, not the hex 0A) and Y is your router number (7 for SW1).

9.4 IPv6 EUI-64 Addressing

- Configure globally aggregatable IPv6 addresses in the format 2001:X:0:146::Y/64 on the VLAN 146 interfaces of R1, R4, and R6.
- Here X is your rack number in decimal notation, (e.g. 10 for Rack10, not the hex 0A) and the router will automatically derive Y based on the interface MAC address.

Copyright © 2010 Internetwork Expert

9.5 IPv6 Auto-Configuration

- Use the unique local address FC00:X:0:58::5/64 for R5's VLAN 58 interface, where X is your rack number in decimal notation.
- Configure an additional IPv6 address of FC00:X:0:85::5/64 on the same interface of R5.
- Configure R5 to broadcast both prefixes using ICMPv6 ND RAs, but only allow the hosts to use the second prefix for auto-configuration.
- The valid and preferred lifetime for both prefixes should be set to 4 hours.
- R5 should advertise itself as the default router on the segment every 40 seconds with a lifetime interval of 60 seconds.
- SW2 should learn its IPv6 address on VLAN 58 automatically and use R5 as its default gateway.

9.6 RIPng

- Enable RIPng on the VLAN 146 segment connecting R1, R4, and R6 using the process name "RIPNG".
- Create new Loopback 100 interfaces on R1, R4, and R6 with the IPv6 addresses FC00:X:0:Y::Y/64, where X is your rack number in decimal notation and Y is your router number.
- Advertise these Loopback interfaces into RIPng.

9.7 RIPng over NBMA

- Enable RIPng on the Frame-Relay interfaces of R1, R4, and R5 using the process name "RIPNG".
- Ensure that R1 and R4 can reach each other's Loopback 100 subnets even if any of their VLAN 146 interfaces are down.

9.8 RIPng Summarization

- Create a new Loopback 100 interface on R5 with the IPv6 address FC00:X:0:5::5/64 where X is your rack number in decimal notation and advertise this interface into RIPng.
- Ensure that R1 and R4 advertise only the summary prefix for the Loopback 100 interfaces of R1, R4, and R5 to R6 across the VLAN 146 segment.

Copyright © 2010 Internetwork Expert

9.9 RIPng Prefix Filtering

- Configure R5 to block the IPv6 prefix corresponding to R6's Loopback 100 interface from entering the local routing table.
- Permit any other IPv6 prefixes

9.10 RIPng Metric Manipulation

- Enable IPv6 and RIPng on the Serial link connecting R4 and R5.
- Ensure that R4 prefers the Serial link to reach the Loopback 100 interface of R5 and uses the Frame-Relay link ONLY as a backup.
- Do not use summarization to accomplish this task.

9.11 RIPng Default Routing

- Configure R6 to advertise a default route into the RIPng routing domain with an initial metric of 5.

9.12 EIGRPv6

- Enable the EIGRPv6 routing process on R4, R5, R6 and SW2.
- Enable EIGRPv6 on all interfaces connecting R4, R5, R6 and SW2.
- Create new Loopback 100 interfaces on the mentioned routers with the IPv6 addresses FC00:X:0:Y::Y/64, where X is your rack number in decimal notation and Y is the router number.
- Advertise these Loopback interfaces into EIGRPv6.
- Enable EIGRPv6 on the Frame-Relay interfaces of R4, and R5.
- Authenticate the EIGRPv6 adjacency over VLAN 146 using the password value of CISCO.

9.13 EIGRPv6 Summarization

- Ensure that R5 advertises an optimal summary prefix for the IPv6 addresses of R5's and SW2's Loopback 100 interfaces.

9.14 EIGRPv6 Prefix Filtering

- Configure R5 to block the IPv6 prefix corresponding to R6's Loopback 100 interface from entering the local routing table.
- Permit any other IPv6 prefixes.

Copyright © 2010 Internetwork Expert

9.15 EIGRPv6 Metric Manipulation

- Allow EIGRPv6 to perform load balancing between multiple paths with unequal metrics so long as the metrics are within the range of the optimal metric scaled three times.
- Configure R4's Serial link to R5 to have a delay that is twice good as R4's Frame-Relay link.

9.16 EIGRPv6 Default Routing

- Configure R6 to advertise a default route into the EIGRPv6 routing domain.

9.17 OSPFv3

- Configure OSPFv3 as the routing protocol on the links connecting R3, SW1, and R6.
- Manually configure the router identifiers for all OSPFv3 processes to match the IPv4 Loopback 0 addresses on these routers.
- Use Area 0 on the link between SW1 and R6 and use Area 37 on the link between R3 and SW1.
- Change the network type to point-to-point on the link between R3 and SW1.
- Make the default hello/dead intervals on the segment connecting R6 and SW1 10 times less than the default.

9.18 OSPFv3 over NBMA

- Configure OSPFv3 area 0 on the Frame-Relay links of R5, R2, and R3.
- Use a network type that does not elect a DR, and ensure that OSPF does not send broadcast hello packets.

9.19 OSPFv3 Virtual Links

- Restore the discontiguous OSPFv3 Area 0 by provisioning a virtual link between R3 and SW1.

9.20 OSPFv3 Summarization

- Configure OSPFv3 Area 58 on the link between SW2 and R5.
- Create two new interfaces, Loopback 100 and Loopback 101 on SW2 with IPv6 addresses FC00:X:0:8::8/64 and FC00:X:0:88::88/64 respectively.
- Advertise both Loopbacks into OSPF Area 58 on SW2.
- Configure R5 to summarize the two subnets to an optimal prefix.

Copyright © 2010 Internetwork Expert

9.21 IPv6 Redistribution

- Perform mutual redistribute between OSPFv3, RIPng and EIGRPv6 on R5.
- Ensure each protocol prefers to reach native prefixes using internal paths.

9.22 IPv6 Filtering

- Configure R3's Frame-Relay interface so that only FTP and HTTP traffic from the users on VLAN 67 can access the interface.
- Allow DNS queries and responses from the same subnet, and ensure IPv6 routing will not affected.
- Enable HTTP Server on R5

9.23 IPv6 NAT-PT

- Configure R6 so that SW1 can access the IPv4 address 150.X.4.4 as 2000::9601:404.
- SW1 should source IPv6 packets off its VLAN 67 interface and R6 should translate it to the IPv4 address 155.X.146.7.
- Create a static route on SW1 to ensure traffic for 2000::/96 does not transit SW1's Fa0/3 to reach R6.

9.24 IPv6 MP-BGP

- Configure R1 and R5 in BGP autonomous systems 100 and 500 respectively, and form a peering relationship over the Frame-Relay cloud.
- Create two new loopback interfaces on R1 with IPv6 addresses 2003:X:0:1::1/64 and 2003:X:0:11::11/61, and advertise them into BGP.
- Ensure R5 sees only an optimal summary route encompassing both IPv6 prefixes.

9.25 IPv6 PIM and MLD

- Enable IPv6 Multicast routing on R1, R3, R4, R5 and R6.
- Ensure PIM does not run on the Frame-Relay link between R4 and R5.
- Configure R5 to join the multicast group FF08::8/128 on its VLAN 58 interface.
- R5 should only accept MLD Reports for the group range FF08::/64 and should limit the maximum number of multicast states on the interface towards SW2 to 10.
- Send the general queries every 10 seconds on R5's VLAN 58 interface.

Copyright © 2010 Internetwork Expert

9.26 IPv6 PIM BSR

- Configure R6 as the RP for the multicast domain and R4 as the BSR to propagate this information.
- R3 should be able to ping the multicast group joined by R5.

9.27 IPv6 Embedded RP

- Configure R6 to stop announcing itself as an RP.
- Configure R5 to join the group FF76:0640:2001:CC1E::8 on its VLAN 58 interface.
- Make sure R1 can still ping this group without any router in the domain learning the RP information via the BSR.

9.28 IPv6 SSM

- Configure R5 to join the group FF36::8 but only accept the multicast streams sourced from R6's IPv6 address on VLAN 146.

> **Note**
>
> Clear all router configurations and load the *IPv6* initial configurations.

9.29 IPv6 Tunneling

- Create new Loopback interfaces on R2, R5, and R6 with the IP addresses 2001:X:0:Y::Y/64 where Y is the router number and X is your rack number in decimal notation (e.g. 10 for rack 10, 11 for rack 11 etc.)
- Create tunnels between R5 & R6 and R2 & R6. Use the IPv6 addresses 2001:X:0:56::Y/64 for the tunnel between R5 and R6 and the IPv6 address 2001:X:0:26::Y/64 for the tunnel between R2 and R6.
- The tunnel connecting R2 and R6 should use encapsulation suitable for transporting different Layer 3 protocols.
- The tunnel connecting R5 and R6 should use the encapsulated specifically designed for transporting IPv6 over IPv4.
- Use static routing to obtain connectivity.

Copyright © 2010 Internetwork Expert

9.30 Automatic 6to4 Tunnels

- Using the IPv4 Loopback 0 interfaces create automatic 6to4 tunnels connecting R3, R4, and R5.
- Create additional Loopback interfaces on every router with the subnet number 0 and the prefix length of 64 under the respective 6to4 /48 prefix.
- Use static routing to obtain connectivity between the newly allocated subnets.

9.31 ISATAP Tunnels

- Remove the 6to4 tunnels connecting R3, R4, and R5.
- Using the Loopback 0 IPv4 addresses connect R1, SW1, and SW2 via ISATAP tunnels.
- Use the /64 IPv6 prefix 2001:1:0:345::/64 to allocated IPv6 addresses to the tunnel endpoints.
- Create additional IPv6 Loopback interfaces on all three routers with the IPv6 addresses 2001:1:0:Y::Y/64, and use static routing to obtain full connectivity.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Quality of Service

> **Note**
>
> Load the *QoS* initial configurations prior to starting.

10.1 Hold-Queue and Tx-Ring

- Configure R1's connection to VLAN 146 to have an input software queue length of 10 packets, and an output software queue length of 30 packets.
- Set the output hardware queue size to 15 packets.

10.2 Weighted Fair Queuing (WFQ)

- Configure the point-to-point Serial link between R4 and R5 with an interface clock rate and interface bandwidth of 128Kbps.
- Set the output hold-queue size to 256.
- Configure WFQ on the links with a length of 16 for the congestive discard threshold, 128 for the number of conversations, and 8 for RSVP reservable queues.
- Set the tx-ring size to the minimal value to engage the WFQ as fast as possible.
- Normalize the packet flows for the link by ajdusting the MTU so that each IP packet takes no more than 10ms to be sent.

10.3 Legacy RTP Reserved Queue

- Reset R4's Serial link to R5 to the default fair-queue values.
- Configure a single command on R4's interface so that 100% of the link bandwidth is reserved for RTP traffic in the UDP port range 16384 – 32767.

10.4 Legacy RTP Prioritization

- Reset R4's Serial link to R5 to the default fair-queue values, and remove the RTP reservation.
- Configure legacy RTP prioritization for UDP ports in the range 16383 – 32767 up to 128Kbps.

Copyright © 2010 Internetwork Expert

10.5 Legacy Custom Queueing

- Remove R4's WFQ configuration on the Serial link connecting to R5.
- Configure legacy custom queueing on this link so that traffic is round-robin scheduled based on the following requirements:
  - RTP VoIP traffic should be guaranteed 30% of the link bandwidth.
  - File transfers from web servers in VLAN 146 should be guaranteed 60% of the link bandwidth.
  - The remaining 10% should be allocated to ICMP, but this traffic should not exceed 10 packets in the queue at any given time.
  - RIP routing packets should use the system priority queue.
- Clock the link at 128Kbps, and ensure that IP packets larger than 156 bytes are fragmented.
- Assume packet sizes of 60 bytes for RTP, 156 bytes for TCP, and 100 bytes for ICMP.
- Ensure to account for the layer 2 HDLC encapsulation overhead.

10.6 Legacy Custom Queueing with Prioritization

- Modify R4's custom queueing configuration so that the RTP VoIP traffic is prioritized after the RIP packets.

10.7 Legacy Priority Queueing

- Remove R4's custom queueing configuration and replace it with legacy priority queueing per the following requirements:
  - All RIP packets going out the Serial link to R5 should be sent first.
  - If there are no RIP packets, RTP VoIP should be sent next.
  - If there are no RIP or VoIP packets, web traffic should be sent next.
  - As a last resort ICMP traffic should be sent.
- Set the queue-sizes for the high, medium, normal, and low queues to 5, 40, 60, and 80 packets respectively.

10.8 Legacy Random Early Detection

- Configure R4's point-to-point Serial interface connecting to R5 to avoid tail drop behavior by randomly dropping packets before the output queue overflows.
- Set the hold-queue size to 10 packets, and the weighted averaging constant to 4.
- Ensure that routing updates are never randomly dropped.

Copyright © 2010 Internetwork Expert

10.9 Legacy Flow-Based Random Early Detection

- Enable flow-based RED on the point-to-point Serial interface of R4.
- Set the FIFO queue depth to 10 packets.
- Set the average flow depth scale factor to 2.
- Set the maximum number of flows to 16.

10.10 Selective Packet Discard

- Enable Selective Packet Discard on R1 in aggressive mode.
- Increase R1's input queue size on its link to VLAN 146 to twice the default.
- Increase the amount of the memory headroom for IGP packets to 150 buffers.
- The headroom for BGP packets should be set to 120 packets.
- Start dropping low-priority packets randomly when the input queue is 50% full.

10.11 Payload Compression on Serial Links

- Configure PPP on the link between R1 and R3 using Predictor compression.
- Enable Stacker compression on the HDLC link between R2 and R3.
- Enable FRF.9 compression on the Frame Relay PVC between R2 and R5.

10.12 Generic TCP/UDP Header Compression

- Optimize the bandwidth usage between R4 and R5 by reducing TCP and RTP header sizes.
- Allow for a maximum of 16 concurrent RTP and TCP sessions.
- R5 should optimize traffic only if it detects already optimized traffic from R4.

10.13 MLP Link Fragmentation and Interleaving

- Enable PPP encapsulation on the serial link between R4 and R5.
- Configure PPP Multilink with interleaving on this connection.
- Ensure the maximum serialization delay is 10ms.
- Assume the bandwidth of the link is 128Kbps.

Copyright © 2010 Internetwork Expert

10.14 Legacy Generic Traffic Shaping

- Configure shaping on R6 to limit the rate of packets going towards R4's Loopback0 via the connection to VLAN 146 to 128Kbps.
- Use a shaping interval of 10ms and disable the extended burst functionality.
- Limit the shaper's queue size to 1024 packets.

10.15 Legacy CAR for Admission Control

- Configure R4 to mark traffic received from R1's IP address 155.X.146.1 as it enters R4's connection to VLAN 146 using legacy committed access rate.
- Traffic up to 256Kbps should be marked with an IP precedence of 1.
- Exceeding traffic should be marked with an IP precedence of 0.
- Use an average traffic burst size of 4000 bytes.

10.16 Oversubscription with Legacy CAR and WFQ

- Configure WFQ on R4's point-to-point link to R5, and clock the link at 128Kbps.
- Configure a CAR policy on R4 to achieve the following:
    - 64Kbps that R4 receives from R1 and R6 each should be guaranteed on the link out to R5.
    - Traffic from R1 and R6 should be allowed up to 128Kbps each total.
    - Traffic above 128Kbps should be dropped.
    - Use the averaging time interval of 200ms.

10.17 Legacy CAR for Rate Limiting

- Configure R6 to drop traffic received in its link to VLAN 146 in excess of 256Kbps.
- Use a committed burst of 384Kbps and an excess burst of 768Kbps.

10.18 Legacy CAR Access-Lists

- Configure R1 to rate limit packets going out its link to VLAN 146 towards the MAC address of R4 to 128Kbps.
- Limit the rate of packets leaving R6 towards SW1 having IP precedence values of one, two, and four to 256Kbps.

Copyright © 2010 Internetwork Expert

10.19 Legacy GTS for Frame Relay

- R3's Frame Relay service provider has agreed to transport up to 128Kbps of incoming traffic, but all traffic that exceeds 96Kbps is marked as Discard Eligible.
- Configure R3 with GTS to slow down to the guaranteed rate if the service provider signals congestion with BECN frames.
- Use 30ms for the traffic-shaping interval.

10.20 Legacy Frame Relay Traffic Shaping

- R2, R3, and R5's physical connection rates to the Frame Relay network are 128Kbps, 384Kbps, and 1536Kbps respectively.
- R3's contracted rate is 256Kbps, but the service provider allows occasional packet bursts up to the physical line capacity without discarding them; the provider does not allow sustained oversubscription by R3.
- Configure Legacy Frame Relay Traffic Shaping on R3 and R5 to meet the following requirements:
    - Ensure that R5 does not overwhelm R2's or R3's physical connections.
    - Ensure that R3 adheres to its service contract.
    - Use a Tc value of 10ms to support a VoIP deployment planned in the near future.

10.21 Legacy Adaptive FRTS

- The service provider used by R3 has agreed to allow link oversubscription up to 384Kbps, but only guarantees delivery up to 256Kbps.
- Configure R3 to send as fast as possible on this circuit, but to slow down its sending rate to the guaranteed rate when service provider network notifies it of congestion or the physical interface is congested.
- Ensure that R5 can inform R3 about congestion in the case that R5 is receiving a unidirectional traffic feed from R3.

10.22 Legacy FRTS with Per-VC WFQ

- Configure WFQ in R3's map-class for the PVC towards R5.
- Set the number of flow queues to 32, the queue size to 512, and start discarding packets when any flow reaches a length of 16 packets.

Copyright © 2010 Internetwork Expert

10.23 Legacy FRTS with Per-VC PQ

- Configure R5 to prioritize HTTP traffic sent to R3 over the Frame Relay link, but only if there are no RIP routing updates to send.
- ICMP traffic should only be sent if there is no other traffic queued on the link.

10.24 Legacy FRTS with Per-VC CQ

- Configure R5 to share the bandwidth for the VC connecting R5 to R2 in the following proportions:
    - 60% of bandwidth for HTTP traffic
    - 20% of bandwidth for telnet traffic
    - 10% of bandwidth for ICMP traffic
    - RIP should use the system queue
- Limit each queue size to 15 packets.
- Ensure all protocols use the same average packet size of 80 bytes.

10.25 Legacy FRTS with Per-VC Fragmentation

- Remove any artificial IP MTU settings from the Frame Relay interfaces of R3 and R5 along with any per-VC queueing strategies.
- The connection between R3 and R5 is used to send data and voice traffic.
- Configure Frame Relay fragmentation so that voice packets are never delayed for more than 10ms when serialized.
- Assume the speed of R3's connection is 384Kbps
- Ensure R2 is not affected by this configuration.

10.26 Legacy FRTS with Per-VC IP RTP Priority

- Modify R3 and R5's FRTS configuration to prioritize RTP VoIP traffic on the PVC between them.
- Allocate 128Kbps of bandwidth to voice traffic in the port range of 16384 – 32767.

10.27 Frame-Relay RTP/TCP Header Compression

- Configure R3 and R5 to compress RTP and TCP headers on the PVC between them.
- Compression should not be enabled on any other PVCs.

Copyright © 2010 Internetwork Expert

10.28 Frame-Relay Broadcast Queue

- Modify R5's Frame Relay broadcast queue per the following requirements:
  - Limit the broadcast queue rate a maximum of 128Kbps
  - Limit the queue depth to 4 packets per spoke
  - Limit the total number of broadcast packets to 10 per second

10.29 Frame-Relay DE Marking

- Configure R3 to mark all IP packets larger than 60 bytes as Discard Eligible as they are sent out to the Frame Relay cloud.

10.30 Legacy FRTS PVC Interface Priority Queue

- Configure PIPQ on R5 so that frames going out to R3 are always sent first, followed by frames going to R2.
- All other PVCs assigned to R5's interface should be serviced after R3 and R2.

10.31 Frame-Relay Priority to DLCI Mapping

- Configure back-to-back Frame Relay on the Serial link between R4 and R5 using DLCIs 100 and 200.
- Configure a static mapping to R4 and R5's IP addresses using DLCI 100 on both sides.
- Create a legacy priority-list on both R4 and R5 that maps RIP updates and packets less than 65 bytes to the high queue.
- Map packets from the high queue to the DLCI 100 on both sides.
- Map all other queues to DLCI 200 on both sides.

Copyright © 2010 Internetwork Expert

10.32 Frame-Relay Traffic Policing & Congestion Mgmt

- Enable Frame Relay switching on R3.
- Configure R3's links to R1 and R2 as Frame Relay DCE interfaces, and clock the links at 128Kbps.
- Configure R3 to terminate DLCI 133 to R1 using the IP address 155.X.13.3/24.
- Configure R3 to terminate DLCI 233 to R2 using the IP address 155.X.23.3/24.
- Configure R3 to switch DLCI 132 from R1 to R2 as DLCI 231.
- Configure two point-to-point subinterfaces on R1's link to R3.
- Configure R1 to use the IP address 155.X.13.1/24 and DLCI 133 for the first subinterface, and IP address 155.X.12.1/24 and DLCI 132 for the second subinterface.
- Configure two point-to-point subinterfaces on R2's link to R3.
- Configure R2 to use the IP address 155.X.23.2/24 and DLCI 233 for the first subinterface, and IP address 155.X.12.2/24 and DLCI 231 for the second subinterface.
- Ensure that R3 has IP connectivity to R1 and R2, and R1 and R2 have IP connectivity to each other over these circuits.
- Create two map-classes on R3 for the switched DLCIs with the following policies:
    - The class for R1 should be configured for an input CIR of 64Kbps, an input B_c and B_e of 8000 bits, and an input T_c of 125ms
    - The class for R2 should be configured for an input CIR zero, an input B_c of zero, an input B_e of 16000 bits, and an input T_c of 125ms
- Enable Frame Relay traffic-policing on R3's links to R1 and R2 and apply the above classes to the respective DLCIs.
- Configure generic traffic-shaping on R1 and R2 to average 128Kbps with a B_c of 16000 bits.
- R1 should reduce its sending rate to the guaranteed rate in the case that BECNs are received.
- Enable congestion management on R3's switched PVCs and set the thresholds for DE dropping to 5% and FECN/BECN marking to 3%.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Reset all devices to the *QoS* initial configurations before proceeding.

10.33 MQC Classification and Marking

- Configure an outbound MQC policy on R4's Serial link to R5 per the following requirements:
  - WWW traffic from servers on VLAN 146 should be marked with an IP Precedence of 2
  - VoIP packets with UDP ports in the destination range of 16384 - 32767 and a Layer 3 packet size of 60 bytes should be marked with DSCP EF
  - ICMP packets larger than 1000 bytes should be dropped
  - All other packets that came from R4's connection to VLAN 146 with an IP precedence of 0 should be remarked with an IP precedence of 1
- Do not use an access-list to classify ICMP packets.

10.34 MQC Bandwidth Reservations and CBWFQ

- Modify R4's MQC policy to meet the following requirements:
  - Set the total size of the MQC queue to 512 buffers
  - The traffic flows for the web servers in VLAN 146 and the IP Precedence 0 packets from VLAN 146 should be guaranteed 32Kbps each
  - Limit the sizes of the FIFO queues for the HTTP and IP Precedence 0 traffic classes to 16 and 24 packets respectively
  - All other unmatched traffic in the policy should run WFQ
  - Dynamic flows in this WFQ should start dropping when they reach 32 packets in length

10.35 MQC Bandwidth Percent

- Modify R4's previous user-defined reservations to use a percentage reservation of 25% of the link bandwidth each, as compared to 32Kbps.

Copyright © 2010 Internetwork Expert

10.36 MQC LLQ and Remaining Bandwidth Reservations

- Modify R4's policy to R5 as follows:
  - Configure Low Latency Queueing to allow for exactly one VoIP call to be prioritized
  - Additional VoIP calls should be allowed only if the link is not congested, but they should not be prioritized
  - Reserve 33% of the remaining bandwidth to the HTTP and SCAVENGER classes each
- Assume that VoIP calls are using the G.729 codec, which generates 50 packets per second with a Layer 3 size of 60 bytes.

10.37 MQC WRED

- Modify R4's policy so that the HTTP traffic class uses random detection for dropping as opposed to tail drop.
- Start dropping packets randomly when the average queue size is between 4 and 16.
- Drop 1 of every 4 packets when the average queue size reaches the maximum threshold.

10.38 MQC Dynamic Flows and WRED

- Modify the CBWFQ configuration on R4 to activate random drops for unclassified traffic's dynamic flows.
- Change the minimum and maximum RED thresholds for traffic with an IP precedence of one to 1 and 40 respectively.
- As the queue depth grows close to the maximum threshold, the probability of packet discard should be 25%.

10.39 MQC WRED with ECN

- Modify the HTTP traffic class on R4 so that explicit congestion notification for TCP is used for RED dropping.

10.40 MQC Class-Based Generic Traffic Shaping

- Configure MQC shaping on R6 to limit the sending rate on its link to VLAN 146 to 384Kbps.
- The link to VLAN 67 should be limited to 512Kbps.
- Use a burst interval of 20ms.

Copyright © 2010 Internetwork Expert

10.41 MQC Class-Based GTS and CBWFQ

- Change the queue for the shaper applied to R6's connection to VLAN 146 as follows.
    - Provide 32Kbps of priority treatment for small packets with a network layer size of 60 bytes (voice packets) using a 4000 byte burst value
    - Guarantee the HTTP traffic 256Kbps of the shaper's bandwidth
    - Unclassified traffic should receive fair-queue treatment

> **Note**
>
> Reset all devices to the *QoS* initial configurations before proceeding.

10.42 MQC Single-Rate Three-Color Policer

- Configure R4 to meter incoming HTTP traffic on its link to VLAN 146 as follows.
    - If the traffic rate is less than 128Kbps, mark the packets with an IP precedence of one
    - If the traffic exceeds 128Kbps, mark the packets with an IP precedence of zero
    - Drop violating traffic
- Ensure the burst size is large enough to accommodate normal and excess burst durations of 200ms and 300ms at a rate of 128Kbps.

10.43 MQC Hierarchical Policers

- Configure R4 to limit the aggregate rate of HTTP traffic entering the connection to VLAN 146 to 128Kbps.
- Transmit conforming packets, mark exceeding packets with an IP precedence of zero, and drop violating packets.
- For HTTP traffic flows from R1 and R6, limit the rate to 64Kbps for each flow.
- For these second-level policers, set the conform action to set the IP precedence to 1 and transmit, the exceed action to set the IP precedence to 0 and transmit, and the violate action to set the IP precedence to 0 and transmit.

Copyright © 2010 Internetwork Expert

10.44 MQC Two-Rate Three-Color Policer

- Modify the first-level policy-map applied directly to R4's connection to VLAN 146 to remove the aggregate policer of 128Kbps.
- Modify the second-level policy-map to replace the single-rate policers with two-rate policers.
- Use the CIR value of 64Kbps and PIR value of 128Kbps.
- Use the values of CIR*400ms and PIR*200ms for normal and excess burst sizes.
- Set the conform action to set IP precedence 1 and transmit, the exceed action to set IP precedence of 0 and transmit, and the violate action to drop.

10.45 MQC Peak Shaping

- Configure R1 and R6 to shape HTTP traffic to a peak rate of 128Kbps as it is sent out to VLAN 146.
- Use B_c and B_e bursts based on a 10ms interval.

10.46 MQC Percent-Based Policing

- Hard code R1's FastEthernet link to a speed of 10Mbps.
- Limit the traffic entering this link to 10% of this rate with a burst value of 125ms.

10.47 MQC Header Compression

- Enable MQC based RTP header compression on the serial link between R1 and R3.
- Assume that RTP packets use the port range 16384-32767.
- Allocate enough priority bandwidth to allow 2 G.729 compressed calls over this link, for a total of 24Kbps.

Copyright © 2010 Internetwork Expert

10.48 Using Class-Based GTS for FRTS

- Configure R3 and R5 to shape traffic going out to the Frame Relay network using MQC based GTS.
- R3's policy should match PVC 305, and shape to an average rate of 256Kbps.
- R5's policy should match PVC 503 with a peak shaping rate of 128Kbps, and match PVC 502 with a peak shaping rate of 256Kbps.
- R3 should use a Bc and Be of 12800 and 6400 for the shaper to R5.
- R5 should use a Bc and Be of 6400 for the shaper to R3, and a Bc and Be of 12800 for the shaper to R2.

10.49 MQC Based Frame-Relay DE-Marking

- Modify R5's shaper so that the Frame Relay Discard Eligible bit is set on all frames going out DLCI 502.

10.50 Using MQC CBWFQ with Legacy FRTS

- Configure legacy FRTS on R2's connection to DLCI 205 as follows:
  - Shape to the average rate of 512Kbps
  - Use an interval of 10ms
  - Adaptively shape down to 256Kbps if BECNs are received
- Configure an MQC policy inside this legacy shaper as follows:
  - Guarantee 64Kbps of PVC bandwidth to HTTP flows
  - Other unmatched flows should be guaranteed 192Kbps

10.51 MQC Compatible FRF.12 Fragmentation

- Configure R2, R3, and R5 to fragment packets to 480 bytes as they are sent out to the Frame Relay network.
- Enable fragmentation on the main Frame-Relay interfaces of R3 and R4.
- Enable fragmentation under the map-class for DLCI 205 on R2.

Copyright © 2010 Internetwork Expert

10.52 MQC Based Frame-Relay Traffic Shaping

- Modify the traffic-shaping configuration on R5 and remove the classes matching the Frame-Relay DLCIs.
- Configure MQC shaping to apply per-VC as follows, but do not enable legacy FRTS on the interface
    - Shape peak with CIR values 128000 and 256000 for DLCI 503 and DLCI 502
    - Select a B_c value based on a T_c of 50ms, and a B_e value so that PIR=2*CIR
    - Set the Frame-Relay DE bit for traffic going out DLCI 502
- Replace the interface-based fragmentation with per-VC fragmentation.

10.53 Voice Adaptive Traffic Shaping

- Allocate 64Kbps of priority bandwidth to voice traffic on the PVCs connecting R2, R3 and R5. Classify voice traffic based on a DSCP value of EF
- Since the provider oversubscribes both PVCs on R5's interface, voice quality may degrade when sending traffic at rates over the CIR
- However, it is undesirable to turn off oversubscription and lose extra bandwidth
- Provide a solution to throttle down the sending rate on R3 and R5 down to the CIR only when the IOS detects traffic in the LLQ queue
- Leave the shaping time interval the same at 50ms for simplicity

10.54 Voice Adaptive Fragmentation

- Configure R5 to activate fragmentation only if voice packets are present in the LLQ queue of the respective PVCs

Copyright © 2010 Internetwork Expert

10.55 MLPPP LFI over Frame Relay

- Configure the link between R4 and R5 for Frame-Relay encapsulation. Use any method of your choice
- The company plans to interwork the PVC between R4 and R5 with an ATM connection and run VoIP traffic over both segments
- Suggest a solution that provides a fragmentation and interleaving scheme common for both ATM and Frame-Relay technologies and implement it over the Frame-Relay link between R4 and R5
- Assume the Frame-Relay PVC CIR of 256Kbps and interface rate of 512Kbps
- Provide enough priority bandwidth to accommodate two G.729 calls and guarantee 5% of PVC bandwidth to voice signaling traffic
- Allocate the remaining bandwidth to unallocated traffic
- Classify voice bearer and voice signaling traffic based on DSCP values of EF and CS3 respectively
- Choose the shaping interval optimal for VoIP traffic

10.56 QoS Pre-Classify

- Create a GRE tunnel between R1 and R6 over VLAN 146.
- Route traffic between the Loopback0 subnets of R1 and R6 across the tunnel using static routes.
- Limit the rate of traffic leaving the VLAN 146 interface of R6 to 256Kbps.
- Configure R6 to guarantee 64Kbps of priority bandwidth to IP traffic between the Loopback0 subnets of R1 and R6 marked with DSCP EF on the VLAN146 interface.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Reset all devices to the *QoS* initial configurations before proceeding.

10.57 RSVP and WFQ

- Temporarily shutdown R4's and R5's Frame Relay connection.
- Configure the link bandwidth between R4 and R5 as 128Kbps.
- Enable RSVP on the connections to VLAN 146 on R1, R4, and R6.
- Allow RSVP to use the maximum possible bandwidth on the link between R4 and R5, but do not allow it to reserve more than 64Kbps per flow.

10.58 RSVP and SBM

- Configure R1 as the DSBM candidate to coordinate the bandwidth usage on the segment shared by R1, R4, and R6.
- Allocate just 1024Kbps to RSVP flows on this segment
- Configure R1 to inform other routers on the segment that they may not send above 50Kbyte/sec of non-RSVP traffic into the segment. Limit the burst size to 5Kbyte.

10.59 RSVP and CBWFQ

- Configure CBWFQ on the Serial link of R5 to ensure that HTTP server replies have a guarantee of all interface bandwidth not reserved to RSVP flows.
- Allow for 64 dynamic flow queues with CBWFQ.

10.60 RSVP and LLQ

- Configure R5 to classify RSVP flows with a PQ-profile that ignores peak-rate, and classifies all flows below 2000 byte/s with the burst sizes below or equal to 2000 bytes as eligible for LLQ.

Copyright © 2010 Internetwork Expert

10.61 RSVP and Per-VC WFQ

- Configure the link between R4 and R5 for Frame-Relay encapsulation using DLCI 100.
- Enable legacy Frame-Relay traffic shaping on R5 and shape DLCI 100 down to 96Kbps, with the burst size corresponding to time-interval of 10ms.
- Allow RSVP to use up to 96Kbps of interface bandwidth with 64Kbps per-flow.
- Enable WFQ as the VC queue and configure per-VC WFQ as the RSVP resource provider.

10.62 Catalyst QoS Port-Based Classification

- Enable MLS QoS in all switches.
- Configure CoS to DSCP maps on SW1 and SW2 so that CoS 3 maps to DSCP 26 and CoS 5 maps to DSCP EF.
- Ensure CoS 4 maps to DSCP 44 on SW1.
- Configure IP Precedence to DSCP map in SW4 to map IP Precedence values 3 and 5 into DSCP 26 and DSCP EF.
- Trust DSCP values in packets coming from R1's interface connected to SW1.
- Trust IP Precedence on SW4 port connected to R4 and set the default CoS to 2.
- Override CoS to a value of four on the port connected to R5's VLAN58 interface.
- Use 802.1p bits in the Ethernet headers for classification of packets coming from R6; packets on native VLAN should use a CoS value of one.
- Use interface-level commands only to accomplish the task.

10.63 Catalyst QoS Marking Pass-Through

- Ensure SW2 classifies packets coming from R6's VLANs 67 and VLAN 146 based on CoS values, but does not change the DSCP value in IP packets.
- Additionally ensure that IP-precedence based classification on the SW4's connection to R4 does not change the CoS value in Ethernet headers.

Copyright © 2010 Internetwork Expert

10.64 Catalyst QoS ACL Based Classification & Marking

- Create a policy-map on SW2 that performs the following on traffic received from R6.
  - Set a DSCP value of EF on IPX packets (EtherType value 0x8137)
  - Set a DSCP value of CS3 on ICMP packets
  - Set an IP Precedence of 2 on TELNET packets
  - Trust CoS under the default class
- Configure SW4 and R4 to change the port connected to R4 into a trunk.
- Configure SW4 to trust IP DSCP values in packets coming from R4, and set the CoS to a value of 2.

10.65 Catalyst 3550 Per-Port Per-VLAN Classification

- Configure SW4 so that ICMP traffic coming in on the port connected to R4 has the DSCP value set to CS3.
- For IPX traffic with EtherType 0x8137 set the CoS field to 2.
- Ensure your classification only affects traffic on VLAN 146 but not any VLANs trunked to R4 in the future.

10.66 Catalyst 3560 Per-VLAN Classification

- Enable VLAN-based QoS on the trunk ports of SW1 and on the port connected to R1.
- Configure SW1 to mark all TCP traffic on VLAN 146 with a DSCP value of EF.
- IPX SNAP-encapsulated traffic should be marked with a CoS value of 4.

Copyright © 2010 Internetwork Expert

10.67 Catalyst QoS Port-Based Policing and Marking

- Configure SW4's port connected to R4 as follows:
  - Limit the input rate of ICMP packets to 64Kbps. Meter using a burst size value of 16Kbytes. Trust DSCP values in conforming packets and re-mark exceeding traffic down to CS1.
  - Irrespective of the DSCP value, use a CoS value of two for ICMP packets entering the switch.
  - Mark traffic from WWW servers with DSCP CS2. If the traffic exceeds 256 Kbps, re-mark it down to CS1. Assume the burst size of 32000 bytes.
  - Meter the rate of non-IP traffic at 128Kbps using the minimum burst size. Set a CoS value of 0 for conforming traffic, and re-mark exceeding packets with a CoS value of 1.
  - Assume that incoming ICMP packets are pre-colored either with DSCP AF31 or DSCP CS3.
- Configure SW1 to limit the total rate of traffic coming from R1 to 128Kbps. Use a large enough burst size to accommodate 10ms of traffic generated by R1 transmitting at the full 100Mbps interface rate.

10.68 Catalyst 3560 Per-Port Per-VLAN Policing

- Remove any previous policing configuration on VLAN146 in SW2.
- Configure SW2 to limit the rate of IP traffic entering VLAN146 on every trunk port to 128Kbps.
- Mark all conforming packets with AF21 and exceeding with CS1.
- Use the burst size of twice the minimal size.
- Restrict the amount of IP traffic entering from R6 on VLAN146 to 256Kbps, and drop any exceeding packets. Use the burst size of 32000 bytes.

10.69 Catalyst 3550 Per-Port Per-VLAN Policing

- Remove any previously attached service-policy on the port connected to the VLAN 146 interface of SW4
- Limit the rate of IP traffic from R4 on VLAN 146 to 128Kbps. Mark conforming packets as AF11 and exceeding as CS1
- Limit the rate of non-IP traffic from R4 on VLAN 146 to 256Kbps and drop the exceeding packets. Mark conforming packets with a CoS value of 3
- The configuration should not affect any future VLANs trunked to R4.

Copyright © 2010 Internetwork Expert

10.70 Catalyst QoS Aggregate Policers

- Limit the aggregate rate of ICMP and IPX traffic coming from R1 on SW1 to 128Kbps, and drop exceeding packets.
- Trunk a new VLAN 47 to the port of R4 connected to SW4.
- Limit the aggregate rate of IP traffic to 256Kbps in both VLAN 146 and the new VLAN 47. Mark conforming packets with DSCP AF11 and remark packets to CS1 if they exceed.

> **Note**
>
> Reset all devices to the *QoS* initial configurations before proceeding.

10.71 Catalyst 3560 Ingress Queueing

- Enable MLS QoS on SW1 and trust IP precedence on the port connected to R1.
- Configure ingress queue 1 as priority queue with 15% of ring bandwidth.
- Map all packets marked with CoS values of 5 to the priority queue.
- Ensure all other packets map to the non-priority queue.
- The bandwidth remaining after the priority queue should be shared between ingress queues in proportions 1:3 for priority and non-priority queues respectively.

10.72 Catalyst 3560 Ingress Queue Tuning

- Configure drop thresholds on SW1 for the non-priority ingress queue to 80% and 100%.
- Ensure that routing and network control traffic mapped to queue 2 is less likely to be dropped than regular traffic.
- Ensure that the queue 2 buffer space is 3 times larger than the priority queue's size.

Copyright © 2010 Internetwork Expert

10.73 Catalyst 3550 Egress Queueing

- Network traffic consists of the following traffic classes:
  - VoIP bearer packets marked with DSCP EF
  - Interactive video traffic marked with DSCP AF41
  - VoIP signaling traffic marked with DSCP CS3
  - Interactive Citrix traffic marked with DSCP AF21
  - Best-effort traffic marked with DSCP 0
- Implement the following traffic policy on SW4's connection to R4:
  - Provide priority treatment for VoIP bearer packets
  - Provide a separate queue for video traffic and guarantee it 30% of the remaining bandwidth
  - Provide a separate queue for interactive traffic and VoIP signaling with 20% of the bandwidth
  - The remaining bandwidth should be used by best-effort traffic
- Ensure no other traffic but voice can use the priority queue

10.74 Catalyst 3550 Regular Queues Tuning

- Configure SW4 to set the global min-reserve levels 7 and 8 to 170 (max) and 10 (min) buffers respectively.
- Under the interface connected to R4 map global level 7 to queue 1 and global level 8 to queue 4.

10.75 Catalyst 3550 Gigabit Interface Queues Tuning

- Configure the first GigbitEthernet interface on SW4 as follows:
  - Set thresholds to 50 and 100 for all four queues
  - Map DSCP values 46, 48, and 56 to threshold 2
  - Set WRR queue weights to 20, 20, 20 and 40 for queues 1 through 4
- Enable WRED on the second GigabitEthernet interface of SW4 and configure as following:
  - Set WRED minimum thresholds to 50 and 80 for all queues
  - Map DSCP values 46, 48 and 56 to WRED threshold 2
  - Set WRR queue weights to 20, 20, 20 and 40 for queues 1 through 4

Copyright © 2010 Internetwork Expert

10.76 Catalyst 3550 Egress Policing

- Enable the expedite queue on SW4's connection to R4.
- Ensure only CoS 5 maps to this queue.
- Limit the rate of the expedite queue to 128Kbps with the burst size of 8000 bytes.
- Limit the rate of traffic marked with CS0 to 384Kbps using a burst size of 32000 bytes for metering.

10.77 Catalyst 3560 SRR Shared Mode

- Configure SW1 to set the speed of the port connected to R1 to 10 Mbps, and limit the bandwidth to 20%.
- Preserve the default DSCP to Queue ID mapping.
- Set all SRR shaped weights to zero.
- Set the SRR shared weights to 1, 10, 20, and 30 for queues 1 through 4.
- Enable expedite queue on the interface.

10.78 Catalyst 3560 SRR Shaped Mode

- Change the shaped weight for queue 4 to 1Mbps.
- Set the shared weights to 1, 10, 20, and 1 for queues 1 through 4 respectively.

Copyright © 2010 Internetwork Expert

10.79 Catalyst 3560 Egress Queues Tuning

- Configure SW1 to modify buffer sharing for queues on the interface connected to R1 only.
- Ensure that queue 2 has 70% of interface buffer pools and remaining queues share buffers in equal proportions.
- Reserve only 10% of allocated buffers for queue 2; the remaining queues should reserve all allocated buffers.
- Ensure that traffic marked with AF11, AF21, AF41, and CS1 maps to the second drop threshold in their respective queues.
- Set the first drop threshold to 200% and the second drop threshold to 150% for all queues.
- Ensure that the switch drops voice packets marked with DSCP EF only if the respective queue is exhausted.
- Set the maximum threshold to 250% for all queues

10.80 Catalyst QoS DSCP Mutation

- Configure SW4 so that R1 sees packets marked at R4 with DSCP values CS0, AF31, and CS5 as DSCP values CS1, CS3, and EF.

10.81 Advanced HTTP Classification with NBAR

- Configure R6's VLAN146 interface so that HTTP transfers of files with extensions ".bin", ".text" and ".taxt" are limited to 256Kbps.
- Use only one line to match the URL strings.
- Use the method most friendly for TCP connections.
- Do not apply this limit to the transfers of any other file types.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Security

> **Note**
>
> Load the *Security* initial configurations prior to starting working on the tasks.

11.1 AAA Authentication Lists

- Configure R1 to use a TACACS+ server at the IP address 155.X.146.100 and encrypt communications using a key value of CISCO.
- Source TACACS+ packets from the Loopback0 interface.
- Configure the router so that access to the console line uses local authentication.
- Ensure the VTY lines authenticate users with TACACS+ and then fall back to line authentication.
- Privilege mode authentication should first attempt TACACS+ and then fall back to the local password.
- Create a user named “ADMIN” with a password of “CISCO” in the local database for these configurations.
- Customize the prompts for AAA user authentication and change the default banner message.

11.2 AAA Exec Authorization

- Authorize exec privilege levels using the remote TACACS+ server.
- Ensure that console line logins are also authorized. The console line should authorize users with the TACACS+ method first and then the local database configuration.
- User “ADMIN” should be granted a privilege level of 7 if authenticated.
- Use the VTY line settings to authorize any authenticated users in the event the TACACS+ server fails.
- Ensure that if the TACACS+ server is unavailable, authenticated users logging in via VTY lines are placed in privilege level 15.

Copyright © 2010 Internetwork Expert

11.3 AAA Local Command Authorization

- Ensure that the user "ADMIN" can use RIP debugging commands and can disable any currently active debugging using a single command.
- The same user should be able to configure any interface IP settings and administratively enable or disable any of these interfaces.
- Ensure the user can see their permitted commands in their running configuration.

11.4 Traffic Filtering with Standard Access-Lists

- Using the minimum amount of ACL lines, allow access to R1 from the Loopback0 subnets of R2 through R6 that have an even 3rd octet.
- Deny access from any other "150.X.0.0/16" subnets.
- Any hosts from the "155.X.0.0/16" subnets should still be able to reach R1.
- Apply the access-list inbound on all interfaces of R1.

11.5 Traffic Filtering with Extended Access-Lists

- Configure R4 to permit any TCP traffic destined for R5 and sourced from the Loopback0 subnets. Ensure R4 allows returning TCP packets for those connections.
- VLAN 146 contains servers running FTP, WWW, SMTP and DNS applications.
- Configure R4 to allow access to those applications from behind R5.
- Allow DNS zone file transfers in addition to DNS access.
- Ensure you only permit active FTP responses from servers in VLAN 146.
- Permit the exchange of RIP routing updates.
- Permit IOS "traceroute" and "ping" commands to function across R4.
- Ensure that the Path MTU discovery procedure works successfully across your firewall.
- Deny all ICMP "unreachable" messages with your configuration.
- Deny all other traffic.

Copyright © 2010 Internetwork Expert

11.6 Traffic Filtering with Reflexive Access-Lists

- Configure R5 to permit TCP, UDP, and ICMP packets sourced from behind the VLAN 58 interface.
- Do not create explicit access-list entries to permit returning TCP, UDP, and ICMP traffic.
- Ensure that Telnet and ICMP traffic originated from the router are also permitted by the access-list, but do not create any entries in the interface access-lists.
- Ensure RIP routing traffic is also permitted by your configuration.

11.7 Filtering Fragmented Packets

- Ensure R3 prevents fragmented packets from reaching its local HTTP server.
- Only configure the Frame-Relay interface for this task.

11.8 Access Control with Dynamic Access-Lists

- R6 should restrict access to the outside web servers for the users located on VLAN 67.
- Prior to being allowed to connect to a WWW server, a user should log in to the router using the name “ENABLE” and password of “CISCO”.
- The above procedure should only create an access-list entry for the IP address of the authenticated user.
- Alternatively, any user may connect to port 7001 of R6 and enter the password of “CISCO”.
- This procedure should enable any user on VLAN 67 and behind this VLAN to access the WWW servers.
- Close inactive connections after 5 minutes and do not allow sessions of more than 15 minutes in length.
- A user should be able to extend the maximum duration by logging into R6 repeatedly.

11.9 Traffic Filtering with Time-Based Access-Lists

- Restrict the users behind R2 from accessing BB1 on weekends.
- Additionally, disallow access to BB1 from 6pm to 9am on the remaining days.

Copyright © 2010 Internetwork Expert

11.10 Traffic Filtering with Policy-Based Routing

- Configure R6 to drop ICMP packets of 100 bytes in size entering the VLAN 146 interface.
- Ensure R6 only drops packets leaving via the VLAN 67 interface.
- The router should not send ICMP unreachable notifications about dropped packets.

11.11 Preventing Spoofing with uRPF

- R4 connects to your ISP at the border of your network.
- Ensure that R4 does not accept packets with IP addresses of the internal subnets on its connection to the ISP.
- There is another connection to the same ISP in the network, so account for possible asymmetric routing issues on R4.
- At the same time, R4 should only accept packets from legitimate subnets of 150.X.0.0/16 and 155.X.0.0/16 learned via IGP on its internal connections.
- Log all packets with spoofed sources.

11.12 Using NBAR for Content-Based Filtering

- Configure R6 to drop any potentially dangerous HTTP downloads.
- The dangerous files have extensions “.exe”, “.com”, or “.bin”.
- Use a single pattern to match all the filenames at once.

11.13 TCP Intercept

- Configure R6 to protect the WWW servers on VLAN 146 from a SYN flood attack.
- Intercept connections as they go to the servers and close connections after 30 seconds of inactivity.
- Allow for a maximum of 100 semi-established connections and drop their number down to 80 when the threshold is crossed.
- The router should start dropping half-open connections when they cross the rate threshold of one connection per second. The router should not stop until the average rate is one per two minutes.
- The router should not account for connection age when dropping them.

Copyright © 2010 Internetwork Expert

11.14 TCP Intercept Watch Mode

- Modify your TCP intercept configuration in order to reset any connection that does not reach the established state in 20 seconds.

11.15 Packet Logging with Access-Lists

- Configure R6 to log all ICMP packets entering its VLAN 146 interface.
- Aggregate logging messages so that a log entry is generated after five access-list entry hits.
- Reduce the burden on the router CPU by process switching only one packet per second.
- Ensure the logged messages allow you to track the MAC address of the host sending the packet.

11.16 Stateful Filtering with CBAC

- Configure R4 as a stateful firewall. R4's VLAN 146 interface is the protected network and the Serial interface between R4 and R5 is the connection to outside networks.
- Ensure that TFTP and FTP protocols function across the firewall.
- Collect connection statistics for HTTP transfers.
- Allow pinging across the firewall but do not add any special ACL entries to accomplish this.
- Allow UDP sessions across the firewall with a 30 second inactivity timeout.
- Ensure that inspection entries for DNS requests expire in 10 seconds but do not inspect the DNS protocol for this.
- Disable CBAC alerts for all protocols with the exception of FTP.
- Do not allow any other TCP-based protocols except for those mentioned above.
- In the future some users are going to use the Cisco ezVPN client from inside the firewall. Ensure you do not need to add any ACL entries in order to allow ezVPN connections across the firewall.

Copyright © 2010 Internetwork Expert

11.17 Advanced CBAC Features

- Configure R4 so that you do not have to create manual openings for router-generated UDP traffic in the inbound access-lists.
- You expect no more than 5000 concurrent sessions in the busiest hours. Optimize CBAC performance to handle this configured maximum number of sessions.
- Only allow Java applets downloaded from the WWW servers in the subnet 150.X.5.0/24.
- Someone runs an FTP server listening on port 80. The IP address of the server is 150.X.5.25. Ensure that CBAC correctly inspects FTP sessions to this host.
- Ensure that Java filtering applies to connections through an outside HTTP proxy using port 8080.

11.18 CBAC TCP/UDP Intercept Feature

- Configure R6 to protect the servers on VLAN 146 subnet from TCP/UDP based DoS attacks per the requirements below.
- Allow for a maximum of 100 half-open connections and configure the firewall to keep dropping the sessions until their number reaches 80.
- Limit the new connections rate to 30 per minute and clamp it down to 15 connections per second once the threshold has been crossed.
- Limit the maximum number of half-open TCP connections per server to 10. Block any new connections for 60 seconds when the number exceeds this limit.

11.19 VLAN Filters for IP Traffic

- Restrict traffic on VLAN 146 so that only R1, R4, and R6 are allowed to communicate.
- Allow RIP routing updates as well as transit TCP connections on VLAN 146.
- Disallow any other traffic to cross VLAN 146.

Copyright © 2010 Internetwork Expert

11.20 VLAN Filters for Non-IP Traffic

- Allow only the following Layer 2 protocols across VLAN 146
  - STP BPDUs (consider both ISL and 802.1q trunks)
  - CDP, VTP, and UDLD
  - ARP
- Disallow any other Layer 2 packets.

11.21 Port Security

- Configure the respective switches to guard VLAN146 from MAC address flooding attacks originated at R1, R4, or R6.
- Limit the number of MAC addresses learned simultaneously on a single port to one.
- In case of a policy violation, apply the shutdown action on the port connected to R1 but recover the port after 3 minutes.
- For the port connected to R4, drop offending packets and generate log records of the violation.
- For the port connected to R6, simply drop the offending traffic.
- Age the learned secure entries after 10 minutes of inactivity.
- Retain the MAC addresses learned on the port connected to R4 in the switch configuration.

11.22 HSRP and Port-Security

- Reconfigure R3 and R4 temporarily for this task.
- Configure R1 and R3 to emulate an HSRP virtual router on VLAN 146 with the IP address 155.1.146.254.
- Configure the R4's Fa0/1 interface and the R6's VLAN 146 interface to emulate a virtual HSRP router on VLAN 146 with the IP address 155.1.146.253.
- Ensure your new configuration works with the port-security enabled ports.
- Do not use the **`switchport port-security maximum`** command on the port connected to R4.

Copyright © 2010 Internetwork Expert

11.23 DHCP Snooping

- Configure R4 as a DHCP server and R1 as a DHCP client on VLAN 146.
- Configure SW1 to prevent potential DHCP attacks in such a way that it dynamically accounts for future hosts added to the VLAN 146 segment.
- SW1 should store the binding database in flash with the filename dhcp-bindings.txt, and use a 15 second delay between changes.
- Limit the amount of DHCP messages SW1 can receive from R1 to 10 per second.

11.24 DHCP Snooping and the Information Option

- Configure R6 to obtain IP address information via DHCP from R4.
- Ensure that SW2 inserts Option 82 in DHCP requests received from R6.
- SW2 should be configured so that it uses the string “SWITCH2” for Option 82 Remote-ID and the string “ROUTER6” as the Circuit-ID for the port connected to R6.
- SW2 should accept information options in DHCP packets even on non-trusted ports.

11.25 Dynamic ARP Inspection

- Configure SW2 to prevent ARP poisoning attacks on VLAN 146.
- Do not trust any trunk ports to other switches, but ensure that the switch enforces ARP security for R4 and R1.
- Log all ARP packets permitted by the ARP access-list but limit their rate to four per ten seconds.
- Additionally, log all packets matched against the DHCP snooping database.
- Store, at maximum, 16 entries in the ARP logging buffer.
- Enable all additional sanity checks for ARP packets.

11.26 IP Source Guard

- Configure SW2 to prevent IP address spoofing on the port connected to R6.
- Ensure that your solution accounts for dynamic IP addresses obtained via DHCP.
- Enforce Layer 2 filtering for the MAC addresses corresponding to secured IP addresses at the same time.
- Do not shutdown ports in case of a security violation.

Copyright © 2010 Internetwork Expert

11.27 Using Catalyst Ingress Access-Lists

- Allow R1 to only send ARP and ICMP packets out of its VLAN 146 connection.

11.28 Controlling Terminal Line Access

- Only allow Telnet connections to R2 from the Loopback0 subnets.
- Authenticate remote users locally using the name "TELNET" and the password of "CISCO".
- Only allow this user to connect to R1 from R2.
- Log any attempt to connect from R2 to any destination on port 80.

11.29 IOS Login Enhancements

- After 3 unsuccessful attempts to log into R3 in 30 seconds, block all further attempts for 40 seconds.
- Ensure that sessions sourced from the R5 Loopback0 are exempted from this restriction.
- Log every successful login attempt and every 3rd unsuccessful attempt.
- Enforce a delay of 2 seconds between successive login attempts.
- Create a local username "TEST" with the password of "TEST" for this task.

11.30 Role Based CLI

- Create roles in R4 named "SUPER", "DEBUG", "INTERFACE1" and "INTERFACE2".
- The role "INTERFACE1" should have access to all IP configuration commands of interface Fa0/0 only. For "INTERFACE2" do the same with the exception that it applies to interface Fa0/1.
- The role "DEBUG" should be able to use any **debug** and **undebug** commands, and it should be able to inspect the running configuration.
- The last role name "SUPER" should be a sum of all the other roles.
- Use the password of "CISCO" to authenticate all roles.

Copyright © 2010 Internetwork Expert

11.31 IP Source Tracker

- You suspect that SW1 is under a distributed DoS attack from unidentified sources.
- Configure R6 to collect attack information for the VLAN 67 IP address of SW1.
- Export the statistics via syslog messages every 5 minutes.

11.32 Router IP Traffic Export

- Configure R1 to export the traffic sent and received on the Frame-Relay interface to the MAC address of R4.
- Configure a traffic filter that only allows TCP and ICMP packets to be monitored.

11.33 Controlling the ICMP Messages Rate

- In the future, you plan to apply an input traffic filter on R4's VLAN 146 interface.
- Configure this interface not to respond with an ICMP message when the filter drops a packet.
- The rate of these messages sent out of all other interfaces should not exceed 100 per second.
- In order to ensure that PMTUD works correctly, increase the rate of ICMP messages used by this feature to 1000 per second on R4.

11.34 Control Plane Policing

- Limit the aggregate rate of ARP traffic towards R1's route processor to 100 packets per second.
- The routing control traffic marked with an IP Precedence value of 6 should be limited to 50 packets per second.
- Drop all Telnet connections sourced from R5's Frame-Relay IP address.
- Limit the rate of outgoing ICMP messages to 10 per second.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Erase the running configurations of all devices and load the *Security* initial configurations.

11.35 Control Plane Protection

- Enhance R3 protection by dropping packets going to all closed ports.
- Ensure this does not affect connections made on ports 3020 and 4040.
- To reduce the effect of broadcast storms, limit the rate of input non-IP packets to 100 per second
- Set the queue-limit for input HTTP packets to 100 packets and limit the packet rate to 10 per second.
- Ensure that all fragmented packets transiting the router are limited to 1 Mpps.

11.36 IOS ACL Selective IP Option Drop

- Configure R3 to drop any packets destined to the router with IP source route option (either loose or strict).
- Apply this protection to the link connected to SW1.

11.37 BGP Generic TTL Security Mechanism

- Configure eBGP session between R5 and R6 and make both routers accept the peer if it's no more than one hop away.
- Use AS numbers 500 and 600 for R5 and R6 respectively.

11.38 Flexible Packet Matching

- Configure R1 to filter ICMP Echo packets with the string "AAA" in the payload. Look no deeper than 256 bytes in the packet.
- Ensure the filtering applies only to ICMP/IP packets received in Ethernet frames.

Copyright © 2010 Internetwork Expert

11.39 Zone Based Firewall

- Configure three security zones on R5: INSIDE, OUTSIDE and DMZ for the interfaces facing R4, SW2 and SW3 respectively.
- Create interface VLAN 5 on SW3 and assign the address 155.1.5.9/24.
- Only allow R5 to be accessed using SSH and HTTPs from zones OUTSIDE and DMZ.
- Allow the INSIDE users to use the following protocols when accessing OUTSIDE: HTTP, FTP, ICMP, DNS, SSH, Telnet, AOL Instant Messenger.
- Account for the AOL Messenger using the non-standard port 80 and users connecting to HTTP proxies on ports 3128 and 8080.
- Allow the OUTSIDE users to access the servers in DMZ using the HTTP, FTP, DNS and TACACS+ protocols.
- The INSIDE users should have the same set of protocols allowed to zone DMZ with the addition of SSH and HTTPS.
- Preserve IP routing with your configuration and allow the users logged into the router to ping/telnet to OUTSIDE and DMZ zones.
- Limit the inside users to subnet 150.X.1.0/24 and the DMZ server to the subnet 155.X.37.0/24.

11.40 ZFW Rate Limiting

- In order to prevent DoS attacks, limit the OUTSIDE to DMZ traffic flow to 512Kbps.
- Enable half-open session limiting, so that no more than 2000 half-open sessions are allowed and no more than 100 new half-open sessions generated per minute are permitted.
- Stop dropping the half-open session states once the absolute amount of half-open connections falls below 1000 and the rate falls below 10.

11.41 ZFW Application Inspection

- Configure the firewall so that the internal users are not allowed to use insecure POP3/IMAP4 logins to the outside servers.
- Block image downloads from digg.com for internal users and log any attempts to use AOL messenger for non text-chat service (e.g. file transfer).
- Image files have filename extensions .jpg, .png, .gif
- Log all violations.

Copyright © 2010 Internetwork Expert

11.42 Classic IOS Transparent Firewall

- Configure R6 as a transparent firewall deployed on the subnet 10.0.0.0/24.
- Change the IP addresses for VLAN67 and VLAN146 interface for SW1, R1 and R4 to reflect this.
- VLAN67 users are allowed to originate HTTP, FTP, ICMP, DNS and SSH traffic across the firewall to VLAN146.
- VLAN146 users are only allowed initiating FTP (passive), HTTP and DNS connections to the serves on VLAN 67
- Do not permit any IPv6 traffic across the firewall.

11.43 ZFW-Based IOS Transparent Firewall

- Modify the previous scenario to use CPL for the firewall configuration.

11.44 IOS IPS

- Configure R6 to deny inline the attacker that attempts to use IP loose source route option.
- For testing purposes, enable the ICMP echo signature and configure the IPS to notify syslog.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

System Management

> **Note**
>
> Erase and reload all devices and load the *System Management* initial configurations prior to starting.

12.1 Exec Aliases

- Configure aliases on R1 to accomplish this following:
  - Issuing the command **`ri`** in exec mode should view the routing table contents, while **`rb`** views the BGP table.
  - Issuing **`i Fa0/0`** in global configuration should switch to interface FastEthernet 0/0 configuration mode.
  - Issuing **`iea ACL`** in global configuration should create the extended access-list named ACL.
  - Issuing **`so`** and **`si`** should apply a policy-map to an interface either outbound or inbound respectively.

12.2 System Message Logging

- Enable system message logging on R4 and R6 as follows:
  - Both routers should save debugging messages to their internal buffers up to 8192 bytes.
  - Debugging messages should be sent to the router consoles, but limited to 1 message per second.
  - Console log messages should not interrupt other command output.
  - Users logged via telnet should only see informational level messages and above.

Copyright © 2010 Internetwork Expert

12.3 Syslog Logging

- Configure R4 and R6 to log messages to syslog as follows:
  - Both R4 and R6 should log to the server 155.X.146.100.
  - Log all messages up to notifications.
  - R4 should use reliable transport at port 5000.
  - R6 should use the default transport.
  - Use IDs ROUTER4 and ROUTER6 respectively and the UNIX facility LOCAL1.
  - Messages should be sourced off of the routers' Loopback0 interfaces.
  - Set the message queue depth to 256.

12.4 Logging Counting and Timestamps

- Configure R4 and R6 to count error messages.
- R4 should insert uptime-based timestamps in log messages, while R6 should insert date-time based timestamps.
- R6 should add milliseconds based timestamps to debugging messages only.
- Include year information in logging messages on R6 but not in debugging messages.
- Provide a solution to prevent against tampering with stored syslog information.

12.5 Logging to Flash Memory

- Configure SW1 to log notification messages in its internal flash memory.
- Create the directory /var/log and store messages in the file named "syslog".
- Set the maximum file size to 32768 bytes.

12.6 Configuration Change Notification and Logging

- Configure R4 to locally track changes made to its running configuration.
- Log these changes via syslog, but ensure passwords in the configuration will not be sent across this communication channel.
- Set the queue size to 1000.

Copyright © 2010 Internetwork Expert

12.7 Configuration Archive & Rollback

- Configure SW1 to store its configuration archive on the TFTP server 155.X.58.100 using “sw1-config” as the prefix.
- Archive the configuration every 24 hours.
- Additionally the configuration should be archived every time the running config is saved to NVRAM.

12.8 Logging with Access-Lists

- Configure R4 to log every RIP packet entering the interface connected to VLAN 146.
- Generate a cumulative log entry for every 2 matched packets.
- Send a logging message no more than once per 10 seconds.
- Log the MAC address of the device sending the packet.

12.9 TCP Keepalives

- Configure R1 so that it’s capable of detecting dead TCP connections and closing them prematurely.

12.10 Generating Exception Core Dumps

- Configure R3 to save core dumps to an FTP server at 155.X.146.100 under the name “r3-core”.
- Use active FTP and the username/password values cisco/cisco.
- Configure the router to generate a memory dump and reload as soon as free memory falls below 1Mbyte.
- The router should also reload in the case that memory fragmentation prohibits a process from allocating more than 64Kbytes of memory.
- Disable the local crash information collection.

12.11 Conditional Debugging

- Configure R6 to debug RIP updates entering and leaving the connection to VLAN 67 only.

Copyright © 2010 Internetwork Expert

12.12 Telnet Service Options

- Configure R3 to source all telnet sessions from its Loopback0 interface, and to use a marking of IP Precedence 3 for these packets.
- Do not display any telnet informational messages.
- Create a hostname entry “R4” that pointing to the IP address of R4.
- Ensure that a user typing “R4” in exec mode is connected without seeing the IP address of R4.
- Idle outgoing telnet sessions should signal the remote host to pause output.
- Display the message “Sorry, your connection failed” when a telnet connection to the above host fails.

12.13 Tuning Packet Buffers

- Enable automatic buffer tuning on R3.
- Configure R4 to double the permanent number of all public (not interface specific) buffers.
- Ensure there are at least 10 Large and Huge buffers always available.

12.14 Terminal Line Settings

- Configure R3 to allow only telnet connections to VTY lines 0 through 4 without using an access-list.
- When a user is telnetted into R3 and mistypes a command in exec mode, R3 should not try to open a telnet session to the command as if it was a hostname.
- Configure VTY line 0 to listen for telnet at port 3001.
- When the virtual terminal line is busy issue the output “Sorry, the line is already in use” to the connecting user.
- Exec sessions on a VTY line should timeout after 2 minutes of inactivity; additionally a user should not be able to hold the line busy for more than 5 minutes.
- Sessions initiated from a VTY line should time out in 1 minute.
- When the console line is idle the user should see the output “Welcome to IOS”
- Allow no more than 1 session to be initiated from the console line.
- The terminal length should be no more than 20 lines.
- IP netmasks should be displayed using hex numbers.
- Allow a user to lock VTY terminal lines.

Copyright © 2010 Internetwork Expert

12.15 SNMPv2c Server

- Configure SNMP on R4 with the SNMP contact set to “Default Contact” and the SNMP Location to “Default Location”.
- Enable the SNMP server service using the community string value of CISCO so that remote hosts can modify the local MIB database.
- Allow the system to be reloaded via SNMP.
- Ensure that interface index numbers persist between reloads.
- Only allow configuration transfers via TFTP to/from the host 155.X.146.100.

12.16 SNMPv2c Access Control

- Configure R4 to restrict read-write access to only hosts in VLAN 146.
- Log any other SNMP access attempts using the community string “CISCO”.
- Allow any host to access R4’s MIBs in read-only mode using the community string “PUBLIC”.
- Read-only access should be permitted to the “cisco” MIB subtree only.

12.17 SNMP Traps and Informs

- Configure R4 to send SNMP traps to the management host 155.X.146.100 and informs to the host 155.X.146.101.
- Only send notifications about interfaces going up or down.
- Ensure no informs are being sent about Serial interfaces changing their status.
- Use the community value of “CISCO” with the above traps and informs.

12.18 CPU and Memory Thresholds

- Configure R4 to monitor CPU usage every 5 seconds using the **process cpu** command, and to generate a rising threshold event every time the CPU usage hits 50%.
- Set up the free memory low threshold to 1000Kbytes, and reserve 512Kbytes of memory for the notification process using the **memory** command.

Copyright © 2010 Internetwork Expert

12.19 SNMPv3

- Create two SNMP views named NORMAL and RESTRICTED on R6.
- The NORMAL view should include the branch “iso”.
- The RESTRICTED view should include the branch “ifEntry.*.*n*”, where *n* is the ifIndex of R6’s serial interface.
- Create an SNMP group named NORMAL with the read/write view assigned to NORMAL, and a security model of “priv”.
- Assign the user named NORMAL to this group, set the SHA1 password to CISCO, and the encryption key to CISCO.
- Create an SNMP group named RESTRICTED with the read view assigned to RESTRICTED, and a security model of “auth”.
- Only users in VLAN 146 should be allowed in the SNMP group RESTRICTED.
- Assign the user named RESTRICTED to this group using the security model which only requires authentication using the password CISCO.
- Create an SNMP group named TRAP with the security model “priv”.
- Assign the user named TRAP to this group using the password and encryption key CISCO.
- Enable SNMP traps for LinkUp and LinkDown events only, and send them to the destination host 155.X.146.100 using the security model “priv” and the username TRAP.

12.20 SNMP MAC Address Notifications

- Configure SW1 to notify the NMS station at the IP address 155.X.146.100 about MAC addresses learned on ports connected to R1 and R5.
- Use the community value of CISCO to send notifications.
- Limit the rate of notifications to 1 per second.
- Report both added and removed MAC addresses.
- Store the latest 100 notification events in history buffer.

12.21 SNMP Notifications of Syslog Messages

- Configure R1 so that all debugging and higher priority level messages are sent via SNMP to an NMS host at the IP address 155.X.146.100.
- Set the syslog to SNMP buffer size to 100 messages.

Copyright © 2010 Internetwork Expert

12.22 CDP

- Configure R4 to send CDP announcement every 10 seconds, and instruct other devices to hold the updates for 40 seconds.
- Use the Loopback0 interface for IP address advertisements in CDP messages.
- Disable logging of duplex mismatch detected via CDP messages.
- Do not send or receive CDP message on R4's connection to VLAN 43.

12.23 RMON Alarms

- Configure R1 to monitor the rate of packets entering its interface connecting to VLAN 146 based on the total input packet counter (ifEntry.11).
- If the rate is above 100 packets per minute generate the log message "VLAN146 Interface Congested" and send a trap to the management host at 155.X.146.100.
- Use the SNMP community string CISCO.
- When the packet rate falls below 50 packets per minute, generate another log message "VLAN146 Interface UnCongested" and send a corresponding trap.
- The owner of events should be set to "CISCO".

12.24 RMON Statistics Collection

- Configure SW1 to collect RMON statistics on packets entering the interface connected to R3.
- Store historical information for the last 100 samples of packet statistics taken every 30 seconds.

12.25 HTTP Server and Client

- Configure R4 to support the transfer of the IOS image stored in R6's flash using HTTP.
- R4 should authenticate with a username/password pair of CISCO/CISCO stored in the local database of R6
- R6 should only allow HTTP connections at port 8080, from the subnet 150.X.0.0/16, and limit the maximum number of concurrent connections to two.
- Change the default WWW port to 8080.
- Allow R4 to use secure HTTP connection on port 4043, but limit the security features to use only the DES symmetric cipher.

Copyright © 2010 Internetwork Expert

12.26 FTP Server and Client

- Configure the FTP server service on R6 to serve IOS images off of flash.
- R4 should initiate an FTP session to R6 with the username/password CISCO/CISCO.
- When a transfer occurs R6 should initiate the FTP data channel back to the client.
- Source FTP connections off R4 Loopback0 interface

12.27 TFTP Server and Client

- Configure the TFTP server service on R6.
- When someone requests the file “r6-ios” from R6 it should return its IOS image file.
- Ensure only R1’s Loopback0 interface is allowed to connect and retrieve files via TFTP from R6.
- Configure R1 to source TFTP packets off its Loopback0 interface.

12.28 Remote Shell

- Configure the network so that R1 is to view the running configuration of R6 using remote shell.
- Source remote-shell connections off the Loopback0 interface of R1
- Allow RCP file-copy from R6 to R1.

12.29 NTP

- Configure R4 and R6 as authoritative NTP time sources with stratum 5.
- Both R4 and R6 should synchronize with each other.
- R5 should poll R4 and R6 for time updates, but prefer R4.
- Configure R5 to broadcast NTP updates on its connection to VLAN 58, and for SW2 to listen to NTP broadcasts on this link.
- Configure R6 to send multicast NTP packets to the address 239.1.1.1 on its connection to VLAN 67.
- SW1 and SW3 should be able to synchronize with R6 using these multicast packets.

Copyright © 2010 Internetwork Expert

12.30 NTP Authentication

- Configure R4 and R6 to authenticate each others peering session using the key “CISCO46”.
- R5 should authenticate NTP packets received from R4 and R6 using key values “CISCO4” and “CISCO6” respectively.
- SW2 should only accept NTP updates on VLAN58 if they are authenticated using key value of “CISCO58”.

12.31 NTP Access Control

- Configure R4 and R6 so that no other devices can update their clocks.
- R5 should only allow R4 and R6 to update its clock.
- R4 and R6 should not serve time out to other devices besides R5.
- SW2 should only accept time from R5 and not other sources, but R5 should be allowed to serve time out to other devices.

12.32 Auto-Install over LAN Interfaces using DHCP

- Configure the network to support auto-install as follows:
    - Upon booting R4 should request an IP address for its FastEthernet0/1 interface via DHCP.
    - R1 should forward this request onto R5.
    - R5 should reply with the IP address 155.X.146.4/24, the default gateway 155.X.146.1, the DNS server 155.X.146.6, and the TFTP server “RackXSW2”.
    - R4 should then ask R6 what its own hostname is, and download the config file “RackXR4-confg” from SW2 via TFTP.

12.33 Auto-Install over Frame-Relay

- Disable R1’s link to VLAN 146.
- Configure R5 as a staging router for AutoInstall over Frame-Relay for R4 as follows:
    - R5 should assign R4 the IP address 155.X.0.4.
    - When R5 receives a TFTP request from R4 for the file “network-confg” it should be forwarded to SW2.
    - R4 should download this config from SW2 and learn its hostname RackXR4.
    - R4 should then download the config file “RackXR4-confg” from SW2.

Copyright © 2010 Internetwork Expert

12.34 Auto-Install over LAN Interfaces using RARP

- Configure R1 to supply an IP address to R4 via RARP on VLAN 146 when it boots.
- SW2 should supply the network and host-specific configuration files to R4.

12.35 IOS Menus

- When a user named OPERATOR telnets into R1 and authenticates with the password of CISCO it should see the following menu:

```
Operator Menu


    1           Display IP Routing Table

    2           Display Running Config

    3           Escape to Shell

    4           Disconnect
```

- Option 1 should show the routing table.
- Option 2 should show the running configuration.
- Option 3 should exit to the exec prompt.
- Option 4 should exit the telnet session.
- Clear the screen when the menu starts.

Copyright © 2010 Internetwork Expert

12.36 IOS Banners

- Change the default IOS banners on R3 as follows:
  - All connecting users, with the exception to console users, should see the notification message "Welcome to IOS Router".
  - Before the login prompt is displayed users should see the message "Please, authenticate yourself".
  - Before a user can see the shell prompt, display "Hi, you are on the line <Line> have a nice time at <Hostname>" where <Line> is the line number the user is connected to and <Hostname> is the current router's hostname.
  - Set the banner for reverse telnet connections to "This is a reverse telnet connection".

12.37 KRON Command Schedule

- Configure a KRON occurrence on R4 that saves the running config to a remote TFTP server daily at 08:00.
- Use the server 155.X.146.100 and the filename "r3-config"

12.38 EEM Scripting: Interface Events

- Create an EEM applet in R5 that send an email message to noc@INE.com from r5@ine.com once the Frame-Relay interface receive load exceeds 50% of interface bandwith.
- Use the SMTP server at the IP address 155.X.146.100.
- After this, apply the access-list CRITICAL_TRAFFIC_ONLY ingress on R5's connection to the Frame-Relay interface.
- The access-list should be pre-configured and permit just the HTTP and DNS traffic responses.
- The interface bandwidth is 64Kbps.

12.39 EEM Scripting: Syslog Events

- Create an EEM script in R5 that applies the "no shutdown" command every time someone attempts to shutdown the interface Serial 0/0/0.
- Send an email containing the result of the command "show users" to the address admin@INE.com if this event occurs.
- Make sure your script does not apply to any other interface.

Copyright © 2010 Internetwork Expert

12.40 EEM Scripting: CLI Events

- Configure R5 so that every time someone types the **`show run`** command it does not display the usernames.

12.41 EEM Scripting: Periodic Scheduling

- Configure R5 to send a copy of the running configuration every 5 minutes to noc@INE.com.
- Log a syslog message “Configuration saved” every time the script does that.
- Use the same mail server as in the previous tasks.

12.42 EEM Scripting: Advanced Features

- Create an EEM applet in R5 that clears all Serial/FastEthernet interface counters every 3 minutes.
- Ensure that the Serial interfaces counters are never cleared though.
- Account for any interfaces added to the router in the future.

Copyright © 2010 Internetwork Expert

IP Services

> **Note**
>
> Load the *IP Services* initial configurations prior to starting.

13.1 Proxy ARP

- Disable IP routing on SW1, configure interface VLAN 146 with the IP address 155.X.146.7/24, and shutdown all other interfaces running IP.
- Disable Proxy-ARP on R4's connection to VLAN 146.
- Test reachability from SW1 to the rest of the devices in the routing domain, and check the ARP cache of SW1.
- Configure R5's interface FastEthernet0/0 in VLAN 146 with the IP address 155.X.146.5/24, and disable Proxy-ARP on the link.
- Configure the layer 2 switchports connecting to R1 and R5 on SW1 as protected ports, and enable Local Proxy ARP on R6's connection to VLAN 146.
- Test reachability from R1 to R5 on VLAN 146 and check the ARP cache of both of these devices.
- Revert to the original configuration on R5 and SW1.

13.2 DHCP Server

- Configure R6 as a DHCP server for hosts on VLAN 146, but to ignore BOOTP packets.
- Store the DHCP binding in the flash memory of R6.
- Clients should use R4 and R6 as the redundant default gateways.
- DHCP clients should use the default-router IP addresses for DNS servers.
- Exclude the address range 155.X.146.100 - 155.X.146.254 from allocation.
- Lease the addresses for 12 hours.

Copyright © 2010 Internetwork Expert

13.3 DHCP Client

- Configure R1 to obtain its IP address for the link to VLAN 146 dynamically via DHCP.
- The DHCP client-id should be based off of the MAC address of the interface connecting to VLAN 146.

13.4 DHCP Relay

- Configure R6 to allocate IP addresses via DHCP to clients on VLAN 58.
- The clients on VLAN 58 should use R5 as the default gateway.
- Configure SW2's VLAN 58 interface to obtain an IP address via DHCP.
- Base the client-id on the MAC address of its VLAN 58 interface.
- Ensure to avoid IP address conflicts with the existing hosts on VLAN 58.

13.5 DHCP Host Pool

- Create a DHCP host pool on R6 corresponding to R1's VLAN 146 interface to lease R1 the IP address 155.X.146.1.

13.6 DHCP On-Demand Pool

- Configure the link between R1 and R3 to use PPP.
- Configure R3 to assign the IP address 155.X.13.1/24 to R1 using IPCP.
- Create a DHCP pool on R1 that imports the subnet, the netmask, and the DNS servers (R4 and R6) from R3 via IPCP.
- Ensure that RIP routing is still functional once this configuration is complete.

13.7 DHCP Proxy

- Configure R2 to request an IP address on its serial connection to R3 via IPCP.
- R3 should relay this request via DHCP to R6, and R6 should allocate the IP address 155.X.23.2 to R2.

Copyright © 2010 Internetwork Expert

13.8 DHCP Information Option

- Configure R5 to insert the DHCP information option field into DHCP requests forwarded to R6 from clients on VLAN 58.
- R5 should set the subscriber-id string for these requests to “VLAN58”.
- Configure R6 to allocate IP addresses based on Option 82 and assign a static IP address to SW2 based on the information option contents.

13.9 DHCP Authorized ARP

- Configure R6 to update the ARP cache on its VLAN 146 interface with the information from the DHCP binding database.
- Ensure that R6 does not accept any ARP mappings on its VLAN 146 interface other than those learned from the DHCP binding database.

> **Note**
>
> Revert all devices to the *IP Services* initial configurations prior to continuing.

13.10 IP SLA

- Configure R6 to automatically monitor BB1’s availability using ping and telnet every one minute.

13.11 Object Tracking

- Create two enhanced objects on R6 to track the state of previously created IP SLA operations.
- Create a third enhanced object on R6 that tracks the first two objects; this third object should only be “up” if both of the first two objects are “up”.
- The first two objects should wait 10 seconds before reporting their state from up to down, or from down to up.

Copyright © 2010 Internetwork Expert

13.12 HSRP

- Configure HSRP on R4 and R6 so that hosts on VLAN 146 can use the virtual IP address 155.X.146.254 as their default gateway.
- Ensure that R6 is the active physical gateway unless it loses connectivity to VLAN 146.
- Set the hello interval to 1 second and dead interval to 3 seconds for faster failed gateway detection.
- Authenticate HSRP message exchanges using an MD5 hash of the key CISCO.

13.13 VRRP

- Configure VRRP on R4 and R6 so that hosts on VLAN 146 can use the virtual IP address 155.X.146.253 as their default gateway.
- Ensure that R4 is the active physical gateway unless it loses connectivity to VLAN 146.
- Set the hello interval to 3 seconds, and authenticate messages using the clear-text password CISCO.

13.14 GLBP

- Configure GLBP for hosts on VLAN 146 so that first-hop redundancy and load balancing occur between R4 and R6.
- Ensure R6 is elected as the Active Virtual Gateway (AVG), and both R4 and R6 are used as Active Virtual Forwarders (AVFs) for the virtual IP address 155.X.146.252.
- Use a 1 second hello interval and a 3 second hold time.
- Distribute the traffic load between R4 and R6 in 2:1 ratio.
- Authenticate GLBP UDP packet exchange using a secure method based on a password value of CISCO.

13.15 Router Redundancy and Objects Tracking

- R6 should be the active HSRP gateway as long as it can ping BB1 and reach it through telnet, but give up the active role once it loses either telnet or ping reachability to BB1.
- Ensure R4 is the active VRRP gateway when it sees the prefix 30.0.0.0/16 received via IGP from BB3.

Copyright © 2010 Internetwork Expert

13.16 IRDP

- R5 and SW2 should advertise themselves as default gateways for hosts on VLAN58 using ICMP messages.
- R5 should be the preferred gateway on the segment.

> **Note**
>
> Revert all devices to the *IP Services* initial configurations prior to continuing.

13.17 Router ICMP Settings

- Configure R1's VLAN146 interface to stop sending ICMP messages about discarded packets, ICMP messages to select a better next-hop, and ICMP messages reporting subnets mask.
- Rate-limit ICMP unreachables to 2 per-second globally.

13.18 Basic NAT

- Ensure that R4 and R6 are not advertising any internal networks (i.e. 155.X.0.0/16 or 150.X.0.0/16) to BB3.
- Configure R4 and R6 to translate IP source addresses for packets from the subnet 155.X.0.0/16 going to BB3 and BB1 respectively using a dynamic pool consisting of the subnets of their BB connections.
- Do not use any route-maps to accomplish this.

13.19 NAT Overload

- Configure R4 and R6 to translate packets sourced from the Loopback0 subnets (150.X.0.0/16) using their backbone connection IP addresses.

13.20 NAT with Route Maps

- Configure R2 with route-map based NAT to support multiple outside interfaces.
- Traffic from R2's Loopback0 network going out the point-to-point link to R3 should be translated to 155.X.23.200.
- All other traffic going out the point-to-point link to R3 should be translated to R2's interface IP address.
- Traffic from R2's Loopback0 network going out the Frame Relay link to R5 should be translated to 150.X.2.200.
- All other traffic going out the Frame Relay link to R5 should be translated to R2's interface IP address.

Copyright © 2010 Internetwork Expert

13.21 Static NAT

- Configure R5 so that R4 can reach SW2's VLAN 58 IP address using the address 155.X.45.8.
- Additionally SW2 should be able reach R4's point-to-point link using the VLAN 58 IP address 155.X.58.4.

13.22 Static PAT

- Remove the previous static NAT entries on R5.
- Configure R5 so that when R4 telnets to 155.X.45.44 at port 8023 they get connected to SW2's VLAN 8 interface at port 23.
- Configure R5 so that when R4 telnets to 155.X.45.44 at port 10023 they get connected to SW4's VLAN 10 interface at port 23.

13.23 NAT and IP Aliasing

- Configure R5 so that it does not add a local alias for the static mapping of the IP address 155.X.45.44.

13.24 Static Policy NAT

- Remove all previous NAT configurations from R2.
- Configure a static translation on R3 so that when traffic is received from SW1's VLAN 7 IP address and it is going out the point-to-point link to R1 it is translated to the address 155.X.13.7.
- Configure a static translation on R3 so that when traffic is received from SW1's VLAN 7 IP address and it is going out the point-to-point link to R2 it is translated to the address 155.X.23.7.

13.25 NAT with Overlapping Subnets

- Create a new Loopback1 interface on R1 and R2 using the IP addresses 10.0.0.1/24 and 10.0.0.2/24 respectively.
- Advertise the 10.0.0.0/24 network into RIP on R2.
- Configure NAT on R1 so that when R2 telnets to the address 11.0.0.Z from its Loopback1 address it is redirected to the address 10.0.0.Z attached to R1, where "Z" is any number.
- The host 10.0.0.Z on R1 should see the packets from R2 as if they were sourced from the network 22.0.0.0/24.

Copyright © 2010 Internetwork Expert

13.26 TCP Load Distribution with NAT

- Remove the previous NAT configurations on R5.
- Configure rotary NAT on R5 so that when SW2 telnets to the address 155.X.58.55 it is redirected to R1, R2, and R3 in an even distribution.

13.27 Stateful NAT with HSRP

- Remove all previous NAT configurations on R4 and R6.
- Configure HSRP on R4 and R6 with the group name “VLAN146” so that hosts on VLAN 146 can use the virtual IP address 155.X.146.254 as their default gateway.
- Confgure R6 to be the active physical gateway unless the line protocol of the link connecting to BB1 goes down.
- Set the hello interval to 1 second and dead interval to 3 seconds for faster failed gateway detection.
- Authenticate HSRP message exchanges using an MD5 hash of the key CISCO.
- Create a single NAT pool on R4 and R6 with the IP addresses in range 155.X.254.1-155.X.254.254.
- Configure BGP AS 100 on R4 and R6, and peer these routers with BB3 and BB1 respectively, who are in AS 54.
- Advertise the NAT pool via EBGP to BB1 and BB3.
- Redistribute all BGP routes into RIP on R4 and R6, and ensure that R6 is the preferred exit point.
- Configure dynamic NAT on R4 and R6 so that traffic from the 155.X.0.0/16 network is port translated to the previously defined NAT pool.
- Ensure R4 and R6 synchronize their NAT entries using HSRP so that TCP sessions from the inside network connected to devices in AS 54 are not dropped if R6 loses connectivity to BB1.

13.28 Stateful NAT Primary/Backup

- Remove the HSRP configuration on R4 and R6.
- Modify the SNAT configuration on these devices so it does not rely on HSRP status for tracking of the active SNAT router.
- R6 should be the primary and R4 should be the backup router.

Copyright © 2010 Internetwork Expert

13.29 NAT Virtual Interface

- Remove all previous NAT configurations on R5.
- Configure R5 so that traffic sourced from SW2's VLAN 8 network is translated to the source address range 155.X.188.0/24.
- Do not use the **`ip nat inside`** or **`ip nat outside`** commands or static routing to accomplish this.

13.30 NAT Default Interface

- Remove all previous NAT configurations on R5.
- Disable R5's connection to the Frame Relay network.
- Configure R5's point-to-point link to R4 as a passive-interface under RIP.
- Configure R5 so that all connections destined to its point-to-point Serial link are redirected to SW2's Loopback0 interface.
- All other inside to outside traffic on R5 should be dynamically overloaded to the point-to-point Serial link's IP address.

13.31 Reversible NAT

- Remove all previous NAT configurations on R5.
- Configure R5 so that all traffic received on the inside interface connecting to VLAN 58 is translated to the pool 155.X.45.100 – 155.X.45.200 as it exits out the point-to-point link to R4.
- Use route-map based NAT for this configuration to allow outside hosts to initiate connections back to hosts on the inside network once an inside to outside dynamic entry is created in the NAT table.

13.32 Static Extendable NAT

- Configure R5 so that SW4's Loopback0 address is reachable from the outside of the network using the IP addresses 155.1.45.201 and 155.1.45.202 at the same time.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Revert all devices to the *IP Services* initial configurations prior to continuing.

13.33 IP Precedence Accounting

- Configure R6 to account for the number of input and output packets based on their IP Precedence values on it's connection to BB1.

13.34 IP Output Packets Accounting

- Configure R1 to account for the number of output packets on all physical interfaces.
- Only account for packets going to subnets of 155.X.0.0/16.
- Limit the accounting database size to 4096 entries.
- The number of records for packets not matching the account list should be set to 1.

13.35 IP Access Violation Accounting

- Configure R6 to deny packets with an IP precedence of 4 coming from BB1.
- Account for the packets denied by this policy.

13.36 MAC Address Accounting

- Configure R1 to account for the MAC addresses in IP packets entering or leaving its connection to VLAN 146.

13.37 TCP Optimization

- Modify R1's TCP configuration to meet the following requirements:
    - Avoid the TCP "silly window syndrome".
    - Enable high-performance TCP options.
    - Set the TCP window size to twice the standard 16 bit maximum value.
    - Limit the wait time for a TCP SYN response to the minimum.
    - Enable the feature to avoiding fragmentation with TCP sessions.
    - Hold no more that 16 packets in outgoing TCP queue.

Copyright © 2010 Internetwork Expert

13.38 IOS Small Services & Finger

- Configure R1 to meet the following requirements:
  - Users who telnet or send UDP packets to R1 at port 7 should have any typed characters echoed back to them.
  - Users who telnet or send UDP packets to R1 at port 9 should have any typed characters thrown away.
  - Users who telnet to R1 at port 13 should be returned the date and time.
  - Users who telnet or send UDP packets to R1 at port 19 should be sent a stream of ASCII characters.
  - Users who telnet to R1 at port 79 should be sent a list of currently logged in users.

13.39 Directed Broadcasts & UDP Forwarding

- Configure R5 so that if it receives packets for the destination address 155.X.58.255 they are forwarded out the link to SW2, but the destination should not be changed to 255.255.255.255.
- Configure R5 to listen to UDP broadcasts on the point-to-point link to R4, and forward them to the address 155.X.58.255.
- This forwarding should only occur for DNS packets.
- Verify this configuration by sending ICMP pings to the address 155.X.58.255 and DNS broadcast requests from R4.

13.40 DRP Server Agent

- Configure R1 to support reporting of information to a Distributed Director.
- Only allow connections from Directors from VLAN 146, authenticate the communication using the key value CISCO.

13.41 WCCPv1 Web-Cache

- Configure R4 and R6 for WCCPv1 web-cache redirection.
- R4 should redirect outgoing packets on its VLAN 43 interface, and R6 should redirect incoming packets on its VLAN 146 connection.
- Exclude packets coming in R4's Serial connection to R5 from redirection.
- Exclude connections sourced from the VLAN 146 subnet from redirection.
- Do not redirect flows excluded by outbound ACLs.

Copyright © 2010 Internetwork Expert

13.42 WCCPv2 Services

- Configure R5 to listen for WCCPv2 cache engines on its FastEthernet0/0 link using a dynamic service-group numbered 50.
- The cache engines will be sending packets to the multicast address 224.0.1.100.
- Authenticate them with the password CISCO.
- Only accept packets from the cache engines 155.X.58.100 and 155.X.58.200.
- Redirect traffic from hosts in the VLAN 58 subnet to these cache engines.

> **Note**
>
> Revert all devices to the *IP Services* initial configurations prior to continuing.

13.43 NBAR Protocol Discovery

- Configure R4 and R6 to collect protocol statistics on their connections to backbone routers using NBAR.
- Both routers should be able to classify connections to HTTP proxy ports 3128 and 8080.
- Change the SOCKS protocol mapping to port 2080.
- Create a new custom protocol mapping called TEST that matches the ASCII character “A” in the beginning of a TCP segment flowing to the destination port 3001.

13.44 Netflow Ingress & Egress

- Configure R4 and R6 to export Netflow version 5 information to a host 155.X.146.100 in VLAN 146 using port 9999.
- R4 should collect statistics on incoming flows on all interfaces, while R6 should collect statistics on outgoing flows on all interfaces.
- Limit the size of the flow cache to 4096 entries and export the BGP AS of Origin if available.
- Collect additional information about ICMP message information as well as layer 2 VLAN numbers where applicable.

Copyright © 2010 Internetwork Expert

13.45 Netflow Top Talkers

- Configure R4 and R6 to collect information about 10 most active ICMP flows.
- R4 should count packets, while R6 counts bytes.
- Match only TCP and ICMP flows coming from the subnet 155.X.0.0/16.

13.46 Netflow Aggregation Cache

- Create an aggregation cache that is destination-prefix based on R4 and source-based on R6.
- Export the new information to 155.X.146.100 at port 9998 with Netflow version 8.
- Do not summarize addresses to more than a /8 prefix length.
- Store no more than 1024 aggregated entries.

13.47 Netflow Random Sampling

- Modify R6's Netflow configuration so that it only samples every 10th packet in each flow.

13.48 Netflow Input Filters

- Modify R6's Netflow sampling so that every packet is sampled for sources on VLAN 146.
- Other packets should still be randomly sampled.

Note

Revert all devices to the *IP Services* initial configurations prior to continuing.

Copyright © 2010 Internetwork Expert

13.49 IOS Authoritative DNS Server

- Configure R4 and R6 as primary DNS servers for the domain cisco.com.
- The primary name server should be called ns.cisco.com with a contact mailbox of ccie.cisco.com.
- Both R4 and R6 should resolve the names R4.cisco.com and R6.cisco.com to all connected IP addresses of the respective router.
- R1 should use both R4 and R6 as its DNS servers in a round-robin fashion.
- R1's DNS client should complete all unqualified domain-names with the name "cisco.com".

13.50 IOS Caching DNS Server

- Configure R3 as a DNS client of R1.
- When R1 receives a DNS query from R3 it should check its local cache.
- If the entry is not cached a request should be sent from R1 to R4 or R6 on behalf of R3.

13.51 IOS DNS Spoofing

- Configure R5 as a DNS client of R4.
- Configure SW2 as a DNS client of R5.
- If R5 loses connectivity to R4 it should respond to all DNS queries with the IP address of its Loopback0 interface.

13.52 IP Event Dampening

- Configure dampening on R6 so that after a reload its connection to BB1 is not advertised into IGP for 30 seconds.
- Additionally configure R6 so that when this connection flaps it does not disappear for more that 60 seconds from the routing table no matter how much penalty it accumulates.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

MPLS VPN

> **Note**
>
> Load the *MPLS VPN* initial configurations prior to starting.

14.1 VRF Lite

- Create a new VLAN 76 connecting R6 and SW1 using the subnet 155.X.76.0/24.
- On R6 and SW1 configure VLAN 67 so that it belongs to vrf group VPN_A and VLAN 76 should belong to a different vrf group named VPN_B.
- Create Loopback interfaces Lo 101 and Lo 102 on SW1 and place them in the VRF tables VPN_A and VPN_B respectively, using the IP addresses 172.16.7.7/24 for Lo 101 and 192.168.7.7/24 for Lo 102.
- Both VRFs on SW1 should use R6 as the default gateway.
- Configure R6 so that from SW1 you can ping interface Lo 101 from VPN_B and Lo 102 from VPN_A.

14.2 MPLS LDP

- Configure R4, R5 and R6 for MPLS label exchange using an IETF standard protocol.
- Authenticate LDP peering sessions using an MD5 hash and the password value of CISCO.
- Use only a single command on R4 to enable LDP on all OSPF-enabled interfaces.
- Use the physical interface's IP addresses for the LDP session between R4 and R6.

14.3 MPLS Label Filtering

- Configure R4, R5 and R6 so that the only labels advertised by LDP are the ones for the Loopback 0 interfaces of the mentioned routers.

Copyright © 2010 Internetwork Expert

14.4 MP-BGP VPNv4

- Create two new VRFs named VPN_A and VPN_B on R5 and assign the VLAN 58 and VLAN 5 interfaces into these VRFs repectively.
- Enable VPN route exchange between R5 and R6 use R4 as the BGP route-reflector.
- Make sure IPv4 peering sessions are not activated by default.
- By the end of this task you should be able to ping the connected interfaces inside the VPNs.

14.5 MP-BGP Prefix Filtering

- Create a new Loopback 101 interface in R5's VRF VPN_A with the IP address of 172.16.5.5/24.
- Create a new Loopback 102 interface in R6's VRF VPN_B with the IP address of 192.168.6.6/24.
- Configure the network to provide bi-directional connectivity between the two new subnets.
- Make sure R6's VPN_A does not see the prefix 172.16.5.0/24 and R5's VPN_B does not see the prefix 192.168.6.0/24.

14.6 PE-CE Routing with RIP

- Use RIP as the PE-CE routing protocol for VPN_B sites.
- Remove all static routing used to provide VPN_B connectivity.
- Configure R4 so that the VLAN 43 connection belongs to VPN_B as well.
- Preserve the RIP metric values learned from the CE routers.

14.7 PE-CE Routing with OSPF

- Use OSPF as the PE-CE routing protocol for the VPN_A sites and configure PE-CE routers in area 1.
- Use the same OSPF process IDs at R6 and R5 but ensure SW1 and SW2 can reach each other.
- Create a new Loopback interface on SW2 with the IP address 172.16.8.8/24 and make sure R6 sees only a /16 summary for this prefix.

Copyright © 2010 Internetwork Expert

14.8 OSPF Sham-Link

- Reconfigure SW1 to a pure CE router in VRF VPN_A; be sure to remove the VRF-lite configuration.
- Preserve the OSPF routing between VPN_A sites at R5 and R6.
- Provision a back-door link between SW1 and SW2 using any of the interconnections as a L3 interface.
- Configure VPN_A OSPF processes on R5 and R6 for the same domain-id.
- Ensure SW1 and SW2 prefer the path across the MPLS core to the backdoor link.

14.9 PE-CE Routing with EIGRP

- Replace OSPF with EIGRP on the PE-CE as the routing protocol used for VPN_A.
- Ensure the backdoor link is used for backup and that the primary path between the two sites is across the MPLS VPN cloud.
- Make R4's VLAN 43 interface a part of VPN_A and advertise it into EIGRP.
- All EIGRP routers should be in the same AS.

14.10 EIGRP Site-of-Origin

- Configure R5 and R6 to prevent temporary routing loops that may occur due to mutual redistribution between EIGRP and MP-BGP.
- Ensure the path across the MPLS VPN core is used as the primary path between SW1 and SW2.

14.11 PE-CE Routing with BGP

- Erase all EIGRP configurations for VPN_A on R5, R6, SW1 and SW2.
- Configure BGP AS 78 on SW1 and SW2 and peer the CE routers with R5 and R6.
- Advertise the Loopback 0 interfaces of SW1 and SW2 into BGP and ensure end-to-end reachability.

14.12 BGP SoO Attribute

- Configure a "backdoor" BGP peering session between SW1 and SW2 using the inter-switch link configured previously.
- Ensure this configuration does not result in routing loops due to BGP's loop-prevention mechanism being disabled by the AS-Override feature.

Copyright © 2010 Internetwork Expert

14.13 Internet Access

- Enable RIP on R6's connection to BB1 and configure R6 so that VPN_A customers may access these routes.
- You are allowed to use one static route to accomplish this task.
- Only the 150.X.0.0/16 subnets should be allowed to access the Internet.

14.14 AToM

- Create a P2P L2VPN connection between the VLAN 5 interface of R5 and the unused interface of R6.
- Use the method that provides the minimum amount of transport overhead.

14.15 L2TPV3

- Modify the solution for the previous task to use L2TPv3 instead of MPLS as the underlying transport.
- Ensure that the packets are never fragmented in the core and automatic MTU detection is in progress.

14.16 MPLS VPN Performance Tuning

- Configure the PE and P routers to minimize the amount of time needed to propagate topology changes between CE-sites.

Copyright © 2010 Internetwork Expert

Bridging & Switching Solutions

1.1 Layer 2 Access Switchports

- Using the diagram for reference configure access VLAN assignments on SW1, SW2, SW3, and SW4 to obtain basic connectivity between the devices with Ethernet segments with the exception of R6.
- Do not use VTP to accomplish this.

Configuration

```
SW1:
vlan 7,58,67,79,146
!
interface FastEthernet0/1
 switchport access vlan 146
!
interface FastEthernet0/5
 switchport access vlan 58

SW2:
vlan 8,22,43,58
!
interface FastEthernet0/2
 switchport access vlan 22
!
interface FastEthernet0/4
 switchport access vlan 43
!
interface FastEthernet0/24
 switchport access vlan 22

SW3:
vlan 5,9,43,79
!
interface FastEthernet0/5
 switchport access vlan 5
!
interface FastEthernet0/24
 switchport access vlan 43

SW4:
vlan 10,146
!
interface FastEthernet0/4
 switchport access vlan 146
```

Copyright © 2010 Internetwork Expert

Verification

Note

For hosts connected to different physical switches but in the same VLAN, such as R1 and R4, to get IP connectivity to each other Spanning-Tree Protocol must be forwarding end-to-end between the hosts. An STP instance is automatically created on the Catalyst 3550 and 3560 platforms for a VLAN when the VLAN is created, which implies that the switches in the transit path for the VLAN need to know about it in their VLAN databases.

In most designs this is accomplished through VTP, but in this design it is accomplished simply by issuing the **vlan** command on all switches that need to know about it. Since trunking is preconfigured between all switches in the initial configurations, end-to-end transport is achieved.

Note that in this solution the VLANs created on the switches are not identical. Instead only the minimum number of necessary VLANs are created. The same connectivity result can be achieved by simply configuring the command **vlan 5,7,8,9,10,22,43,58,67,79,146** on all devices. The functional difference is that SW4 for example, who does not need VLAN 5, does not have an STP instance created for VLAN 5. In many production designs these considerations must be taken into account as all platforms have a maximum limitation of the amount of VLANs and STP instances they can support.

In either case for this example however, the final verification is to ensure that the VLANs are assigned correctly, per the **show interface status** or **show vlan** output, and that end-to-end connectivity exists.

```
Rack1SW1#ping 155.1.79.9

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.79.9, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/9 ms

Rack1SW1#ping 155.1.37.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.37.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/4/9 ms
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#ping 155.1.58.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.58.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/8 ms

Rack1R1#ping 155.1.146.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms

Rack1R2#ping 192.10.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.10.1.254, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/5/8 ms

Rack1R4#ping 204.12.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 204.12.1.254, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/5/8 ms

Rack1SW1#show interface status

Port      Name               Status       Vlan       Duplex  Speed Type
Fa0/1                        connected    146        a-full  a-100 10/100BaseTX
Fa0/2                        notconnect   1            auto   auto 10/100BaseTX
Fa0/3                        connected    routed     a-half  a-10 10/100BaseTX
Fa0/4                        notconnect   1            auto   auto 10/100BaseTX
Fa0/5                        connected    58         a-half  a-10 10/100BaseTX
Fa0/6                        notconnect   1            auto   auto 10/100BaseTX
Fa0/7                        notconnect   1            auto   auto 10/100BaseTX
Fa0/8                        notconnect   1            auto   auto 10/100BaseTX
Fa0/9                        notconnect   1            auto   auto 10/100BaseTX
Fa0/10                       notconnect   1            auto   auto 10/100BaseTX
Fa0/11                       notconnect   1            auto   auto 10/100BaseTX
Fa0/12                       notconnect   1            auto   auto 10/100BaseTX
Fa0/13                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/14                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/15                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/16                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/17                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/18                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/19                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/20                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/21                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/22                       notconnect   1            auto   auto 10/100BaseTX
Fa0/23                       notconnect   1            auto   auto 10/100BaseTX
Fa0/24                       notconnect   1            auto   auto 10/100BaseTX
Gi0/1                        notconnect   1            auto   auto Not Present
Gi0/2                        notconnect   1            auto   auto Not Present
```

Copyright © 2010 Internetwork Expert

Rack1SW2#show interface status

Port	Name	Status	Vlan	Duplex	Speed	Type
Fa0/1		notconnect	1	auto	auto	10/100BaseTX
Fa0/2		connected	22	a-full	a-100	10/100BaseTX
Fa0/3		notconnect	1	auto	auto	10/100BaseTX
Fa0/4		connected	43	a-half	a-10	10/100BaseTX
Fa0/5		notconnect	1	auto	auto	10/100BaseTX
Fa0/6		notconnect	1	auto	auto	10/100BaseTX
Fa0/7		notconnect	1	auto	auto	10/100BaseTX
Fa0/8		notconnect	1	auto	auto	10/100BaseTX
Fa0/9		notconnect	1	auto	auto	10/100BaseTX
Fa0/10		notconnect	1	auto	auto	10/100BaseTX
Fa0/11		notconnect	1	auto	auto	10/100BaseTX
Fa0/12		notconnect	1	auto	auto	10/100BaseTX
Fa0/13		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/14		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/15		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/16		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/17		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/18		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/19		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/20		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/21		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/22		notconnect	1	auto	auto	10/100BaseTX
Fa0/23		notconnect	1	auto	auto	10/100BaseTX
Fa0/24		connected	22	a-half	a-10	10/100BaseTX
Gi0/1		notconnect	1	auto	auto	Not Present
Gi0/2		notconnect	1	auto	auto	Not Present

Rack1SW3#show interface status

Port	Name	Status	Vlan	Duplex	Speed	Type
Fa0/1		notconnect	1	auto	auto	10/100BaseTX
Fa0/2		notconnect	1	auto	auto	10/100BaseTX
Fa0/3		connected	1	a-half	a-10	10/100BaseTX
Fa0/4		notconnect	1	auto	auto	10/100BaseTX
Fa0/5		connected	5	a-half	a-10	10/100BaseTX
Fa0/6		notconnect	1	auto	auto	10/100BaseTX
Fa0/7		notconnect	1	auto	auto	10/100BaseTX
Fa0/8		notconnect	1	auto	auto	10/100BaseTX
Fa0/9		notconnect	1	auto	auto	10/100BaseTX
Fa0/10		notconnect	1	auto	auto	10/100BaseTX
Fa0/11		notconnect	1	auto	auto	10/100BaseTX
Fa0/12		notconnect	1	auto	auto	10/100BaseTX
Fa0/13		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/14		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/15		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/16		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/17		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/18		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/19		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/20		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/21		connected	trunk	a-full	a-100	10/100BaseTX
Fa0/22		notconnect	1	auto	auto	10/100BaseTX
Fa0/23		notconnect	1	auto	auto	10/100BaseTX
Fa0/24		connected	43	a-half	a-10	10/100BaseTX
Gi0/1		notconnect	1	auto	auto	Not Present
Gi0/2		notconnect	1	auto	auto	Not Present

Copyright © 2010 Internetwork Expert

```
SW4#show interface status

Port      Name               Status       Vlan       Duplex  Speed Type
Fa0/1                        notconnect   1            auto   auto 10/100BaseTX
Fa0/2                        notconnect   1            auto   auto 10/100BaseTX
Fa0/3                        notconnect   1            auto   auto 10/100BaseTX
Fa0/4                        connected    146        a-half   a-10 10/100BaseTX
Fa0/5                        notconnect   1            auto   auto 10/100BaseTX
Fa0/6                        notconnect   1            auto   auto 10/100BaseTX
Fa0/7                        notconnect   1            auto   auto 10/100BaseTX
Fa0/8                        notconnect   1            auto   auto 10/100BaseTX
Fa0/9                        notconnect   1            auto   auto 10/100BaseTX
Fa0/10                       notconnect   1            auto   auto 10/100BaseTX
Fa0/11                       notconnect   1            auto   auto 10/100BaseTX
Fa0/12                       notconnect   1            auto   auto 10/100BaseTX
Fa0/13                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/14                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/15                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/16                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/17                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/18                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/19                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/20                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/21                       connected    trunk      a-full  a-100 10/100BaseTX
Fa0/22                       notconnect   1            auto   auto 10/100BaseTX
Fa0/23                       notconnect   1            auto   auto 10/100BaseTX
Fa0/24                       notconnect   1            auto   auto 10/100BaseTX
Gi0/1                        notconnect   1            auto   auto unknown
Gi0/2                        notconnect   1            auto   auto unknown
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.2 Layer 2 Dynamic Switchports

- Configure all inter-switch links on SW2, SW3, and SW4 to be in dynamic auto state.
- Configure all inter-switch links on SW1 to be in dynamic desirable state.
- Using the CAM table verify that all layer 2 traffic between devices in the same VLAN, but not attached to the same switch, is transiting SW1.

Configuration

```
SW1:
interface range FastEthernet0/13 - 21
 switchport mode dynamic desirable

SW2:
interface range FastEthernet0/13 - 21
 switchport mode dynamic auto

SW3:
interface range FastEthernet0/13 - 21
 switchport mode dynamic auto

SW4:
interface range FastEthernet0/13 - 21
 switchport mode dynamic auto
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> The remainder of this lab verification is to be performed after R6's router-on-a-stick configuration is completed and verified in section 1.7.

```
Rack1R4#ping 155.1.146.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms

Rack1R4#show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  155.1.146.4            -   0011.2031.4461  ARPA   FastEthernet0/1
Internet  155.1.146.6            0   000f.24da.2220  ARPA   FastEthernet0/1
```

With SW1's inter-switch links in dynamic desirable state, and all other switches inter-switch links in dynamic auto state, trunks will only be formed from SW1 to SW2, SW1 to SW3, and SW1 to SW4. This is because SW1 initiates trunking negotiation through DTP (desirable), and SW2, SW3, and SW4 only respond to DTP negotiation requests (auto). The result of this is indirectly verified by correlating the MAC addresses of R4 and R6 to the CAM table.

R4's port Fa0/1 is connected to SW4's port Fa0/4.

```
Rack1SW4#show mac address-table dynamic address 0011.2031.4461
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
 146    0011.2031.4461    DYNAMIC     Fa0/4
Total Mac Addresses for this criterion: 1
```

R6's port Fa0/0 is connected to SW2's port Fa0/6.

```
Rack1SW2#show mac address-table dynamic address 000f.24da.2220
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    000f.24da.2220    DYNAMIC     Fa0/6
 146    000f.24da.2220    DYNAMIC     Fa0/6
Total Mac Addresses for this criterion: 2
```

Copyright © 2010 Internetwork Expert

If SW2 and SW4 were trunking directly, traffic would forward between their connected ports for VLAN 146. Instead SW2 sees R4's MAC address reachable via port Fa0/13 to SW1, and SW4 sees R6's MAC address reachable via port Fa0/13 to SW1. The CAM table, which is built from the result of STP forwarding and blocking, is the final layer 2 verification of how traffic is actually forwarded through the switched network.

```
Rack1SW2#show mac address-table dynamic address 0011.2031.4461
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
 146    0011.2031.4461    DYNAMIC     Fa0/13
Total Mac Addresses for this criterion: 1

Rack1SW4#show mac address-table dynamic address 000f.24da.2220
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
 146    000f.24da.2220    DYNAMIC     Fa0/13
Total Mac Addresses for this criterion: 1
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.3 ISL Trunking

- Statically set the trunking encapsulation of SW1's inter-switch links to ISL.
- Verify that SW2, SW3, & SW4 are negotiating ISL as the trunking encapsulation to SW1, and that SW1 is not negotiating ISL to SW2, SW3, and SW4.

Configuration

```
SW1:
interface range FastEthernet0/13 - 21
 switchport trunk encapsulation isl
```

Verification

> **Note**
>
> SW1's inter-switch links are running in DTP desirable mode (initiating trunking) with ISL encapsulation statically set. These can be seen under the *Mode* and *Encapsulation* columns from the **`show interface trunk`** output.

```
Rack1SW1#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      desirable    isl            trunking      1
Fa0/14      desirable    isl            trunking      1
Fa0/15      desirable    isl            trunking      1
Fa0/16      desirable    isl            trunking      1
Fa0/17      desirable    isl            trunking      1
Fa0/18      desirable    isl            trunking      1
Fa0/19      desirable    isl            trunking      1
Fa0/20      desirable    isl            trunking      1
Fa0/21      desirable    isl            trunking      1

<output omitted>
```

Copyright © 2010 Internetwork Expert

SW2, SW3, and SW4's inter-switch links are in DTP auto mode, which means they will accept negotiation in from the other side but not initiate it. Since SW1 is statically set to ISL encapsulation, SW2, SW3, and SW4 must agree to this or DTP negotiation will fail. Successful negotiation can be seen in this output since the encapsulation is *n-isl*, for *negotiated* ISL.

```
Rack1SW2#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      auto         n-isl          trunking      1
Fa0/14      auto         n-isl          trunking      1
Fa0/15      auto         n-isl          trunking      1

<output omitted>

Rack1SW3#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      auto         n-isl          trunking      1
Fa0/14      auto         n-isl          trunking      1
Fa0/15      auto         n-isl          trunking      1

<output omitted>

Rack1SW4#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      auto         n-isl          trunking      1
Fa0/14      auto         n-isl          trunking      1
Fa0/15      auto         n-isl          trunking      1

<output omitted>
```

Copyright © 2010 Internetwork Expert

1.4 802.1q Trunking

- Change the trunking encapsulation on SW1's inter-switch links from static ISL to static 802.1q.
- Verify that SW2, SW3, & SW4 are negotiating 802.1q as the trunking encapsulation to SW1, and that SW1 is not negotiating 802.1q to SW2, SW3, and SW4.

Configuration

```
SW1:
interface range FastEthernet0/13 - 21
 switchport trunk encapsulation dot1q
```

Verification

> **Note**
>
> Similar to the previous case, SW1 is running in DTP desirable mode, but now has its trunking encapsulation statically set to 802.1q.

```
Rack1SW1#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      desirable    802.1q         trunking      1
Fa0/14      desirable    802.1q         trunking      1
Fa0/15      desirable    802.1q         trunking      1
Fa0/16      desirable    802.1q         trunking      1
Fa0/17      desirable    802.1q         trunking      1
Fa0/18      desirable    802.1q         trunking      1
Fa0/19      desirable    802.1q         trunking      1
Fa0/20      desirable    802.1q         trunking      1
Fa0/21      desirable    802.1q         trunking      1
<output omitted>
```

Copyright © 2010 Internetwork Expert

SW2, SW3, and SW4 must now agree to using dot1q trunking, as seen in the *n-802.1q* output, for *negotiated* dot1q.

```
Rack1SW2#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      auto         n-802.1q       trunking      1
Fa0/14      auto         n-802.1q       trunking      1
Fa0/15      auto         n-802.1q       trunking      1
<output omitted>

Rack1SW3#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      auto         n-802.1q       trunking      1
Fa0/14      auto         n-802.1q       trunking      1
Fa0/15      auto         n-802.1q       trunking      1
<output omitted>

Rack1SW4#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      auto         n-802.1q       trunking      1
Fa0/14      auto         n-802.1q       trunking      1
Fa0/15      auto         n-802.1q       trunking      1
<output omitted>
```

Copyright © 2010 Internetwork Expert

1.5 802.1q Native VLAN

- Modify the native VLAN on the 802.1q trunks of SW1 so that traffic between devices in VLAN 146 is not tagged when sent over the trunk links.

Configuration

```
SW1:
interface range FastEthernet0/13 - 21
 switchport trunk native vlan 146

SW2:
interface range FastEthernet0/13 - 15
 switchport trunk native vlan 146

SW3:
interface range FastEthernet0/13 - 15
 switchport trunk native vlan 146

SW4:
interface range FastEthernet0/13 - 15
 switchport trunk native vlan 146
```

Verification

Note

The IEEE 802.1q trunking encapsulation standard defines the term *native* VLAN to describe traffic sent and received on an interface running 802.1q encapsulation that does not have an 802.1q tag actually inserted. When the switch sends a frame that belongs to the native VLAN, it is sent the same as if 802.1q was not configured. When the switch receives a frame on an interface running 802.1q that does not have a tag, it assumes it is part of the native VLAN. For this reason the switches on both ends of an 802.1q trunk link must agree on what the native VLAN is, otherwise traffic can unexpectedly leak between broadcast domain boundaries.

The native VLAN defaults to 1 unless modified. In this case the native VLAN is modified to 146 on both ends of the link.

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      desirable    802.1q         trunking      146
Fa0/14      desirable    802.1q         trunking      146
Fa0/15      desirable    802.1q         trunking      146
Fa0/16      desirable    802.1q         trunking      146
Fa0/17      desirable    802.1q         trunking      146
Fa0/18      desirable    802.1q         trunking      146
Fa0/19      desirable    802.1q         trunking      146
Fa0/20      desirable    802.1q         trunking      146
Fa0/21      desirable    802.1q         trunking      146

<output omitted>

Rack1SW2#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      auto         n-802.1q       trunking      146
Fa0/14      auto         n-802.1q       trunking      146
Fa0/15      auto         n-802.1q       trunking      146

<output omitted>

Rack1SW3#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      auto         n-802.1q       trunking      146
Fa0/14      auto         n-802.1q       trunking      146
Fa0/15      auto         n-802.1q       trunking      146

<output omitted>

Rack1SW4#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      auto         n-802.1q       trunking      146
Fa0/14      auto         n-802.1q       trunking      146
Fa0/15      auto         n-802.1q       trunking      146

<output omitted>
```

Copyright © 2010 Internetwork Expert

1.6 Disabling DTP Negotiation

- Disable Dynamic Trunking Protocol on the trunk links of SW1.
- Verify that trunking is still occurring between SW1 & SW2, SW1 & SW3, and SW1 & SW4 without the use of DTP.

Configuration

```
SW1:
interface range FastEthernet0/13 - 21
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate

SW2:
interface range FastEthernet0/13 - 15
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate

SW3:
interface range FastEthernet0/13 - 15
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate

SW4:
interface range FastEthernet0/13 - 15
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
```

Copyright © 2010 Internetwork Expert

Verification

Note

DTP negotiation can be disabled two ways, with the **switchport mode access** command, or with the **switchport nonegotiate** command. If trunking is needed, but DTP is disabled, it must be statically configured with the **switchport mode trunk** command. This design is most commonly used when a switch is trunking to a device that does not support DTP, such as an IOS router's routed Ethernet interface (not an EtherSwitch interface), or a server's NIC card.

```
Rack1SW1#show interface fa0/13 switchport | include Negotiation
Negotiation of Trunking: Off

Rack1SW1#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      on           802.1q         trunking      146
Fa0/14      on           802.1q         trunking      146
Fa0/15      on           802.1q         trunking      146
Fa0/16      on           802.1q         trunking      146
Fa0/17      on           802.1q         trunking      146
Fa0/18      on           802.1q         trunking      146
Fa0/19      on           802.1q         trunking      146
Fa0/20      on           802.1q         trunking      146
Fa0/21      on           802.1q         trunking      146

<output omitted>
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      on           802.1q         trunking      146
Fa0/14      on           802.1q         trunking      146
Fa0/15      on           802.1q         trunking      146

<output omitted>

Rack1SW3#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      on           802.1q         trunking      146
Fa0/14      on           802.1q         trunking      146
Fa0/15      on           802.1q         trunking      146

<output omitted>

Rack1SW4#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      on           802.1q         trunking      146
Fa0/14      on           802.1q         trunking      146
Fa0/15      on           802.1q         trunking      146

<output omitted>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.7 Router-On-A-Stick

- Configure the link between SW2 and R6 as an 802.1q trunk link.
- Using the subinterfaces listed in the diagram configure R6 to route traffic for both VLANs 67 and 146 on its Ethernet link.
- Verify that R6 has reachability to devices both on VLAN 67 and 146.

Configuration

```
SW2:
vlan 67,146
!
interface FastEthernet0/6
 switchport trunk encapsulation dot1q
 switchport mode trunk

R6:
interface FastEthernet0/0.67
 encapsulation dot1q 67
 ip address 155.1.67.6 255.255.255.0
!
interface FastEthernet0/0.146
 encapsulation dot1q 146
 ip address 155.1.146.6 255.255.255.0
!
interface FastEthernet0/0
no shut
```

Verification

Note

Router-on-a-stick is the legacy implementation of inter-VLAN routing, which is typically replaced in most designs now with layer 3 Switch Virtual Interfaces (SVIs) on layer 3 switches. In router-on-a-stick a layer 2 switch trunks multiple VLANs to a router, the router accepts a layer 2 packet in the physical interface, categorizes it based on the VLAN tag, rebuilds the layer 2 frame, and sends the packet back to the switch.

Note that since the router does not support DTP negotiation on its routed Ethernet interface, the attached switch must issue the **`switchport mode trunk`** command. The **`switchport nonegotiate`** command, while recommended, is not required on the switch. Also to minimize the amount of broadcast traffic that the router receives the switch should ideally edit the allowed list of the trunk going to the router to only allow the VLANs that the router is encapsulating. This is generally necessary since the router does not support VTP pruning on its routed trunk interface.

Copyright © 2010 Internetwork Expert

```
Rack1R6#ping 155.1.67.7

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.67.7, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/5 ms

Rack1R6#ping 155.1.146.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1SW2#show interface fa0/6 trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/6       on           802.1q         trunking      1

Port        Vlans allowed on trunk
Fa0/6       1-4094

Port        Vlans allowed and active in management domain
Fa0/6       1,8,22,43,58,67,146

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/6       1,8,22,43,58,67,146
```

Copyright © 2010 Internetwork Expert

1.8 VTP

- Configure all inter-switch links on SW2, SW3, and SW4 to be in dynamic auto state.
- Configure all inter-switch links on SW1 to be in dynamic desirable state.
- Configure SW2 as a VTP server in the domain CCIE.
- Configure SW1, SW3, and SW4 as VTP clients in the domain CCIE.
- Configure necessary VLAN definitions on SW2 using the diagram for reference.
- Configure access VLAN assignments on SW1, SW2, SW3, and SW4 to obtain basic connectivity between the devices with Ethernet segments.
- Configure router-on-a-stick between SW2 and R6 per the diagram so R6 has reachability to devices on VLANs 67 and 146.

Configuration

```
R6:
interface FastEthernet0/0.67
 encapsulation dot1q 67
 ip address 155.1.67.6 255.255.255.0
!
interface FastEthernet0/0.146
 encapsulation dot1q 146
 ip address 155.1.146.6 255.255.255.0

SW1:
vtp domain CCIE
vtp mode client
!
interface range FastEthernet0/13 - 21
 switchport mode dynamic desirable
!
interface FastEthernet0/1
 switchport access vlan 146
!
interface FastEthernet0/5
 switchport access vlan 58

SW2:
vtp domain CCIE
vlan 5,7,8,9,10,22,43,58,67,79,146
!
interface FastEthernet0/2
 switchport access vlan 22
!
interface FastEthernet0/4
 switchport access vlan 43
!
interface FastEthernet0/6
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
```

Copyright © 2010 Internetwork Expert

```
interface FastEthernet0/24
 switchport access vlan 22
!
interface range FastEthernet0/13 - 21
 switchport mode dynamic auto

SW3:
vtp domain CCIE
vtp mode client
!
interface FastEthernet0/5
 switchport access vlan 5
!
interface FastEthernet0/24
 switchport access vlan 43
!
interface range FastEthernet0/13 - 21
 switchport mode dynamic auto

SW4:
vtp domain CCIE
vtp mode client
!
interface FastEthernet0/4
 switchport access vlan 146
!
interface range FastEthernet0/13 - 21
 switchport mode dynamic auto
```

Copyright © 2010 Internetwork Expert

Verification

Note

VLAN Trunking Protocol (VTP) can be used in the Ethernet domain to simplify the creation and management of VLANs, however it does not dictate the traffic flow of VLANs or the actual assignments. The first step in running VTP is to ensure that the switches are trunking with each other. Next, the VTP domain name is configured, and all other switches without domain names configured inherit this. Lastly on the VTP server the VLAN definitions are created.

To verify this configuration compare the output of the **`show vtp status`** command on all devices in the domain. If the domain name, the number of existing VLANs, and the Configuration Revision Number all match, the domain is converged. If authentication is configured the MD5 digest field should be compared as well.

```
Rack1SW1#show vtp status
VTP Version                     : running VTP1 (VTP2 capable)
Configuration Revision          : 1
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 16
VTP Operating Mode              : Client
VTP Domain Name                 : CCIE
VTP Pruning Mode                : Disabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x7C 0x80 0x15 0x50 0xA2 0x06 0x41
0x6A
Configuration last modified by 150.1.10.10 at 5-20-08 07:55:18

Rack1SW2#show vtp status
VTP Version                     : running VTP1 (VTP2 capable)
Configuration Revision          : 1
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 16
VTP Operating Mode              : Server
VTP Domain Name                 : CCIE
VTP Pruning Mode                : Disabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x7C 0x80 0x15 0x50 0xA2 0x06 0x41
0x6A
Configuration last modified by 150.1.10.10 at 5-20-08 07:55:18
```

Copyright © 2010 Internetwork Expert

```
Rack1SW3#show vtp status
VTP Version                     : running VTP1 (VTP2 capable)
Configuration Revision          : 1
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 16
VTP Operating Mode              : Client
VTP Domain Name                 : CCIE
VTP Pruning Mode                : Disabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x7C 0x80 0x15 0x50 0xA2 0x06 0x41
0x6A
Configuration last modified by 150.1.10.10 at 5-20-08 07:55:18

Rack1SW4#show vtp status
VTP Version                     : running VTP1 (VTP2 capable)
Configuration Revision          : 1
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 16
VTP Operating Mode              : Client
VTP Domain Name                 : CCIE
VTP Pruning Mode                : Disabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x7C 0x80 0x15 0x50 0xA2 0x06 0x41
0x6A
Configuration last modified by 150.1.10.10 at 5-20-08 07:55:18
Local updater ID is 155.1.10.10 on interface Vl10 (lowest numbered VLAN
interface found)
```

Copyright © 2010 Internetwork Expert

show vlan or **show vlan brief** can also be compared to ensure that the VLAN numbers and names properly propagated throughout the VTP domain.

```
Rack1SW1#show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/2, Fa0/4, Fa0/6, Fa0/7
                                                Fa0/8, Fa0/9, Fa0/10, Fa0/11
                                                Fa0/12, Fa0/22, Fa0/23, Fa0/24
                                                Gi0/1, Gi0/2
5    VLAN0005                         active
7    VLAN0007                         active
8    VLAN0008                         active
9    VLAN0009                         active
10   VLAN0010                         active
22   VLAN0022                         active
43   VLAN0043                         active
58   VLAN0058                         active    Fa0/5
67   VLAN0067                         active
79   VLAN0079                         active
146  VLAN0146                         active    Fa0/1
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup

Rack1SW2#show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/1, Fa0/3, Fa0/5, Fa0/7
                                                Fa0/8, Fa0/9, Fa0/10, Fa0/11
                                                Fa0/12, Fa0/16, Fa0/17, Fa0/18
                                                Fa0/19, Fa0/20, Fa0/21, Fa0/22
                                                Fa0/23, Gi0/1, Gi0/2
5    VLAN0005                         active
7    VLAN0007                         active
8    VLAN0008                         active
9    VLAN0009                         active
10   VLAN0010                         active
22   VLAN0022                         active    Fa0/2, Fa0/24
43   VLAN0043                         active    Fa0/4
58   VLAN0058                         active
67   VLAN0067                         active
79   VLAN0079                         active
146  VLAN0146                         active
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup
```

Copyright © 2010 Internetwork Expert

```
Rack1SW3#show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/6, Fa0/7, Fa0/8, Fa0/9
                                                Fa0/10, Fa0/11, Fa0/12, Fa0/16
                                                Fa0/17, Fa0/18, Fa0/19, Fa0/20
                                                Fa0/21, Fa0/22, Fa0/23, Gi0/1
                                                Gi0/2
5    VLAN0005                         active    Fa0/5
7    VLAN0007                         active
8    VLAN0008                         active
9    VLAN0009                         active
10   VLAN0010                         active
22   VLAN0022                         active
43   VLAN0043                         active    Fa0/24
58   VLAN0058                         active
67   VLAN0067                         active
79   VLAN0079                         active
146  VLAN0146                         active
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup

Rack1SW4#show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/5
                                                Fa0/6, Fa0/7, Fa0/8, Fa0/9
                                                Fa0/10, Fa0/11, Fa0/12, Fa0/16
                                                Fa0/17, Fa0/18, Fa0/19, Fa0/20
                                                Fa0/21, Fa0/22, Fa0/23, Fa0/24
                                                Gi0/1, Gi0/2
5    VLAN0005                         active
7    VLAN0007                         active
8    VLAN0008                         active
9    VLAN0009                         active
10   VLAN0010                         active
22   VLAN0022                         active
43   VLAN0043                         active
58   VLAN0058                         active
67   VLAN0067                         active
79   VLAN0079                         active
146  VLAN0146                         active    Fa0/4
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup
```

Copyright © 2010 Internetwork Expert

1.9 VTP Transparent

- Configure SW1 in VTP transparent mode and remove all previous VLAN definitions from it.
- Configure SW1 with only the VLAN definitions necessary to obtain basic connectivity between the devices with Ethernet segments.

Configuration

```
SW1:
vtp mode transparent
no vlan 2-1000
vlan 7,43,58,67,79,146
```

Verification

> **Note**
>
> VTP devices running in transparent mode do not install VTP updates received, but will continue to forward them on unmodified if the domain name matches its locally configured domain. The configuration revision number of zero indicates that it is not participating in the update sequence of the rest of the domain.

```
Rack1SW1#show vtp status
VTP Version                     : running VTP1 (VTP2 capable)
Configuration Revision          : 0
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 11
VTP Operating Mode              : Transparent
VTP Domain Name                 : CCIE
VTP Pruning Mode                : Disabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x4D 0xD1 0x7E 0x5F 0xE4 0x00 0xB6 0x86
Configuration last modified by 155.1.37.7 at 5-20-08 07:55:18
```

Copyright © 2010 Internetwork Expert

Since VTP does not directly relate to STP forwarding, traffic from the server/client or from an entirely different VTP domain can be in the same broadcast domain as the transparent switches ports as long as STP is forwarding end to end. In this particular case SW1 does not have VLANs 7, 43, 67, or 79 locally assigned, but it is in the physical layer 2 transit path for these. This implies that these VLANs must be created, otherwise traffic will be received inbound but not forwarded outbound as there will be no STP instance associated with the VLAN.

```
Rack1SW1#show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/2, Fa0/4, Fa0/6, Fa0/7
                                                Fa0/8, Fa0/9, Fa0/10, Fa0/11
                                                Fa0/12, Fa0/22, Fa0/23, Fa0/24
                                                Gi0/1, Gi0/2
7    VLAN0007                         active
43   VLAN0043                         active
58   VLAN0058                         active    Fa0/5
67   VLAN0067                         active
79   VLAN0079                         active
146  VLAN0146                         active    Fa0/1
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup
```

Changes in the rest of the VTP domain, such as VLAN adds or removes, do not affect the transparent switches.

```
Rack1SW2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW2(config)#vlan 123
Rack1SW2(config-vlan)#end
Rack1SW2#

Rack1SW2#show vlan | include ^123
123  VLAN0123                         active
123  enet  100123     1500  -      -      -        -    -        0      0

Rack1SW3#show vlan | include ^123
123  VLAN0123                         active
123  enet  100123     1500  -      -      -        -    -        0      0

Rack1SW1#show vlan | include ^123
```

1.10 VTP Pruning

- Configure SW1 in VTP client mode.
- Enable VTP pruning in the layer 2 network so that inter-switch broadcast replication is minimized.
- Verify this configuration is functional through the **`show interface trunk`** output.

Configuration

```
SW1:
vtp mode client

SW2:
vtp pruning
```

Verification

> **Note**
>
> VTP pruning eliminates the need to statically remove VLANs from the allowed trunking list of a port by having the switches automatically communicate to each other which VLANs they have locally assigned or are in the transit path for.
>
> The **`show interface pruning`** command indicates what traffic the local switch told its neighbor that it needs, via the *VLAN traffic requested of neighbor* field. These VLANs are either ones locally assigned or those that the local switch is in the layer 2 transit path for. The *Vlans pruned for lack of request by neighbor* field indicates the VLANs that the upstream neighbor did *not* request.
>
> In the below output this means that SW1 is not forwarding VLAN 7 to SW3, because SW3 did not request it. This output can be confusing because what SW1 sees as pruned for lack of request is the opposite of what SW3 sees as requested.

```
Rack1SW1#show interface fa0/16 pruning

Port        Vlans pruned for lack of request by neighbor
Fa0/16      7-8,10,22,58,67,146

Port        Vlan traffic requested of neighbor
Fa0/16      1,5,7-10,22,43,58,67,79,146
```

Copyright © 2010 Internetwork Expert

```
Rack1SW3#show interface fa0/13 pruning

Port        Vlans pruned for lack of request by neighbor
Fa0/13      none

Port        Vlan traffic requested of neighbor
Fa0/13      1,5,9,43,79
```

If the network is converged all devices in the VTP domain should agree that pruning is enabled, as seen in the below **show vtp status** output. Note that transparent switches cannot participate in pruning because they do not read the payload of the VTP updates they are receiving from their adjacent neighbors.

```
Rack1SW1#show vtp status
VTP Version                     : running VTP1 (VTP2 capable)
Configuration Revision          : 6
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 16
VTP Operating Mode              : Client
VTP Domain Name                 : CCIE
VTP Pruning Mode                : Enabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x4F 0x03 0x83 0x1F 0x24 0xE1 0x01
0x45
Configuration last modified by 155.1.8.8 at 5-20-08 08:27:49

Rack1SW2#show vtp status
VTP Version                     : running VTP1 (VTP2 capable)
Configuration Revision          : 6
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 16
VTP Operating Mode              : Server
VTP Domain Name                 : CCIE
VTP Pruning Mode                : Enabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x4F 0x03 0x83 0x1F 0x24 0xE1 0x01
0x45
Configuration last modified by 155.1.8.8 at 5-20-08 08:27:49
Local updater ID is 155.1.8.8 on interface Vl8 (lowest numbered VLAN
interface found)
```

Copyright © 2010 Internetwork Expert

```
Rack1SW3#show vtp status
VTP Version                     : running VTP1 (VTP2 capable)
Configuration Revision          : 6
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 16
VTP Operating Mode              : Client
VTP Domain Name                 : CCIE
VTP Pruning Mode                : Enabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x4F 0x03 0x83 0x1F 0x24 0xE1 0x01
0x45
Configuration last modified by 155.1.8.8 at 5-20-08 08:27:49

Rack1SW4#show vtp status
VTP Version                     : running VTP1 (VTP2 capable)
Configuration Revision          : 6
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 16
VTP Operating Mode              : Server
VTP Domain Name                 : CCIE
VTP Pruning Mode                : Enabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x4F 0x03 0x83 0x1F 0x24 0xE1 0x01
0x45
Configuration last modified by 155.1.8.8 at 5-20-08 08:27:49
Local updater ID is 155.1.10.10 on interface Vl10 (lowest numbered VLAN
interface found)
```

To quickly view what traffic is not being pruned, and hence actually forwarded, issue the show interface trunk command. The final field of Vlans in spanning tree forwarding state and not pruned means that the VLAN is created, is allowed on the link, is running STP, and is not pruned.

```
Rack1SW1#show interface trunk | begin pruned
Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      1,5,7-10,22,43,58,67,79,146
Fa0/14      1
Fa0/15      1
Fa0/16      1,5,9,43,79
Fa0/17      none
Fa0/18      none
Fa0/19      10,146
Fa0/20      none
Fa0/21      none
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show interface trunk | begin pruned
Port        Vlans in spanning tree forwarding state and not pruned
Fa0/6       1,5,7-10,22,43,58,67,79,146
Fa0/13      5,7,9-10,43,58,67,79,146
Fa0/14      none
Fa0/15      none

Rack1SW3#show interface trunk | begin pruned
Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      1,5,7-10,22,43,58,67,79,146
Fa0/14      1
Fa0/15      1

Rack1SW4#show interface trunk | begin pruned
Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      1,5,7-10,22,43,58,67,79,146
Fa0/14      1
Fa0/15      1
```

Copyright © 2010 Internetwork Expert

1.11 VTP Prune-Eligible List

- Edit the prune-eligible list to ensure that traffic for VLAN 7 is carried on all active trunk links in the layer 2 network.
- Verify this configuration is functional through the **`show interface trunk`** output.

Configuration

```
SW1:
interface FastEthernet0/13
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/14
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/15
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/16
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/17
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/18
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/19
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/20
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/21
 switchport trunk pruning vlan 2-6,8-1001

SW2:
interface FastEthernet0/13
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/14
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/15
 switchport trunk pruning vlan 2-6,8-1001
```

Copyright © 2010 Internetwork Expert

```
SW3:
interface FastEthernet0/13
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/14
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/15
 switchport trunk pruning vlan 2-6,8-1001

SW4:
interface FastEthernet0/13
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/14
 switchport trunk pruning vlan 2-6,8-1001
!
interface FastEthernet0/15
 switchport trunk pruning vlan 2-6,8-1001
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> The implementation of the prune eligible list, which is controlled by the **`switchport trunk pruning vlan`** command, is commonly confusing because it is essentially the *opposite* of the editing the allowed list of the trunk. By default all VLANs 2-1001 (not the default or extended VLANs) can be pruned off of a trunk link.
>
> This means that if the switch does not have VLAN 7 assigned, and is not in the transit path for VLAN 7, it can tell its adjacent switches not to send it VLAN 7 traffic. However, if VLAN 7 is removed from the prune eligible list, the switch must report that it *does* need VLAN 7, and the traffic cannot be pruned.
>
> This can be seen in the change of the output below, where SW1 sends VLAN 7 traffic over all links that are forwarding for STP, even though the devices on the other end of the link don't actually need VLAN 7.

```
Rack1SW1#show interface trunk | begin pruned
Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      1,5,7-10,22,43,58,67,79,146
Fa0/14      1,7
Fa0/15      1,7
Fa0/16      1,5,7,9,43,79
Fa0/17      none
Fa0/18      none
Fa0/19      7,10,146
Fa0/20      7
Fa0/21      7

Rack1SW2#show interface trunk | begin pruned
Port        Vlans in spanning tree forwarding state and not pruned
Fa0/6       1,5,7-10,22,43,58,67,79,146
Fa0/13      5,7,9-10,43,58,67,79,146
Fa0/14      none
Fa0/15      none

Rack1SW3#show interface trunk | begin pruned
Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      1,5,7-10,22,43,58,67,79,146
Fa0/14      1,7
Fa0/15      1,7

Rack1SW4#show interface trunk | begin pruned
Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      1,5,7-10,22,43,58,67,79,146
Fa0/14      1
Fa0/15      1
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.12 Layer 2 EtherChannel

- Remove all previous configurations on the links connecting SW1, SW2, SW3, and SW4.
- Configure all inter-switch links on SW2, SW3, and SW4 to be in dynamic auto state.
- Configure all inter-switch links on SW1 to be in dynamic desirable state.
- Configure Layer 2 EtherChannels on all inter-switch links between SW1 & SW2, SW1 & SW3, and SW1 & SW4.
- Use Port-Channel numbers 12, 13, and 14 respectively.
- These links should not use dynamic EtherChannel negotiation.

Configuration

```
SW1:
interface FastEthernet0/13
 switchport mode dynamic desirable
 channel-group 12 mode on
!
interface FastEthernet0/14
 switchport mode dynamic desirable
 channel-group 12 mode on
!
interface FastEthernet0/15
 switchport mode dynamic desirable
 channel-group 12 mode on
!
interface FastEthernet0/16
 switchport mode dynamic desirable
 channel-group 13 mode on
!
interface FastEthernet0/17
 switchport mode dynamic desirable
 channel-group 13 mode on
!
interface FastEthernet0/18
 switchport mode dynamic desirable
 channel-group 13 mode on
!
interface FastEthernet0/19
 switchport mode dynamic desirable
 channel-group 14 mode on
!
interface FastEthernet0/20
 switchport mode dynamic desirable
 channel-group 14 mode on
!
interface FastEthernet0/21
 switchport mode dynamic desirable
 channel-group 14 mode on
```

Copyright © 2010 Internetwork Expert

```
SW2:
interface FastEthernet0/13
 channel-group 12 mode on
!
interface FastEthernet0/14
 channel-group 12 mode on
!
interface FastEthernet0/15
 channel-group 12 mode on

SW3:
interface FastEthernet0/13
 channel-group 13 mode on
!
interface FastEthernet0/14
 channel-group 13 mode on
!
interface FastEthernet0/15
 channel-group 13 mode on

SW4:
interface FastEthernet0/13
 channel-group 14 mode on
!
interface FastEthernet0/14
 channel-group 14 mode on
!
interface FastEthernet0/15
 channel-group 14 mode on
```

Copyright © 2010 Internetwork Expert

Verification

Note

For an EtherChannel to form all member interfaces must agree on the same configuration, and both ends of the channel must agree on the same negotiation protocol. In the below **`show etherchannel summary`** output the *Protocol* field is null, which means that no negotiation was used. This comes from the *on* mode of the **`channel-group`** command. This output also shows that the *Port-channel* is in the (SU) state, which means layer 2 switchport that is up, and the members *Ports* are in the (P) state, which is in the port-channel.

```
Rack1SW1#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 3
Number of aggregators:           3

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
12     Po12(SU)         -        Fa0/13(P)   Fa0/14(P)   Fa0/15(P)
13     Po13(SU)         -        Fa0/16(P)   Fa0/17(P)   Fa0/18(P)
14     Po14(SU)         -        Fa0/19(P)   Fa0/20(P)   Fa0/21(P)
```

Copyright © 2010 Internetwork Expert

Since SW1's member interfaces were dynamic desirable switchports, the Port-Channel interfaces that are spawned from them inherit these attributes. This means that the channel interfaces on SW1 will initiate negotiation, and the other channels on SW2, SW3, and SW4 should respond.

```
Rack1SW1#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Po12        desirable    n-isl          trunking      1
Po13        desirable    n-isl          trunking      1
Po14        desirable    n-isl          trunking      1

Port        Vlans allowed on trunk
Po12        1-4094
Po13        1-4094
Po14        1-4094

Port        Vlans allowed and active in management domain
Po12        1,5,7-10,22,43,58,67,79,146
Po13        1,5,7-10,22,43,58,67,79,146
Po14        1,5,7-10,22,43,58,67,79,146

Port        Vlans in spanning tree forwarding state and not pruned
Po12        1,5,7-10,22,43,58,67,79,146
Po13        1,5,9,43,79
Po14        1,10,146
```

Copyright © 2010 Internetwork Expert

An additional way to verify that a layer 2 channel is working correctly is to view the spanning-tree topology. If STP sees the single port-channel interface running one instance of STP, channeling has occurred properly. This is due to the fact that without channeling some member interfaces would be in the STP forwarding state, and some blocking, but with channeling they are all forwarding.

```
Rack1SW1#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    32778
              Address     000c.3045.4180
              Cost        9
              Port        168 (Port-channel13)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     001b.d490.7c00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Po12             Desg FWD 9         128.160  P2p
Po13             Root FWD 9         128.168  P2p
Po14             Desg FWD 9         128.176  P2p

Rack1SW2#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
12     Po12(SU)         -        Fa0/13(P)   Fa0/14(P)   Fa0/15(P)
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/6       on           802.1q         trunking      1
Po12        auto         n-isl          trunking      1

Port        Vlans allowed on trunk
Fa0/6       1-4094
Po12        1-4094

Port        Vlans allowed and active in management domain
Fa0/6       1,5,7-10,22,43,58,67,79,146
Po12        1,5,7-10,22,43,58,67,79,146

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/6       1,5,7-10,22,43,58,67,79,146
Po12        1,5,7,9-10,43,58,67,79,146

Rack1SW2#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    32778
              Address     000c.3045.4180
              Cost        18
              Port        160 (Port-channel12)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     001b.d4df.ec80
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/6            Desg FWD 19        128.8    P2p
Po12             Root FWD 9         128.160  P2p
```

Copyright © 2010 Internetwork Expert

```
Rack1SW3#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
13     Po13(SU)         -        Fa0/13(P)   Fa0/14(P)   Fa0/15(P)

Rack1SW3#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Po13        auto         n-isl          trunking      1

Port        Vlans allowed on trunk
Po13        1-4094

Port        Vlans allowed and active in management domain
Po13        1,5,7-10,22,43,58,67,79,146

Port        Vlans in spanning tree forwarding state and not pruned
Po13        1,5,7-10,22,43,58,67,79,146

Rack1SW3#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID    Priority    32778
             Address     000c.3045.4180
             This bridge is the root
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32778  (priority 32768 sys-id-ext 10)
             Address     000c.3045.4180
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Po13             Desg FWD 9         128.66   P2p
```

```
Rack1SW4#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
14     Po14(SU)         -        Fa0/13(P)   Fa0/14(P)   Fa0/15(P)

Rack1SW4#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Po14        auto         n-isl          trunking      1

Port        Vlans allowed on trunk
Po14        1-4094

Port        Vlans allowed and active in management domain
Po14        1,5,7-10,22,43,58,67,79,146

Port        Vlans in spanning tree forwarding state and not pruned
Po14        1,5,7-10,22,43,58,67,79,146

Rack1SW4#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    32778
              Address     000c.3045.4180
              Cost        18
              Port        65 (Port-channel14)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     000c.3045.d600
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Po14             Root FWD 9         128.65   P2p
```

Copyright © 2010 Internetwork Expert

1.13 Layer 2 EtherChannel with PAgP

- Modify the previous EtherChannel configuration to use PAgP for dynamic negotiation.
- SW1 should initiate negotiation and the other devices should respond.

Configuration

```
SW1:
interface FastEthernet0/13
 switchport mode dynamic desirable
 channel-group 12 mode desirable
!
interface FastEthernet0/14
 switchport mode dynamic desirable
 channel-group 12 mode desirable
!
interface FastEthernet0/15
 switchport mode dynamic desirable
 channel-group 12 mode desirable
!
interface FastEthernet0/16
 switchport mode dynamic desirable
 channel-group 13 mode desirable
!
interface FastEthernet0/17
 switchport mode dynamic desirable
 channel-group 13 mode desirable
!
interface FastEthernet0/18
 switchport mode dynamic desirable
 channel-group 13 mode desirable
!
interface FastEthernet0/19
 switchport mode dynamic desirable
 channel-group 14 mode desirable
!
interface FastEthernet0/20
 switchport mode dynamic desirable
 channel-group 14 mode desirable
!
interface FastEthernet0/21
 switchport mode dynamic desirable
 channel-group 14 mode desirable

SW2:
interface FastEthernet0/13
 channel-group 12 mode auto
!
interface FastEthernet0/14
 channel-group 12 mode auto
!
interface FastEthernet0/15
 channel-group 12 mode auto
```

Copyright © 2010 Internetwork Expert

```
SW3:
interface FastEthernet0/13
 channel-group 13 mode auto
!
interface FastEthernet0/14
 channel-group 13 mode auto
!
interface FastEthernet0/15
 channel-group 13 mode auto

SW4:
interface FastEthernet0/13
 channel-group 14 mode auto
!
interface FastEthernet0/14
 channel-group 14 mode auto
!
interface FastEthernet0/15
 channel-group 14 mode auto
```

Verification

> **Note**
>
> Port Aggregation Protocol (PAgP) is a Cisco proprietary negotiation protocol for EtherChannel links. The desirable mode of PAgP, like DTP, is used to initiate negotiation, while the auto mode is used to listen for negotiation. This implies that one side running desirable with the other side running desirable or auto will result in a channel, but both sides running auto will not.

```
Rack1SW1#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 3
Number of aggregators:           3

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
12     Po12(SU)         PAgP      Fa0/13(P)   Fa0/14(P)   Fa0/15(P)
13     Po13(SU)         PAgP      Fa0/16(P)   Fa0/17(P)   Fa0/18(P)
14     Po14(SU)         PAgP      Fa0/19(P)   Fa0/20(P)   Fa0/21(P)
```

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Po12        desirable    n-isl          trunking      1
Po13        desirable    n-isl          trunking      1
Po14        desirable    n-isl          trunking      1

Port        Vlans allowed on trunk
Po12        1-4094
Po13        1-4094
Po14        1-4094

Port        Vlans allowed and active in management domain
Po12        1,5,7-10,22,43,58,67,79,146
Po13        1,5,7-10,22,43,58,67,79,146
Po14        1,5,7-10,22,43,58,67,79,146

Port        Vlans in spanning tree forwarding state and not pruned
Po12        1,5,7-10,22,43,58,67,79,146
Po13        1,5,9,43,79
Po14        1,10,146

Rack1SW1#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    32778
              Address     000c.3045.4180
              Cost        9
              Port        168 (Port-channel13)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     001b.d490.7c00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Po12             Desg FWD 9         128.160  P2p
Po13             Root FWD 9         128.168  P2p
Po14             Desg FWD 9         128.176  P2p
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
12     Po12(SU)         PAgP      Fa0/13(P)   Fa0/14(P)   Fa0/15(P)

Rack1SW2#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/6       on           802.1q         trunking      1
Po12        auto         n-isl          trunking      1

Port        Vlans allowed on trunk
Fa0/6       1-4094
Po12        1-4094

Port        Vlans allowed and active in management domain
Fa0/6       1,5,7-10,22,43,58,67,79,146
Po12        1,5,7-10,22,43,58,67,79,146

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/6       1,5,7-10,22,43,58,67,79,146
Po12        1,5,7,9-10,43,58,67,79,146


Rack1SW2#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID    Priority    32778
             Address     000c.3045.4180
             Cost        18
             Port        160 (Port-channel12)
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32778  (priority 32768 sys-id-ext 10)
             Address     001b.d4df.ec80
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/6            Desg FWD 19        128.8    P2p
Po12             Root FWD 9         128.160  P2p
```

Copyright © 2010 Internetwork Expert

```
Rack1SW3#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
13     Po13(SU)         PAgP      Fa0/13(P)   Fa0/14(P)   Fa0/15(P)
```

Rack1SW3#show interface trunk

```
Port        Mode         Encapsulation  Status        Native vlan
Po13        desirable    n-isl          trunking      1

Port        Vlans allowed on trunk
Po13        1-4094

Port        Vlans allowed and active in management domain
Po13        1,5,7-10,22,43,58,67,79,146

Port        Vlans in spanning tree forwarding state and not pruned
Po13        1,5,7-10,22,43,58,67,79,146
```

Rack1SW3#show spanning-tree vlan 10

```
VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    32778
              Address     000c.3045.4180
              This bridge is the root
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     000c.3045.4180
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Po13             Desg FWD 9         128.66   P2p
```

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
14     Po14(SU)         PAgP      Fa0/13(P)   Fa0/14(P)   Fa0/15(P)

Rack1SW4#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Po14        desirable    n-isl          trunking      1

Port        Vlans allowed on trunk
Po14        1-4094

Port        Vlans allowed and active in management domain
Po14        1,5,7-10,22,43,58,67,79,146

Port        Vlans in spanning tree forwarding state and not pruned
Po14        1,5,7-10,22,43,58,67,79,146

Rack1SW4#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    32778
              Address     000c.3045.4180
              Cost        18
              Port        65 (Port-channel14)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     000c.3045.d600
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Po14             Root FWD 9         128.65   P2p
```

Copyright © 2010 Internetwork Expert

1.14 Layer 2 EtherChannel with LACP

- Modify the previous EtherChannel configuration to use LACP for dynamic negotiation.
- SW1 should initiate negotiation and the other devices should respond.

Configuration

```
SW1:
interface FastEthernet0/13
 switchport mode dynamic desirable
 channel-group 12 mode active
!
interface FastEthernet0/14
 switchport mode dynamic desirable
 channel-group 12 mode active
!
interface FastEthernet0/15
 switchport mode dynamic desirable
 channel-group 12 mode active
!
interface FastEthernet0/16
 switchport mode dynamic desirable
 channel-group 13 mode active
!
interface FastEthernet0/17
 switchport mode dynamic desirable
 channel-group 13 mode active
!
interface FastEthernet0/18
 switchport mode dynamic desirable
 channel-group 13 mode active
!
interface FastEthernet0/19
 switchport mode dynamic desirable
 channel-group 14 mode active
!
interface FastEthernet0/20
 switchport mode dynamic desirable
 channel-group 14 mode active
!
interface FastEthernet0/21
 switchport mode dynamic desirable
 channel-group 14 mode active

SW2:
interface FastEthernet0/13
 channel-group 12 mode passive
!
interface FastEthernet0/14
 channel-group 12 mode passive
!
interface FastEthernet0/15
 channel-group 12 mode passive
```

Copyright © 2010 Internetwork Expert

```
SW3:
interface FastEthernet0/13
 channel-group 13 mode passive
!
interface FastEthernet0/14
 channel-group 13 mode passive
!
interface FastEthernet0/15
 channel-group 13 mode passive

SW4:
interface FastEthernet0/13
 channel-group 14 mode passive
!
interface FastEthernet0/14
 channel-group 14 mode passive
!
interface FastEthernet0/15
 channel-group 14 mode passive
```

Verification

> **Note**
>
> Similar to the previous variation of EtherChannel, Link Aggregation Control Protocol (LACP) is used to negotiate the formation of the channels from SW1 to SW2, SW3, and SW4. LACP is an open standard defined in IEEE 802.3ad. The active mode of LACP, like the desirable mode of PAgP, is used to initiate LACP negotiation, while the passive most is used to only respond to negotiation. Like PAgP this implies that a channel will form via LACP if one side is active and the other side is active or passive, but a channel will not form if both sides are passive.

```
Rack1SW1#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 3
Number of aggregators:           3

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
12     Po12(SU)         LACP      Fa0/13(P)   Fa0/14(P)   Fa0/15(P)
13     Po13(SU)         LACP      Fa0/16(P)   Fa0/17(P)   Fa0/18(P)
14     Po14(SU)         LACP      Fa0/19(P)   Fa0/20(P)   Fa0/21(P)
```

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Po12        desirable    n-isl          trunking      1
Po13        desirable    n-isl          trunking      1
Po14        desirable    n-isl          trunking      1

Port        Vlans allowed on trunk
Po12        1-4094
Po13        1-4094
Po14        1-4094

Port        Vlans allowed and active in management domain
Po12        1,5,7-10,22,43,58,67,79,146
Po13        1,5,7-10,22,43,58,67,79,146
Po14        1,5,7-10,22,43,58,67,79,146

Port        Vlans in spanning tree forwarding state and not pruned
Po12        1,5,7-10,22,43,58,67,79,146
Po13        1,5,9,43,79
Po14        1,10,146

Rack1SW1#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    32778
              Address     000c.3045.4180
              Cost        9
              Port        168 (Port-channel13)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     001b.d490.7c00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 15

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Po12             Desg FWD 9         128.160  P2p
Po13             Root FWD 9         128.168  P2p
Po14             Desg FWD 9         128.176  P2p
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
12     Po12(SU)         LACP      Fa0/13(P)   Fa0/14(P)   Fa0/15(P)

Rack1SW2#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/6       on           802.1q         trunking      1
Po12        auto         n-isl          trunking      1

Port        Vlans allowed on trunk
Fa0/6       1-4094
Po12        1-4094

Port        Vlans allowed and active in management domain
Fa0/6       1,5,7-10,22,43,58,67,79,146
Po12        1,5,7-10,22,43,58,67,79,146

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/6       1,5,7-10,22,43,58,67,79,146
Po12        1,5,7,9-10,43,58,67,79,146

Rack1SW2#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID    Priority    32778
             Address     000c.3045.4180
             Cost        18
             Port        160 (Port-channel12)
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32778  (priority 32768 sys-id-ext 10)
             Address     001b.d4df.ec80
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time 15

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/6            Desg FWD 19        128.8    P2p
Po12             Root FWD 9         128.160  P2p
```

Copyright © 2010 Internetwork Expert

```
Rack1SW3#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
13     Po13(SU)         LACP      Fa0/13(P)   Fa0/14(P)   Fa0/15(P)

Rack1SW3#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Po13        desirable    n-isl          trunking      1

Port        Vlans allowed on trunk
Po13        1-4094

Port        Vlans allowed and active in management domain
Po13        1,5,7-10,22,43,58,67,79,146

Port        Vlans in spanning tree forwarding state and not pruned
Po13        1,5,7-10,22,43,58,67,79,146

Rack1SW3#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID    Priority    32778
             Address     000c.3045.4180
             This bridge is the root
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32778  (priority 32768 sys-id-ext 10)
             Address     000c.3045.4180
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time 15

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Po13             Desg FWD 9         128.66   P2p
```

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
14     Po14(SU)         LACP      Fa0/13(P)   Fa0/14(P)   Fa0/15(P)

Rack1SW4#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Po14        desirable    n-isl          trunking      1

Port        Vlans allowed on trunk
Po14        1-4094

Port        Vlans allowed and active in management domain
Po14        1,5,7-10,22,43,58,67,79,146

Port        Vlans in spanning tree forwarding state and not pruned
Po14        1,5,7-10,22,43,58,67,79,146

Rack1SW4#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    32778
              Address     000c.3045.4180
              Cost        18
              Port        65 (Port-channel14)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     000c.3045.d600
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 15

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Po14             Root FWD 9         128.65   P2p
```

Copyright © 2010 Internetwork Expert

1.15 Layer 3 EtherChannel

- Configure links Fa0/16 & Fa0/17 on SW4 and links Fa0/19 & Fa0/20 on SW2 to be bound together as a Layer 3 EtherChannel.
- Use Port-Channel number 24 and the subnet 155.X.108.0/24 per the diagram.
- Use an industry standard protocol to negotiate the Port-Channel configuration.
- Ensure IP reachability is obtained between these devices over the segment.

Configuration

```
SW2:
interface Port-channel24
 no switchport
 ip address 155.1.108.8 255.255.255.0
!
interface FastEthernet0/19
 no switchport
 channel-group 24 mode passive
!
interface FastEthernet0/20
 no switchport
 channel-group 24 mode passive

SW4:
interface Port-channel24
 no switchport
 ip address 155.1.108.10 255.255.255.0
!
interface FastEthernet0/16
 no switchport
 channel-group 24 mode active
!
interface FastEthernet0/17
 no switchport
 channel-group 24 mode active
```

Copyright © 2010 Internetwork Expert

Verification

Pitfall

One common problem with forming layer 3 EtherChannel links is the order of operations. The important point to remember is that when the **channel-group** command is issued, the attributes of the member interfaces are immediately inherited by the Port-Channel interface. This means that if the **channel-group** command is issued before the **no switchport** command, the channel interface will be layer 2 and the member interfaces will be layer 3. A subsequent attempt to issue the **channel-group** command will generate an error message saying that the channel interface and the members are not compatible. To resolve this problem simply issue the **no switchport** command before the **channel-group** command.

Note

If configured properly the state of the Port-channel from the show etherchannel summary command should show (RU) for routed and in use.

```
Rack1SW2#show etherchannel 24 summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 2
Number of aggregators:           2

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
24     Po24(RU)         LACP      Fa0/19(P)   Fa0/20(P)

Rack1SW2#ping 155.1.108.10

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.108.10, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```

Copyright © 2010 Internetwork Expert

The Port-Channel interface should show up as a normal layer 3 routed interface in the IP routing table.

```
Rack1SW2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW2(config)#ip routing
Rack1SW2(config)#end

Rack1SW2#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     155.1.0.0/24 is subnetted, 3 subnets
C       155.1.8.0 is directly connected, Vlan8
C       155.1.58.0 is directly connected, Vlan58
C       155.1.108.0 is directly connected, Port-channel24
     150.1.0.0/24 is subnetted, 1 subnets
C       150.1.8.0 is directly connected, Loopback0

Rack1SW4#show etherchannel 24 summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 2
Number of aggregators:           2

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
24     Po24(RU)         LACP      Fa0/16(P)   Fa0/17(P)
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.16 802.1q Tunneling

- Configure 802.1q trunk links between SW1 & SW2's interfaces Fa0/13, SW2's interface Fa0/16 & SW3's interface Fa0/16, and SW3's interface Fa0/19 & SW4's interface Fa0/19.
- Disable all other inter-switch links.
- Configure two Ethernet subinterfaces on R1 with the IP addresses 14.0.0.1/24 and 41.0.0.1/24 using VLANs 14 and 41 respectively.
- Configure two Ethernet subinterfaces on R4's second Ethernet interface1 with the IP addresses 14.0.0.4/24 and 41.0.0.4/24 using VLANs 14 and 41 respectively.
- Using VLAN 100 configure an 802.1q tunnel between SW1 and SW4 to connect R1 and R4.
- R1 and R4 should appear to be directly connected when viewing the **`show cdp neighbor`** output.

Configuration

```
R1:
interface FastEthernet0/0
 no shutdown
!
interface FastEthernet0/0.14
 encapsulation dot1Q 14
 ip address 14.0.0.1 255.255.255.0
!
interface FastEthernet0/0.41
 encapsulation dot1Q 41
 ip address 41.0.0.1 255.255.255.0

R4:
interface FastEthernet0/1
 no shutdown
!
interface FastEthernet0/1.14
 encapsulation dot1Q 14
 ip address 14.0.0.4 255.255.255.0
!
interface FastEthernet0/1.41
 encapsulation dot1Q 41
 ip address 41.0.0.4 255.255.255.0
```

Copyright © 2010 Internetwork Expert

```
SW1:
system mtu 1504
!
interface FastEthernet0/1
 switchport access vlan 100
 switchport mode dot1q-tunnel
 l2protocol-tunnel cdp
 no cdp enable
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk

SW2:
system mtu 1504
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk

SW3:
system mtu 1504
!
interface FastEthernet0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/19
 switchport trunk encapsulation dot1q
 switchport mode trunk

SW4:
system mtu 1504
!
interface FastEthernet0/4
 switchport access vlan 100
 switchport mode dot1q-tunnel
 l2protocol-tunnel cdp
 no cdp enable
!
interface FastEthernet0/19
 switchport trunk encapsulation dot1q
 switchport mode trunk
```

Copyright © 2010 Internetwork Expert

Verification

Note

802.1q tunneling, or QinQ tunneling, is commonly used by Metro Ethernet providers to offer a transparent layer 2 VPN to end customers. This design has the distinct advantage over layer 3 MPLS tunnels, as the customer edge device does not have to run a routing protocol with the service provider, and an advantage over layer 2 MPLS AToM or VPLS tunnels for the service provider as the equipment and platform requirements are very moderate.

Dot1q tunneling works by simply taking all traffic received by the customer, and appending a new Ethernet header with a new 802.1q tag onto it. This *metro tag* is used as a unique identifier for the particular customer. Combined with the layer 2 tunneling feature protocols such as CDP, STP, and VTP can be transparently transported between customer sites with no complex requirements in the customer network.

In this example VLAN 100 is used as the metro tag, or tunnel VLAN, for the dot1q tunnel transport between R1 and R4. When R1 and R4 send traffic that is already dot1q tagged from their subinterfaces into the switch network, the new tag of 100 is appended. This can be easily verified through the **`show cdp neighbor`** output on R1 or R4, as even though they are not connected CDP thinks that they are.

```
Rack1R4#show cdp neighbor
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
Rack1R1          Fas 0/1            125         R S I      2610XM    Fas 0/0
```

Copyright © 2010 Internetwork Expert

With the addition of the second 802.1q encapsulation an Ethernet frame already at the normal MTU of 1500 bytes will be using 1504 bytes in the service provider transit path. For this reason the **system mtu** command is adjusted on the layer 2 switches to allow for frames of this size. Note that a reload of the device is necessary before the MTU change actually goes into effect.

```
Rack1R4#ping 14.0.0.1 size 1500 df-bit

Type escape sequence to abort.
Sending 5, 1500-byte ICMP Echos to 14.0.0.1, timeout is 2 seconds:
Packet sent with the DF bit set
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/5/8 ms

Rack1R4#ping 41.0.0.1 size 1500 df-bit

Type escape sequence to abort.
Sending 5, 1500-byte ICMP Echos to 41.0.0.1, timeout is 2 seconds:
Packet sent with the DF bit set
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/5/8 ms
```

Although SW1, SW2, SW3, and SW4 do not know about the customer's VLANs of 14 and 41, they are still able to transport these as they are encapsulated inside VLAN 100.

```
Rack1SW1#show spanning-tree vlan 14

Spanning tree instance(s) for vlan 14 does not exist.

Rack1SW1#show spanning-tree vlan 41

Spanning tree instance(s) for vlan 41 does not exist.

Rack1SW1#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      on           802.1q         trunking      1

Port        Vlans allowed on trunk
Fa0/13      1-4094

Port        Vlans allowed and active in management domain
Fa0/13      1,100

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      1,100
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show spanning-tree vlan 14

Spanning tree instance(s) for vlan 14 does not exist.

Rack1SW2#show spanning-tree vlan 41

Spanning tree instance(s) for vlan 41 does not exist.

Rack1SW2#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      on           802.1q         trunking      1
Fa0/16      on           802.1q         trunking      1

Port        Vlans allowed on trunk
Fa0/13      1-4094
Fa0/16      1-4094

Port        Vlans allowed and active in management domain
Fa0/13      1,100
Fa0/16      1,100

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      1,100
Fa0/16      1,100

Rack1SW3#show spanning-tree vlan 14

Spanning tree instance(s) for vlan 14 does not exist.

Rack1SW3#show spanning-tree vlan 41

Spanning tree instance(s) for vlan 41 does not exist.

Rack1SW3#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/16      on           802.1q         trunking      1
Fa0/19      on           802.1q         trunking      1

Port        Vlans allowed on trunk
Fa0/16      1-4094
Fa0/19      1-4094

Port        Vlans allowed and active in management domain
Fa0/16      1,100
Fa0/19      1,100

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/16      1,100
Fa0/19      1,100
```

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show spanning-tree vlan 14

Spanning tree instance(s) for vlan 14 does not exist.

Rack1SW4#show spanning-tree vlan 41

Spanning tree instance(s) for vlan 41 does not exist.

Rack1SW4#show interface trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/19      on           802.1q         trunking      1

Port        Vlans allowed on trunk
Fa0/19      1-4094

Port        Vlans allowed and active in management domain
Fa0/19      1,100

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/19      1,100
```

Copyright © 2010 Internetwork Expert

1.17 EtherChannel over 802.1q Tunneling

- Remove the previous trunking and tunneling configuration.
- Configure an 802.1q trunk link between SW2 and SW3.
- Configure interfaces Fa0/13, Fa0/14, and Fa0/15 on SW1 as a layer 2 EtherChannel using PAgP for negotiation.
- Configure interfaces Fa0/19, Fa0/20, and Fa0/21 on SW4 as a layer 2 EtherChannel using PAgP for negotiation.
- Disable all other inter-switch links on SW1 and SW4.
- Configure SW2 and SW3 to tunnel the EtherChannel link between SW1 and SW4 using VLANs 100, 200, and 300.
- Tunnel Spanning-Tree Protocol along with CDP over these links so that SW1 and SW4 appear to be directly connected when viewing the **`show cdp neighbor`** output.
- SW1 and SW4 should form an 802.1q trunk link over this EtherChannel.
- To verify this configure SW1 and SW4's links to R1 and R4 in VLAN 146 per the diagram and ensure connectivity between R1 and R4.

Configuration

```
R1:
interface FastEthernet0/0
 ip address 155.1.146.1 255.255.255.0

R4:
interface FastEthernet0/1
 ip address 155.1.146.4 255.255.255.0

SW1:
vlan 146
!
interface FastEthernet0/1
 switchport access vlan 146
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
 channel-group 14 mode desirable
!
interface FastEthernet0/14
 switchport trunk encapsulation dot1q
 switchport mode trunk
 channel-group 14 mode desirable
!
interface FastEthernet0/15
 switchport trunk encapsulation dot1q
 switchport mode trunk
 channel-group 14 mode desirable
```

Copyright © 2010 Internetwork Expert

```
SW2:
vlan 100,200,300
!
interface FastEthernet0/13
 switchport access vlan 100
 switchport mode dot1q-tunnel
 l2protocol-tunnel cdp
 l2protocol-tunnel stp
 l2protocol-tunnel point-to-point pagp
!
interface FastEthernet0/14
 switchport access vlan 200
 switchport mode dot1q-tunnel
 l2protocol-tunnel cdp
 l2protocol-tunnel stp
 l2protocol-tunnel point-to-point pagp
!
interface FastEthernet0/15
 switchport access vlan 300
 switchport mode dot1q-tunnel
 l2protocol-tunnel cdp
 l2protocol-tunnel stp
 l2protocol-tunnel point-to-point pagp
!
interface FastEthernet0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk

SW3:
vlan 100,200,300
!
interface FastEthernet0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/19
 switchport access vlan 100
 switchport trunk encapsulation dot1q
 switchport mode dot1q-tunnel
 l2protocol-tunnel cdp
 l2protocol-tunnel stp
 l2protocol-tunnel point-to-point pagp
!
interface FastEthernet0/20
 switchport access vlan 200
 switchport mode dot1q-tunnel
 l2protocol-tunnel cdp
 l2protocol-tunnel stp
 l2protocol-tunnel point-to-point pagp
!
interface FastEthernet0/21
 switchport access vlan 300
 switchport mode dot1q-tunnel
 l2protocol-tunnel cdp
 l2protocol-tunnel stp
 l2protocol-tunnel point-to-point pagp
```

Copyright © 2010 Internetwork Expert

```
SW4:
vlan 146
!
interface FastEthernet0/4
 switchport access vlan 146
!
interface FastEthernet0/19
 switchport trunk encapsulation dot1q
 switchport mode trunk
 channel-group 14 mode auto
!
interface FastEthernet0/20
 switchport trunk encapsulation dot1q
 switchport mode trunk
 channel-group 14 mode auto
!
interface FastEthernet0/21
 switchport trunk encapsulation dot1q
 switchport mode trunk
 channel-group 14 mode auto
```

Verification

> **Note**
>
> By creating separate point-to-point tunnels through the usage of separate metro tags an EtherChannel between two customer edge switches can be transparently tunneled over the service provider network.

```
Rack1R4#ping 155.1.146.1 size 1500 df-bit

Type escape sequence to abort.
Sending 5, 1500-byte ICMP Echos to 155.1.146.1, timeout is 2 seconds:
Packet sent with the DF bit set
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/5/8 ms
```

Copyright © 2010 Internetwork Expert

SW1, who is the customer edge device, sees the root port for VLAN 146 as the port-channel interface, and the root bridge ID of 000c.3045.d600.

```
Rack1SW1#show spanning-tree vlan 146

VLAN0146
  Spanning tree enabled protocol ieee
  Root ID     Priority    32914
              Address     000c.3045.d600
              Cost        9
              Port        176 (Port-channel14)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32914  (priority 32768 sys-id-ext 146)
              Address     001b.d490.7c00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1            Desg FWD 19        128.3    P2p
Po14             Root FWD 9         128.176  P2p
```

SW4 agrees with this root bridge election for VLAN 146, indicating that these two devices are in the same spanning-tree domain for this VLAN.

```
Rack1SW4#show spanning-tree vlan 146

VLAN0146
  Spanning tree enabled protocol ieee
  Root ID     Priority    32914
              Address     000c.3045.d600
              This bridge is the root
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32914  (priority 32768 sys-id-ext 146)
              Address     000c.3045.d600
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/4            Desg FWD 19        128.4    P2p
Po14             Desg FWD 9         128.65   P2p
```

Copyright © 2010 Internetwork Expert

SW2 and SW3 do not agree on the STP topology for VLAN 146 the same as SW1 and SW4, because STP BPDUs received from SW1 and SW4 are transparently tunneled inside the metro VLAN tags of 100, 200, and 300.

```
Rack1SW2#show spanning-tree vlan 146

Spanning tree instance(s) for vlan 146 does not exist.

Rack1SW3#show spanning-tree vlan 146

Spanning tree instance(s) for vlan 146 does not exist.

Rack1SW1#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 3
Number of aggregators:           3

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
14     Po14(SU)         PAgP      Fa0/13(P)    Fa0/14(P)    Fa0/15(P)
```

SW1 and SW4 think that they are directly connected over these tunneled channel ports via CDP.

```
Rack1SW1#show cdp neighbor
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
Rack1SW4         Fas 0/15          153           S I      WS-C3550-2Fas 0/21
Rack1SW4         Fas 0/14          153           S I      WS-C3550-2Fas 0/20
Rack1SW4         Fas 0/13          153           S I      WS-C3550-2Fas 0/19
Rack1R1          Fas 0/1           131         R S I      2610XM    Fas 0/0
Rack1R3          Fas 0/3           128         R S I      2611XM    Fas 0/0
Rack1R5          Fas 0/5           124         R S I      2611XM    Fas 0/0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.18 STP Root Bridge Election

- Configure the inter-switch links between SW1 & SW2, SW1 & SW3, SW2 & SW4, and SW3 & SW4 as 802.1q trunk links.
- Disable all other inter-switch links.
- Configure SW4 as a VTP server using the domain name CCIE with SW1, SW2, and SW3 as its clients.
- Configure VLAN assignments per the diagram.
- Configure SW1 as the STP Root Bridge for all active VLANs.
- If SW1 goes down SW4 should take over as the STP Root Bridge for all active VLANs.

Configuration

```
SW1:
vtp domain CCIE
vtp mode client
!
spanning-tree vlan 1,5,7-10,22,43,58,67,79,146 priority 0
!
interface FastEthernet0/1
 switchport access vlan 146
!
interface FastEthernet0/5
 switchport access vlan 58
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/14
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/15
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/17
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/18
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/19
 shutdown
```

Copyright © 2010 Internetwork Expert

```
interface FastEthernet0/20
 shutdown
!
interface FastEthernet0/21
 shutdown

SW2:
vtp domain CCIE
vtp mode client
!
interface FastEthernet0/2
 switchport access vlan 22
!
interface FastEthernet0/4
 switchport access vlan 43
!
interface FastEthernet0/6
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/14
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/15
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/16
 shutdown
!
interface FastEthernet0/17
 shutdown
!
interface FastEthernet0/18
 shutdown
!
interface FastEthernet0/19
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/20
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/21
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/24
 switchport access vlan 22
```

Copyright © 2010 Internetwork Expert

```
SW3:
vtp domain CCIE
vtp mode client
!
interface FastEthernet0/5
 switchport access vlan 5
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/14
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/15
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/16
 shutdown
!
interface FastEthernet0/17
 shutdown
!
interface FastEthernet0/18
 shutdown
!
interface FastEthernet0/19
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/20
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/21
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/24
 switchport access vlan 43
```

Copyright © 2010 Internetwork Expert

```
SW4:
vtp domain CCIE
vlan 5,7,8,9,10,22,43,58,67,79,146
!
spanning-tree vlan 1,5,7-10,22,43,58,67,79,146 priority 4096
!
interface FastEthernet0/4
 switchport access vlan 146
!
interface FastEthernet0/13
 shutdown
!
interface FastEthernet0/14
 shutdown
!
interface FastEthernet0/15
 shutdown
!
interface FastEthernet0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/17
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/18
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/19
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/20
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/21
 switchport trunk encapsulation dot1q
 switchport mode trunk
```

Copyright © 2010 Internetwork Expert

Verification

```
Rack1SW1#show spanning-tree vlan 146 | include root
             This bridge is the root
Rack1SW1#show spanning-tree vlan 1 | include root
             This bridge is the root
Rack1SW1#show spanning-tree vlan 5 | include root
             This bridge is the root
Rack1SW1#show spanning-tree vlan 7 | include root
             This bridge is the root
Rack1SW1#show spanning-tree vlan 8 | include root
             This bridge is the root
Rack1SW1#show spanning-tree vlan 9 | include root
             This bridge is the root
Rack1SW1#show spanning-tree vlan 10 | include root
             This bridge is the root
Rack1SW1#show spanning-tree vlan 22 | include root
             This bridge is the root
Rack1SW1#show spanning-tree vlan 43 | include root
             This bridge is the root
Rack1SW1#show spanning-tree vlan 58 | include root
             This bridge is the root
Rack1SW1#show spanning-tree vlan 67 | include root
             This bridge is the root
Rack1SW1#show spanning-tree vlan 79 | include root
             This bridge is the root
Rack1SW1#show spanning-tree vlan 146 | include root
             This bridge is the root
```

Copyright © 2010 Internetwork Expert

Note

STP root bridge election is based on the priority and MAC address fields of the Bridge ID. The device with the lowest priority value is elected the root. If there is a tie in priority the device with the lowest MAC address is elected root. SW1 with the local priority of one, the configured priority of zero plus the system id extension (VLAN number), shows that *This bridge is the root*. The root bridge should show the same priority and MAC address for both the Root ID and the Bridge ID, and list all interfaces as Designated (downstream facing). In this case SW1's BID is 1.001b.d490.7c00.

```
Rack1SW1#show spanning-tree vlan 1

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    1
              Address     001b.d490.7c00
              This bridge is the root
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    1      (priority 0 sys-id-ext 1)
              Address     001b.d490.7c00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/13           Desg FWD 19        128.15   P2p
Fa0/14           Desg FWD 19        128.16   P2p
Fa0/15           Desg FWD 19        128.17   P2p
Fa0/16           Desg FWD 19        128.18   P2p
Fa0/17           Desg FWD 19        128.19   P2p
Fa0/18           Desg FWD 19        128.20   P2p
```

Copyright © 2010 Internetwork Expert

SW2 agrees that the device with the BID 1.001b.d490.7c00 is the root, and uses the port Fa0/13 with a total cost of 19 to reach it. SW2's local BID is a priority of 32769, the default of 32768 plus the system id extension 1, and the MAC address 001b.d4df.ec80.

```
Rack1SW2#show spanning-tree vlan 1

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    1
              Address     001b.d490.7c00
              Cost        19
              Port        15 (FastEthernet0/13)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32769  (priority 32768 sys-id-ext 1)
              Address     001b.d4df.ec80
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/6            Desg FWD 19        128.8    P2p
Fa0/13           Root FWD 19        128.15   P2p
Fa0/14           Altn BLK 19        128.16   P2p
Fa0/15           Altn BLK 19        128.17   P2p
Fa0/19           Desg FWD 19        128.21   P2p
Fa0/20           Desg FWD 19        128.22   P2p
Fa0/21           Desg FWD 19        128.23   P2p
```

Copyright © 2010 Internetwork Expert

SW3 agrees that the device with the BID 1.001b.d490.7c00 is the root, and uses the port Fa0/13 with a total cost of 19 to reach it. SW3's local BID is a priority of 32769, the default of 32768 plus the system id extension 1, and the MAC address 000c.3045.4180.

```
Rack1SW3#show spanning-tree vlan 1

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    1
              Address     001b.d490.7c00
              Cost        19
              Port        13 (FastEthernet0/13)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32769  (priority 32768 sys-id-ext 1)
              Address     000c.3045.4180
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/13           Root FWD 19        128.13   P2p
Fa0/14           Altn BLK 19        128.14   P2p
Fa0/15           Altn BLK 19        128.15   P2p
Fa0/19           Desg FWD 19        128.19   P2p
Fa0/20           Desg FWD 19        128.20   P2p
Fa0/21           Desg FWD 19        128.21   P2p
```

Copyright © 2010 Internetwork Expert

Likewise SW4 agrees that the device with the BID 1.001b.d490.7c00 is the root, but SW4 has a lower priority than SW2 or SW3. This means that if the root bridge were to fail SW4 would be next in line to take over the root status.

```
Rack1SW4#show spanning-tree vlan 1

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    1
              Address     001b.d490.7c00
              Cost        38
              Port        19 (FastEthernet0/19)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    4097   (priority 4096 sys-id-ext 1)
              Address     000c.3045.d600
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/16           Altn BLK 19        128.16   P2p
Fa0/17           Altn BLK 19        128.17   P2p
Fa0/18           Altn BLK 19        128.18   P2p
Fa0/19           Root FWD 19        128.19   P2p
Fa0/20           Altn BLK 19        128.20   P2p
Fa0/21           Altn BLK 19        128.21   P2p
```

Copyright © 2010 Internetwork Expert

When SW1's trunk links are down SW4 should assume the role of the root bridge since it has the next lowest bridge priority value.

```
Rack1SW1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface range fa0/13 - 18
Rack1SW1(config-if-range)#shut
Rack1SW1(config-if-range)#

Rack1SW4#show spanning-tree vlan 1

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    4097
              Address     000c.3045.d600
              This bridge is the root
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    4097   (priority 4096 sys-id-ext 1)
              Address     000c.3045.d600
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 15

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/16           Desg LIS 19        128.16   P2p
Fa0/17           Desg LIS 19        128.17   P2p
Fa0/18           Desg LIS 19        128.18   P2p
Fa0/19           Desg FWD 19        128.19   P2p
Fa0/20           Desg LIS 19        128.20   P2p
Fa0/21           Desg LIS 19        128.21   P2p
```

Copyright © 2010 Internetwork Expert

1.19 STP Load Balancing with Port Cost

- Using Spanning-Tree cost modify the layer 2 transit network so that traffic for all active VLANs from SW2 to SW1 uses the last link between SW2 and SW4.
- If this link goes down traffic should fall over to the second link between SW2 and SW4.

Configuration

```
SW2:
interface FastEthernet0/13
 spanning-tree cost 1000
!
interface FastEthernet0/14
 spanning-tree cost 1000
!
interface FastEthernet0/15
 spanning-tree cost 1000
!
interface FastEthernet0/20
 spanning-tree cost 2
!
interface FastEthernet0/21
 spanning-tree cost 1
```

Verification

```
Rack1SW2#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    10
              Address     001b.d490.7c00
              Cost        19
              Port        15 (FastEthernet0/13)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     001b.d4df.ec80
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/13           Root FWD 19        128.15   P2p
Fa0/14           Altn BLK 19        128.16   P2p
Fa0/15           Altn BLK 19        128.17   P2p
Fa0/19           Desg FWD 19        128.21   P2p
Fa0/20           Desg FWD 19        128.22   P2p
Fa0/21           Desg FWD 19        128.23   P2p
```

Copyright © 2010 Internetwork Expert

Note

The default cost to the root bridge from SW2 before configuration changes is 19. By changing the links to SW1 to a cost of 1000 they are the least preferred path. By changing the last link to SW4 to a cost of 1 the end to end path cost on that link becomes 39, which is the most preferred (1 to SW4, 19 from SW4 to SW3, 19 from SW3 to SW1). With the second to last link having a cost of 2, the end to end path cost will be 40, and will therefore be the second most preferred link.

```
Rack1SW2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW2(config)#interface range fa0/13 - 15
Rack1SW2(config-if-range)#spanning-tree cost 1000
Rack1SW2(config-if-range)#interface fa0/21
% Command exited out of interface range and its sub-modes.
  Not executing the command for second and later interfaces
Rack1SW2(config-if)#spanning-tree cost 1
Rack1SW2(config-if-range)#interface fa0/20
Rack1SW2(config-if)#spanning-tree cost 2
Rack1SW2(config-if)#end
Rack1SW2#

Rack1SW2#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    10
              Address     001b.d490.7c00
              Cost        39
              Port        23 (FastEthernet0/21)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     001b.d4df.ec80
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 15

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/13           Altn BLK 1000      128.15   P2p
Fa0/14           Altn BLK 1000      128.16   P2p
Fa0/15           Altn BLK 1000      128.17   P2p
Fa0/19           Altn BLK 19        128.21   P2p
Fa0/20           Altn BLK 2         128.22   P2p
Fa0/21           Root FWD 1         128.23   P2p
```

Copyright © 2010 Internetwork Expert

1.20 STP Load Balancing with Port Priority

- Using Spanning-Tree priority modify the layer 2 transit network so that traffic for all active VLANs from SW4 to SW1 uses the last link between SW3 and SW4.
- If this link goes down traffic should fall over to the second link between SW3 and SW4.

Configuration

```
SW3:
interface FastEthernet0/20
 spanning-tree port-priority 16
!
interface FastEthernet0/21
 spanning-tree port-priority 0
```

Verification

Note

Before configuration changes:

```
Rack1SW4#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    10
              Address     001b.d490.7c00
              Cost        38
              Port        19 (FastEthernet0/19)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    4106   (priority 4096 sys-id-ext 10)
              Address     000c.3045.d600
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/16           Desg FWD 19        128.16   P2p
Fa0/17           Desg FWD 19        128.17   P2p
Fa0/18           Desg FWD 19        128.18   P2p
Fa0/19           Root FWD 19        128.19   P2p
Fa0/20           Altn BLK 19        128.20   P2p
Fa0/21           Altn BLK 19        128.21   P2p
```

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show spanning-tree vlan 10 detail

 VLAN0010 is executing the ieee compatible Spanning Tree protocol
  Bridge Identifier has priority 4096, sysid 10, address 000c.3045.d600
  Configured hello time 2, max age 20, forward delay 15
  Current root has priority 10, address 001b.d490.7c00
  Root port is 19 (FastEthernet0/19), cost of root path is 38
  Topology change flag not set, detected flag not set
  Number of topology changes 5 last change occurred 00:08:08 ago
          from FastEthernet0/16
  Times:  hold 1, topology change 35, notification 2
          hello 2, max age 20, forward delay 15
  Timers: hello 0, topology change 0, notification 0, aging 300

 Port 16 (FastEthernet0/16) of VLAN0010 is forwarding
   Port path cost 19, Port priority 128, Port Identifier 128.16.
   Designated root has priority 10, address 001b.d490.7c00
   Designated bridge has priority 4106, address 000c.3045.d600
   Designated port id is 128.16, designated path cost 38
   Timers: message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is point-to-point by default
   BPDU: sent 266, received 119

 Port 17 (FastEthernet0/17) of VLAN0010 is forwarding
   Port path cost 19, Port priority 128, Port Identifier 128.17.
   Designated root has priority 10, address 001b.d490.7c00
   Designated bridge has priority 4106, address 000c.3045.d600
   Designated port id is 128.17, designated path cost 38
   Timers: message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is point-to-point by default
   BPDU: sent 266, received 118

 Port 18 (FastEthernet0/18) of VLAN0010 is forwarding
   Port path cost 19, Port priority 128, Port Identifier 128.18.
   Designated root has priority 10, address 001b.d490.7c00
   Designated bridge has priority 4106, address 000c.3045.d600
   Designated port id is 128.18, designated path cost 38
   Timers: message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is point-to-point by default
   BPDU: sent 266, received 119

 Port 19 (FastEthernet0/19) of VLAN0010 is forwarding
   Port path cost 19, Port priority 128, Port Identifier 128.19.
   Designated root has priority 10, address 001b.d490.7c00
   Designated bridge has priority 32778, address 000c.3045.4180
   Designated port id is 128.19, designated path cost 19
   Timers: message age 2, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is point-to-point by default
   BPDU: sent 71, received 1126
```

Copyright © 2010 Internetwork Expert

```
Port 20 (FastEthernet0/20) of VLAN0010 is blocking
  Port path cost 19, Port priority 128, Port Identifier 128.20.
  Designated root has priority 10, address 001b.d490.7c00
  Designated bridge has priority 32778, address 000c.3045.4180
  Designated port id is 128.20, designated path cost 19
  Timers: message age 3, forward delay 0, hold 0
  Number of transitions to forwarding state: 1
  Link type is point-to-point by default
  BPDU: sent 69, received 1125

Port 21 (FastEthernet0/21) of VLAN0010 is blocking
  Port path cost 19, Port priority 128, Port Identifier 128.21.
  Designated root has priority 10, address 001b.d490.7c00
  Designated bridge has priority 32778, address 000c.3045.4180
  Designated port id is 128.21, designated path cost 19
  Timers: message age 3, forward delay 0, hold 0
  Number of transitions to forwarding state: 1
  Link type is point-to-point by default
  BPDU: sent 69, received 1125
```

Since interfaces Fa0/19 – 21 on SW4 all have the same end to end path cost of 38 the designated (upstream) bridge-id is compared. Since SW4 is connected to SW3 out all three links, there is a tie in the designated bridge-id, and the designated (upstream) port id is compared. Since the upstream port number of Fa0/19 is 19, versus 20 and 21, Fa0/19 is the root port on SW4.

By changing the upstream priority on SW3 on ports Fa0/20 and Fa0/21, SW4 prefers the port with the lowest designated port priority. If interface Fa0/21 on SW4 goes down it will compare the upstream priority of Fa0/20 (16) with the upstream priority of Fa0/19 (128), and Fa0/20 will be chosen.

```
Rack1SW3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW3(config)#interface fa0/21
Rack1SW3(config-if)#spanning-tree port-priority 0
Rack1SW3(config-if)#interface fa0/20
Rack1SW3(config-if)#spanning-tree port-priority 16
Rack1SW3(config-if)#end
Rack1SW3#
```

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    10
              Address     001b.d490.7c00
              Cost        38
              Port        21 (FastEthernet0/21)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    4106   (priority 4096 sys-id-ext 10)
              Address     000c.3045.d600
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 15

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/16           Desg FWD 19        128.16   P2p
Fa0/17           Desg FWD 19        128.17   P2p
Fa0/18           Desg FWD 19        128.18   P2p
Fa0/19           Altn BLK 19        128.19   P2p
Fa0/20           Altn BLK 19        128.20   P2p
Fa0/21           Root LRN 19        128.21   P2p

Rack1SW4#show spanning-tree vlan 10 detail

 VLAN0010 is executing the ieee compatible Spanning Tree protocol
  Bridge Identifier has priority 4096, sysid 10, address 000c.3045.d600
  Configured hello time 2, max age 20, forward delay 15
  Current root has priority 10, address 001b.d490.7c00
  Root port is 21 (FastEthernet0/21), cost of root path is 38
  Topology change flag set, detected flag not set
  Number of topology changes 6 last change occurred 00:00:19 ago
          from FastEthernet0/19
  Times:  hold 1, topology change 35, notification 2
          hello 2, max age 20, forward delay 15
  Timers: hello 0, topology change 0, notification 0, aging 15

 Port 16 (FastEthernet0/16) of VLAN0010 is forwarding
   Port path cost 19, Port priority 128, Port Identifier 128.16.
   Designated root has priority 10, address 001b.d490.7c00
   Designated bridge has priority 4106, address 000c.3045.d600
   Designated port id is 128.16, designated path cost 38
   Timers: message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is point-to-point by default
   BPDU: sent 293, received 119

 Port 17 (FastEthernet0/17) of VLAN0010 is forwarding
   Port path cost 19, Port priority 128, Port Identifier 128.17.
   Designated root has priority 10, address 001b.d490.7c00
   Designated bridge has priority 4106, address 000c.3045.d600
   Designated port id is 128.17, designated path cost 38
   Timers: message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is point-to-point by default
   BPDU: sent 294, received 118
```

Copyright © 2010 Internetwork Expert

```
Port 18 (FastEthernet0/18) of VLAN0010 is forwarding
  Port path cost 19, Port priority 128, Port Identifier 128.18.
  Designated root has priority 10, address 001b.d490.7c00
  Designated bridge has priority 4106, address 000c.3045.d600
  Designated port id is 128.18, designated path cost 38
  Timers: message age 0, forward delay 0, hold 0
  Number of transitions to forwarding state: 1
  Link type is point-to-point by default
  BPDU: sent 294, received 119

Port 19 (FastEthernet0/19) of VLAN0010 is blocking
  Port path cost 19, Port priority 128, Port Identifier 128.19.
  Designated root has priority 10, address 001b.d490.7c00
  Designated bridge has priority 32778, address 000c.3045.4180
  Designated port id is 128.19, designated path cost 19
  Timers: message age 3, forward delay 0, hold 0
  Number of transitions to forwarding state: 1
  Link type is point-to-point by default
  BPDU: sent 71, received 1152

Port 20 (FastEthernet0/20) of VLAN0010 is blocking
  Port path cost 19, Port priority 128, Port Identifier 128.20.
  Designated root has priority 10, address 001b.d490.7c00
  Designated bridge has priority 32778, address 000c.3045.4180
  Designated port id is 16.20, designated path cost 19
  Timers: message age 2, forward delay 0, hold 0
  Number of transitions to forwarding state: 1
  Link type is point-to-point by default
  BPDU: sent 69, received 1152

Port 21 (FastEthernet0/21) of VLAN0010 is learning
  Port path cost 19, Port priority 128, Port Identifier 128.21.
  Designated root has priority 10, address 001b.d490.7c00
  Designated bridge has priority 32778, address 000c.3045.4180
  Designated port id is 0.21, designated path cost 19
  Timers: message age 2, forward delay 7, hold 0
  Number of transitions to forwarding state: 1
  Link type is point-to-point by default
  BPDU: sent 70, received 1153
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.21 Tuning STP Convergence Timers

- Configure the switches so that they broadcast Spanning-Tree hello packets every three seconds.
- When a new port becomes active it should wait twenty seconds before transitioning to the forwarding state.
- If the switches do not hear a configuration message within ten seconds they should attempt reconfiguration.
- This configuration should impact all currently active VLANs and any additional VLANs created in the future.

Configuration

```
SW1:
spanning-tree vlan 1-4094 hello-time 3
spanning-tree vlan 1-4094 forward-time 10
spanning-tree vlan 1-4094 max-age 10
```

Verification

```
Rack1SW3#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    10
              Address     001b.d490.7c00
              Cost        19
              Port        13 (FastEthernet0/13)
              Hello Time   3 sec  Max Age 10 sec  Forward Delay 10 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     000c.3045.4180
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/13           Root FWD 19        128.13   P2p
Fa0/14           Altn BLK 19        128.14   P2p
Fa0/15           Altn BLK 19        128.15   P2p
Fa0/19           Desg FWD 19        128.19   P2p
Fa0/20           Desg FWD 19         16.20   P2p
Fa0/21           Desg FWD 19           0.21  P2p
```

Copyright © 2010 Internetwork Expert

Note

Downstream devices from the root bridge inherit the timers configured on the root. With a forward delay of 10 seconds configured on SW1 the downstream switches should take 10 seconds in each of the listening and learning phases during convergence. The below timestamps indicate that a new root port was elected at 04:56:40 on SW3 and transitions from blocking to listening. 10 seconds later, at 04:56:50, the port transitions from listening to learning. Finally 10 seconds after that, at 04:57:00, the port transitions into forwarding.

```
Rack1SW3#debug spanning-tree events
Spanning Tree event debugging is on
Rack1SW3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW3(config)#service timestamps log
Rack1SW3(config)#logging console 7
Rack1SW3(config)#interface fa0/13
Rack1SW3(config-if)#shut

04:56:40: STP: VLAN0001 new root port Fa0/14, cost 19
04:56:40: STP: VLAN0001 Fa0/14 -> listening
04:56:40: STP: VLAN0005 new root port Fa0/14, cost 19
04:56:40: STP: VLAN0005 Fa0/14 -> listening
<output omitted>
04:56:43: STP: VLAN0001 sent Topology Change Notice on Fa0/14
04:56:43: STP: VLAN0005 sent Topology Change Notice on Fa0/14
<output omitted>
04:56:50: STP: VLAN0001 Fa0/14 -> learning
04:56:50: STP: VLAN0005 Fa0/14 -> learning
<output omitted>
04:57:00: STP: VLAN0001 sent Topology Change Notice on Fa0/14
04:57:00: STP: VLAN0001 Fa0/14 -> forwarding
04:57:00: STP: VLAN0005 sent Topology Change Notice on Fa0/14
04:57:00: STP: VLAN0005 Fa0/14 -> forwarding
<output omitted>
```

Copyright © 2010 Internetwork Expert

1.22 STP PortFast

- Configure Spanning-Tree PortFast on the switches so that ports connected to the internal and external routers do not have to wait for the Spanning-Tree listening and learning phases to begin forwarding.
- Do not use any global Spanning-Tree commands to accomplish this.

Configuration

```
SW1:
interface FastEthernet0/1
 spanning-tree portfast
!
interface FastEthernet0/5
 spanning-tree portfast

SW2:
interface FastEthernet0/2
 spanning-tree portfast
!
interface FastEthernet0/4
 spanning-tree portfast
!
interface FastEthernet0/6
 spanning-tree portfast trunk
!
interface FastEthernet0/24
 spanning-tree portfast

SW3:
interface FastEthernet0/5
 spanning-tree portfast
!
interface FastEthernet0/24
 spanning-tree portfast

SW4:
interface FastEthernet0/4
 spanning-tree portfast
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Portfast is used to override the listening and learning phases of spanning-tree, also called the forwarding delay, and transition immediately to forwarding.

```
Rack1SW1#show spanning-tree interface fa0/1 portfast
VLAN0146         enabled

Rack1SW1#debug spanning-tree event
Spanning Tree event debugging is on
Rack1SW1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#service timestamp log
Rack1SW1(config)#logging console 7
Rack1SW1(config)#interface fa0/1
Rack1SW1(config-if)#shutdown
05:08:43: %LINK-5-CHANGED: Interface FastEthernet0/1, changed state to
administratively down
05:08:44: %LINEPROTO-5-UPDOWN: Line protocol on Interface
FastEthernet0/1, changed state to down
```

When interface Fa0/1 is shutdown and subsequently brought back up it immediately transitions to the forwarding state.

```
Rack1SW1(config-if)#no shutdown
Rack1SW1(config-if)#
05:08:52: set portid: VLAN0146 Fa0/1: new port id 8003
05:08:52: STP: VLAN0146 Fa0/1 ->jump to forwarding from blocking
Rack1SW1(config-if)#
05:08:53: %LINK-3-UPDOWN: Interface FastEthernet0/1, changed state to
up
05:08:54: %LINEPROTO-5-UPDOWN: Line protocol on Interface
FastEthernet0/1, changed state to up
Rack1SW1(config-if)#end
Rack1SW1#
```

Copyright © 2010 Internetwork Expert

1.23 STP PortFast Default

- Remove the previous PortFast configuration.
- Configure Spanning-Tree PortFast on the switches so that ports connected to the internal and external routers do not have to wait for the Spanning-Tree listening and learning phases to begin forwarding.
- Do not use any interface level Spanning-Tree commands to accomplish this.

Configuration

```
SW1:
spanning-tree portfast default

SW2:
spanning-tree portfast default

SW3:
spanning-tree portfast default

SW4:
spanning-tree portfast default
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Portfast default has the same affect as the interface level portfast command, however it is automatically enabled on all interfaces at the same time. This command is the equivalent of issuing the **spanning-tree portfast** command under an interface range that encompasses all interfaces.

```
Rack1SW1#show run interface fa0/1
Building configuration...

Current configuration : 61 bytes
!
interface FastEthernet0/1
 switchport access vlan 146
end

Rack1SW1#show spanning-tree interface fa0/1 portfast
VLAN0146         enabled

Rack1SW1#debug spanning-tree event
Spanning Tree event debugging is on
Rack1SW1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface fa0/1
Rack1SW1(config-if)#shutdown
Rack1SW1(config-if)#
05:13:55: %LINK-5-CHANGED: Interface FastEthernet0/1, changed state to
administratively down
05:13:56: %LINEPROTO-5-UPDOWN: Line protocol on Interface
FastEthernet0/1, changed state to down
Rack1SW1(config-if)#no shutdown
Rack1SW1(config-if)#
05:14:03: set portid: VLAN0146 Fa0/1: new port id 8003
05:14:03: STP: VLAN0146 Fa0/1 ->jump to forwarding from blocking
Rack1SW1(config-if)#
05:14:03: %LINK-3-UPDOWN: Interface FastEthernet0/1, changed state to
up
05:14:04: %LINEPROTO-5-UPDOWN: Line protocol on Interface
FastEthernet0/1, changed state to up
Rack1SW1(config-if)#
```

Copyright © 2010 Internetwork Expert

1.24 STP UplinkFast

- Configure SW2, SW3, and SW4 with Spanning-Tree UplinkFast such that if their root port is lost they immediately reconverge to an alternate connection to their upstream bridge.
- Verify this by shutting down the root port of SW2.

Configuration

```
SW2:
spanning-tree uplinkfast

SW3:
spanning-tree uplinkfast

SW4:
spanning-tree uplinkfast
```

Verification

> **Note**
>
> The Cisco proprietary UplinkFast feature is used to speed up convergence time when the direct failure of the local root port occurs. In this particular design interface Fa0/13 on SW2 is the current root port.

```
Rack1SW2#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    10
              Address     001b.d490.7c00
              Cost        4000
              Port        15 (FastEthernet0/13)
              Hello Time   3 sec  Max Age 10 sec  Forward Delay 10 sec

  Bridge ID   Priority    49162  (priority 49152 sys-id-ext 10)
              Address     001b.d4df.ec80
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300
  Uplinkfast enabled

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/6            Desg FWD 3019      128.8    P2p
Fa0/13           Root FWD 4000      128.15   P2p
Fa0/14           Altn BLK 4000      128.16   P2p
Fa0/15           Altn BLK 4000      128.17   P2p
Fa0/19           Altn BLK 3019      128.21   P2p
Fa0/20           Altn BLK 3002      128.22   P2p
Fa0/21           Altn BLK 3001      128.23   P2p
```

Copyright © 2010 Internetwork Expert

With the failure of the root port the next alternate port is immediately transitioned to the root port in forwarding state, and the CAM table is flooded out this new root port to expedite the learning phase of upstream neighbors.

```
Rack1SW2#debug spanning-tree event
Spanning Tree event debugging is on
Rack1SW2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW2(config)#service timestamp log
Rack1SW2(config)#logging console 7
Rack1SW2(config)#interface fa0/13
Rack1SW2(config-if)#shut
Rack1SW2(config-if)#
05:16:42: STP: VLAN0001 new root port Fa0/14, cost 4000
05:16:42: %SPANTREE_FAST-7-PORT_FWD_UPLINK: VLAN0001 FastEthernet0/14
moved to Forwarding (UplinkFast).
05:16:42: STP: VLAN0005 new root port Fa0/14, cost 4000
05:16:42: STP: VLAN0007 new root port Fa0/14, cost 4000
05:16:42: STP: VLAN0008 new root port Fa0/14, cost 4000
05:16:42: STP: VLAN0009 new root port Fa0/14, cost 4000
05:16:42: STP: VLAN0010 new root port Fa0/14, cost 4000
05:16:42: STP: VLAN0022 new root port Fa0/14, cost 4000
05:16:42: STP: VLAN0043 new root port Fa0/14, cost 4000
05:16:42: STP: VLAN0058 new root port Fa0/14, cost 4000
05:16:42: STP: VLAN0067 new root port Fa0/14, cost 4000
05:16:42: STP: VLAN0079 new root port Fa0/14, cost 4000
05:16:42: STP: VLAN0146 new root port Fa0/14, cost 4000
05:16:44: %LINK-5-CHANGED: Interface FastEthernet0/13, changed state to
administratively down
05:16:45: %LINEPROTO-5-UPDOWN: Line protocol on Interface
FastEthernet0/13, changed state to down
05:16:45: STP: VLAN0001 sent Topology Change Notice on Fa0/14
05:16:45: STP: VLAN0005 sent Topology Change Notice on Fa0/14
05:16:45: STP: VLAN0007 sent Topology Change Notice on Fa0/14
05:16:45: STP: VLAN0008 sent Topology Change Notice on Fa0/14
05:16:45: STP: VLAN0009 sent Topology Change Notice on Fa0/14
05:16:45: STP: VLAN0010 sent Topology Change Notice on Fa0/14
05:16:45: STP: VLAN0022 sent Topology Change Notice on Fa0/14
05:16:45: STP: VLAN0043 sent Topology Change Notice on Fa0/14
05:16:45: STP: VLAN0058 sent Topology Change Notice on Fa0/14
05:16:45: STP: VLAN0067 sent Topology Change Notice on Fa0/14
05:16:45: STP: VLAN0079 sent Topology Change Notice on Fa0/14
05:16:45: STP: VLAN0146 sent Topology Change Notice on Fa0/14
05:16:58: %SYS-5-CONFIG_I: Configured from console by console
```

Copyright © 2010 Internetwork Expert

1.25 STP BackboneFast

- Configure Spanning-Tree BackboneFast such that if the links between SW3 and SW4 go down SW2 immediately expires its maxage timer and begins Spanning-Tree reconvergence.

Configuration

```
SW1:
spanning-tree backbonefast

SW2:
spanning-tree backbonefast

SW3:
spanning-tree backbonefast

SW4:
spanning-tree backbonefast
```

Verification

> **Note**
>
> The Cisco proprietary BackboneFast feature is used to speed up convergence when an indirect failure occurs upstream in the network by immediately expiring the max_age timer. In this design SW2's root port is towards SW4 on Fa0/21.

```
Rack1SW2#show spanning-tree vlan 10

VLAN0010
  Spanning tree enabled protocol ieee
  Root ID     Priority    10
              Address     001b.d490.7c00
              Cost        39
              Port        23 (FastEthernet0/21)
              Hello Time   3 sec  Max Age 10 sec  Forward Delay 10 sec

  Bridge ID   Priority    32778  (priority 32768 sys-id-ext 10)
              Address     001b.d4df.ec80
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 10

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/6            Desg FWD 19        128.8    P2p
Fa0/13           Altn BLK 1000      128.15   P2p
Fa0/14           Altn BLK 1000      128.16   P2p
Fa0/15           Altn BLK 1000      128.17   P2p
Fa0/19           Altn BLK 19        128.21   P2p
Fa0/20           Altn BLK 2         128.22   P2p
Fa0/21           Root FWD 1         128.23   P2p
```

SW4 loses its path to the root bridge causing it to send inferior BPDUs downstream to SW2. Since BackboneFast is enabled, SW2 generates Root Link Query (RLQ) PDUs to check if it should expire max_age for its current BPDUs and begin reconvergence.

```
Rack1SW4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW4(config)#interface range fa0/19 - 21
Rack1SW4(config-if)#shutdown
Rack1SW4(config-if)#

Rack1SW2#debug spanning-tree backbonefast
Spanning Tree backbonefast general debugging is on
Rack1SW2#
STP FAST: received inferior BPDU on VLAN0001 FastEthernet0/19.
STP FAST: sending RLQ request PDU on VLAN0001(1) Fa0/13 Vlan1
STP FAST: sending RLQ request PDU on VLAN0001(1) Fa0/14 Vlan1
STP FAST: sending RLQ request PDU on VLAN0001(1) Fa0/15 Vlan1
STP FAST: sending RLQ request PDU on VLAN0001(1) Fa0/20 Vlan1
STP FAST: sending RLQ request PDU on VLAN0001(1) Fa0/21 Vlan1
STP FAST: received inferior BPDU on VLAN0001 FastEthernet0/20.
STP FAST: sending RLQ request PDU on VLAN0001(1) Fa0/13 Vlan1
STP FAST: sending RLQ request PDU on VLAN0001(1) Fa0/14 Vlan1
STP FAST: sending RLQ request PDU on VLAN0001(1) Fa0/15 Vlan1
STP FAST: sending RLQ request PDU on VLAN0001(1) Fa0/21 Vlan1
STP FAST: received inferior BPDU on VLAN0001 FastEthernet0/21.
STP FAST: sending RLQ request PDU on VLAN0001(1) Fa0/13 Vlan1
STP FAST: sending RLQ request PDU on VLAN0001(1) Fa0/14 Vlan1
STP FAST: sending RLQ request PDU on VLAN0001(1) Fa0/15 Vlan1
STP FAST: received inferior BPDU on VLAN0005 FastEthernet0/19.
STP FAST: sending RLQ request PDU on VLAN0005(5) Faa0/13 Vlan5
STP FAST: sending RLQ request PDU on VLAN0005(5) Fa0/14 Vlan5
STP FAST: sending RLQ request PDU on VLAN0005(5) Fa0/15 Vlan5
STP FAST: sending RLQ request PDU on VLAN0005(5) Fa0/20 Vlan5
STP FAST: sending RLQ request PDU on VLAN0005(5) Fa0/21 Vlan5
STP FAST: received inferior BPDU on VLAN0005 FastEthernet0/20.
STP FAST: sending RLQ request PDU on VLAN0005(5) Fa0/13 Vlan5
STP FAST: sending RLQ request PDU on VLAN0005(5) Fa0/14 Vlan5
STP FAST: sending RLQ request PDU on VLAN0005(5) Fa0/15 Vlan5
STP FAST: sending RLQ request PDU on VLAN0005(5) Fa0/21 Vlan5
STP FAST: received inferior BPDU on VLAN0005 FastEthernet0/21.
STP FAST: sending RLQ request PDU on VLAN0005(5) Fa0/13 Vlan5
STP FAST: sending RLQ request PDU on VLAN0005(5) Fa0/14 Vlan5
STP FAST: sending RLQ request PDU on VLAN0005(5) Fa0/15 Vlan5
<output omitted>
```

Copyright © 2010 Internetwork Expert

1.26 STP BPDU Guard

- Configure Spanning-Tree BPDU Guard on the switches so that ports connected to the internal and external routers are disabled if a Spanning-Tree BPDU is detected.
- Once disabled the switches should attempt to re-enable the ports after two minutes.
- Do not use the global **portfast** command to accomplish this.

Configuration

```
SW1:
interface FastEthernet0/1
 spanning-tree bpduguard enable
!
interface FastEthernet0/5
 spanning-tree bpduguard enable
!
errdisable recovery cause bpduguard
errdisable recovery interval 120

SW2:
interface FastEthernet0/2
 spanning-tree bpduguard enable
!
interface FastEthernet0/4
 spanning-tree bpduguard enable
!
interface FastEthernet0/6
 spanning-tree bpduguard enable
!
interface FastEthernet0/24
 spanning-tree bpduguard enable
!
errdisable recovery cause bpduguard
errdisable recovery interval 120

SW3:
interface FastEthernet0/5
 spanning-tree bpduguard enable
!
interface FastEthernet0/24
 spanning-tree bpduguard enable
!
errdisable recovery cause bpduguard
errdisable recovery interval 120

SW4:
interface FastEthernet0/4
 spanning-tree bpduguard enable
!
errdisable recovery cause bpduguard
errdisable recovery interval 120
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> The STP BPDU Guard feature is used to enforce access layer security on the termination of the STP domain. When an interface running BPDU Guard receives a BPDU (STP packet), the interface is transitioned into err-disable state. This ensures that unauthorized switches cannot be plugged into the network, for example, to perform a layer 2 man-in-the-middle (MiM) attack. If configured, the **`errdisable recovery`** feature can then be used to bring the interface out of err-disable state automatically after a configured interval.
>
> By configuring bridging on R1's link to SW1, STP BPDUs are generated and the link is sent to err-disable state.

```
Rack1R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#bridge 1 protocol ieee
Rack1R1(config)#interface fa0/0
Rack1R1(config-if)#bridge-group 1

Rack1SW1#show spanning-tree interface fa0/1 detail
 Port 3 (FastEthernet0/1) of VLAN0146 is forwarding
   Port path cost 19, Port priority 128, Port Identifier 128.3.
   Designated root has priority 146, address 001b.d490.7c00
   Designated bridge has priority 146, address 001b.d490.7c00
   Designated port id is 128.3, designated path cost 0
   Timers: message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is point-to-point by default
   Bpdu guard is enabled
   BPDU: sent 4500, received 0

Rack1SW1#
09:00:09: %SPANTREE-2-BLOCK_BPDUGUARD: Received BPDU on port FastEthernet0/1
with BPDU Guard enabled. Disabling port.
09:00:09: %PM-4-ERR_DISABLE: bpduguard error detected on Fa0/1, putting Fa0/1
in err-disable state
09:00:10: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1,
changed state to down
09:02:09: %PM-4-ERR_RECOVER: Attempting to recover from bpduguard err-disable
state on Fa0/1
09:02:12: %SPANTREE-2-BLOCK_BPDUGUARD: Received BPDU on port FastEthernet0/1
with BPDU Guard enabled. Disabling port.
09:02:12: %PM-4-ERR_DISABLE: bpduguard error detected on Fa0/1, putting Fa0/1
in err-disable state

Rack1SW1#show interface fa0/1 status

Port      Name               Status       Vlan       Duplex  Speed Type
Fa0/1                        err-disabled 146          auto   auto 10/100BaseTX
```

Copyright © 2010 Internetwork Expert

1.27 STP BPDU Guard Default

- Remove the previous BPDU Guard configuration.
- Configure Spanning-Tree PortFast on the switches so that ports connected to the internal and external routers do not have to wait for the Spanning-Tree listening and learning phases to begin forwarding.
- Configure Spanning-Tree BPDU Guard so that if a Spanning-Tree BPDU is detected on any of these ports they are disabled.
- Do not use any interface level Spanning-Tree commands to accomplish this.

Configuration

```
SW1:
spanning-tree portfast bpduguard default
spanning-tree portfast default

SW2:
spanning-tree portfast bpduguard default
spanning-tree portfast default

SW3:
spanning-tree portfast bpduguard default
spanning-tree portfast default

SW4:
spanning-tree portfast bpduguard default
spanning-tree portfast default
```

Verification

> **Note**
>
> The BPDU Guard default feature works in conjunction with Portfast default in order to automatically enable BPDU Guard on any interfaces in the Portfast state.

```
Rack1R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#bridge 1 protocol ieee
Rack1R1(config)#interface fa0/0
Rack1R1(config-if)#bridge-group 1

Rack1SW1#
09:07:57: %SPANTREE-2-BLOCK_BPDUGUARD: Received BPDU on port
FastEthernet0/1 with BPDU Guard enabled. Disabling port.
09:07:57: %PM-4-ERR_DISABLE: bpduguard error detected on Fa0/1, putting
Fa0/1 in err-disable state
09:07:57: %LINK-3-UPDOWN: Interface FastEthernet0/1, changed state to
down
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.28 STP BPDU Filter

- Remove the previous BPDU Guard configuration.
- Configure the switches so that ports connected to the internal and external routers do not send Spanning-Tree packets sent out them.
- Do not use any global Spanning-Tree commands to accomplish this.

Configuration

```
SW1:
interface FastEthernet0/1
 spanning-tree bpdufilter enable
!
interface FastEthernet0/5
 spanning-tree bpdufilter enable

SW2:
interface FastEthernet0/2
 spanning-tree bpdufilter enable
!
interface FastEthernet0/4
 spanning-tree bpdufilter enable
!
interface FastEthernet0/6
 spanning-tree bpdufilter enable
!
interface FastEthernet0/24
 spanning-tree bpdufilter enable

SW3:
interface FastEthernet0/5
 spanning-tree bpdufilter enable
!
interface FastEthernet0/24
 spanning-tree bpdufilter enable

SW4:
interface FastEthernet0/4
 spanning-tree bpdufilter enable
```

Copyright © 2010 Internetwork Expert

Verification

Note

The BPDU Filter feature, like the BPDU Guard feature, is used to terminate the STP domain. The difference between them is that when configured at the interface level the BPDU Filter feature drops all inbound BPDUs and does not send BPDUs out the interface. Unlike BPDU Guard the interface does not go into err-disable when a violation occurs. Other user traffic will continued to be forwarded inbound and outbound the port.

```
Rack1R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#bridge 1 protocol ieee
Rack1R1(config)#interface fa0/0
Rack1R1(config-if)#bridge-group 1
Rack1R1(config-if)#end
```

R1 is configured to bridge on the Fa0/0 interface and 2 BPDUs are sent.

```
Rack1R1#show spanning-tree 1

 Bridge group 1 is executing the ieee compatible Spanning Tree protocol
  Bridge Identifier has priority 1, address 0011.bbbd.3bc0
  Configured hello time 2, max age 20, forward delay 15
  We are the root of the spanning tree
  Topology change flag set, detected flag set
  Number of topology changes 3 last change occurred 00:00:23 ago
          from FastEthernet0/0
  Times:  hold 1, topology change 35, notification 2
          hello 2, max age 20, forward delay 15
  Timers: hello 0, topology change 12, notification 0, aging 15

 Port 4 (FastEthernet0/0) of Bridge group 1 is listening
   Port path cost 19, Port priority 128, Port Identifier 128.4.
   Designated root has priority 1, address 0011.bbbd.3bc0
   Designated bridge has priority 1, address 0011.bbbd.3bc0
   Designated port id is 128.4, designated path cost 0
   Timers: message age 0, forward delay 10, hold 0
   Number of transitions to forwarding state: 0
   BPDU: sent 2, received 0
```

Copyright © 2010 Internetwork Expert

SW1 does not acknowledge that it received these BPDUs because BPDU Filter is configured.

```
Rack1SW1#show spanning-tree interface fa0/1 detail
 Port 3 (FastEthernet0/1) of VLAN0146 is forwarding
   Port path cost 19, Port priority 128, Port Identifier 128.3.
   Designated root has priority 146, address 001b.d490.7c00
   Designated bridge has priority 146, address 001b.d490.7c00
   Designated port id is 128.3, designated path cost 0
   Timers: message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is point-to-point by default
   Bpdu filter is enabled
   BPDU: sent 0, received 0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.29 STP BPDU Filter Default

- Remove the previous BPDU Filter configuration.
- Configure Spanning-Tree PortFast on the switches so that ports connected to the internal and external routers do not have to wait for the Spanning-Tree listening and learning phases to begin forwarding.
- Configure Spanning-Tree BPDU Filter on the switches so that the PortFast enabled ports are reverted out of PortFast state if a Spanning-Tree packet is received in them.
- Do not use any interface level Spanning-Tree commands to accomplish this.

Configuration

```
SW1:
spanning-tree portfast bpdufilter default
spanning-tree portfast default

SW2:
spanning-tree portfast bpdufilter default
spanning-tree portfast default

SW3:
spanning-tree portfast bpdufilter default
spanning-tree portfast default

SW4:
spanning-tree portfast bpdufilter default
spanning-tree portfast default
```

Copyright © 2010 Internetwork Expert

Verification

Note

BPDU Filter Default works with Portfast default by allowing interfaces that should not have Portfast enabled on them to be automatically detected. When both features are configured together all interfaces run in Portfast mode except those which are receiving BPDUs.

In the below output we can see that Portfast is enabled on SW1's link Fa0/1 to R1. Once bridging is enabled on R1's link to SW1, SW1 detects that R1 is sending BPDUs and reverts the interface out of Portfast state. Note that the interface can still forward traffic and is not sent into err-disable state.

```
Rack1SW1#show spanning-tree interface fa0/1 portfast
VLAN0146         enabled

Rack1R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#bridge 1 protocol ieee
Rack1R1(config)#interface fa0/0
Rack1R1(config-if)#bridge-group 1
Rack1R1(config-if)#end
Rack1R1#

Rack1SW1#show spanning-tree interface fa0/1 portfast
VLAN0146         disabled
```

Copyright © 2010 Internetwork Expert

1.30 STP Root Guard

- Configure SW1 so that the links to either SW2 or SW3 are disabled if SW2, SW3, or SW4 is elected the Spanning-Tree Root Bridge for any VLAN.

Configuration

```
SW1:
interface FastEthernet0/13
 spanning-tree guard root
!
interface FastEthernet0/14
 spanning-tree guard root
!
interface FastEthernet0/15
 spanning-tree guard root
!
interface FastEthernet0/16
 spanning-tree guard root
!
interface FastEthernet0/17
 spanning-tree guard root
!
interface FastEthernet0/18
 spanning-tree guard root
```

Copyright © 2010 Internetwork Expert

Verification

Note

Root Guard is similar to the BPDU Guard feature in the manner that it is used to detect STP packets and disable the interface they were received on. The difference between them is that with Root Guard the interface is only disabled (via root inconsistent state) if a *superior* BPDU is received. A superior BPDU indicates a better cost to the root bridge than what is currently installed. Therefore design-wise this feature is used to prevent a rogue device from announcing itself as the new root bridge and possibly implementing a layer 2 man-in-the-middle attack.

In the below output SW4 starts announcing superior BPDUs to SW1 by lowering its bridge priority to zero. Once SW1 receives these announcements the forwarding of VLAN 1 is disabled on the links that these BPDUs were received.

```
Rack1SW4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW4(config)#spanning-tree vlan 1 priority 0
Rack1SW4(config)#

Rack1SW1#
09:20:23: %SPANTREE-2-ROOTGUARD_BLOCK: Root guard blocking port
FastEthernet0/13 on VLAN0001.

Rack1SW1#show spanning-tree vlan 1

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    1
              Address     001b.d490.7c00
              This bridge is the root
              Hello Time   3 sec  Max Age 10 sec  Forward Delay 10 sec

  Bridge ID   Priority    1      (priority 0 sys-id-ext 1)
              Address     001b.d490.7c00
              Hello Time   3 sec  Max Age 10 sec  Forward Delay 10 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/5            Desg FWD 19        128.7    P2p
Fa0/13           Desg BKN*19        128.15   P2p *ROOT_Inc
Fa0/14           Desg BKN*19        128.16   P2p *ROOT_Inc
Fa0/15           Desg BKN*19        128.17   P2p *ROOT_Inc
Fa0/16           Desg BKN*19        128.18   P2p *ROOT_Inc
Fa0/17           Desg BKN*19        128.19   P2p *ROOT_Inc
Fa0/18           Desg BKN*19        128.20   P2p *ROOT_Inc
```

Copyright © 2010 Internetwork Expert

1.31 STP Loop Guard

- Configure Spanning-Tree Loop Guard to prevent unidirectional links from forming on any of the inter-switch links in the layer 2 network.

Configuration

```
SW1:
interface FastEthernet0/13
 spanning-tree guard loop
!
interface FastEthernet0/14
 spanning-tree guard loop
!
interface FastEthernet0/15
 spanning-tree guard loop
!
interface FastEthernet0/16
 spanning-tree guard loop
!
interface FastEthernet0/17
 spanning-tree guard loop
!
interface FastEthernet0/18
 spanning-tree guard loop

SW2:
interface FastEthernet0/13
 spanning-tree guard loop
!
interface FastEthernet0/14
 spanning-tree guard loop
!
interface FastEthernet0/15
 spanning-tree guard loop
!
interface FastEthernet0/19
 spanning-tree guard loop
!
interface FastEthernet0/20
 spanning-tree guard loop
!
interface FastEthernet0/21
 spanning-tree guard loop

SW3:
interface FastEthernet0/13
 spanning-tree guard loop
!
interface FastEthernet0/14
 spanning-tree guard loop
!
interface FastEthernet0/15
 spanning-tree guard loop
!
interface FastEthernet0/19
```

Copyright © 2010 Internetwork Expert

```
 spanning-tree guard loop
!
interface FastEthernet0/20
 spanning-tree guard loop
!
interface FastEthernet0/21
 spanning-tree guard loop

SW4:
interface FastEthernet0/16
 spanning-tree guard loop
!
interface FastEthernet0/17
 spanning-tree guard loop
!
interface FastEthernet0/18
 spanning-tree guard loop
!
interface FastEthernet0/19
 spanning-tree guard loop
!
interface FastEthernet0/20
 spanning-tree guard loop
!
interface FastEthernet0/21
 spanning-tree guard loop
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> STP Loop Guard is used to prevent STP loops from occurring due to unidirectional links. This feature is similar to Unidirectional Link Detection (UDLD), but it uses STP BPDU keepalives to determine if there is a unidirectional link.
>
> In normal STP operation in a redundant topology some links will be designated forwarding while the other end will be blocking. If one of these blocking links transitions to forwarding state erroneously, a loop can occur. Specifically this can happen if there is a unidirectional link and the blocking port stops receiving the BPDUs that the designated port it sending. Loop guard prevents this by transitioning blocking ports into loop-inconsistent state instead of forwarding if BPDUs stop being received from the designated port.

```
Rack1SW1#show spanning-tree interface fa0/13 detail
 Port 15 (FastEthernet0/13) of VLAN0001 is blocking
   Port path cost 19, Port priority 128, Port Identifier 128.15.
   Designated root has priority 1, address 001b.d490.7c00
   Designated bridge has priority 1, address 001b.d490.7c00
   Designated port id is 128.15, designated path cost 0
   Timers: message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is point-to-point by default
   Loop guard is enabled on the port
   BPDU: sent 193, received 109

 Port 15 (FastEthernet0/13) of VLAN0005 is forwarding
   Port path cost 19, Port priority 128, Port Identifier 128.15.
   Designated root has priority 5, address 001b.d490.7c00
   Designated bridge has priority 5, address 001b.d490.7c00
   Designated port id is 128.15, designated path cost 0
   Timers: message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is point-to-point by default
   Loop guard is enabled on the port
   BPDU: sent 268, received 0

<ouput omitted>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.32 Unidirectional Link Detection

- Remove the previous Loop Guard configuration.
- Configure UDLD to prevent unidirectional links from forming on any of the inter-switch links in the layer 2 network.

Configuration

```
SW1:
interface FastEthernet0/13
 udld port aggressive
!
interface FastEthernet0/14
 udld port aggressive
!
interface FastEthernet0/15
 udld port aggressive
!
interface FastEthernet0/16
 udld port aggressive
!
interface FastEthernet0/17
 udld port aggressive
!
interface FastEthernet0/18
 udld port aggressive

SW2:
interface FastEthernet0/13
 udld port aggressive
!
interface FastEthernet0/14
 udld port aggressive
!
interface FastEthernet0/15
 udld port aggressive
!
interface FastEthernet0/19
 udld port aggressive
!
interface FastEthernet0/20
 udld port aggressive
!
interface FastEthernet0/21
 udld port aggressive

SW3:
interface FastEthernet0/13
 udld port aggressive
!
interface FastEthernet0/14
 udld port aggressive
!
interface FastEthernet0/15
 udld port aggressive
```

Copyright © 2010 Internetwork Expert

```
!
interface FastEthernet0/19
 udld port aggressive
!
interface FastEthernet0/20
 udld port aggressive
!
interface FastEthernet0/21
 udld port aggressive

SW4:
interface FastEthernet0/16
 udld port aggressive
!
interface FastEthernet0/17
 udld port aggressive
!
interface FastEthernet0/18
 udld port aggressive
!
interface FastEthernet0/19
 udld port aggressive
!
interface FastEthernet0/20
 udld port aggressive
!
interface FastEthernet0/21
 udld port aggressive
```

Copyright © 2010 Internetwork Expert

Verification

Note

UDLD, like Loop Guard, is used to prevent loops due to unidirectional links. The difference between the features is that Loop Guard uses STP BPDUs to detect these failures, while UDLD uses its own keepalive.

UDLD is a Cisco proprietary feature in which peers discover each other by exchanging frames sent to the well-known MAC address 01:00:0C:CC:CC:CC. Each switch sends its own device ID along with the originator port ID and timeout value to its peer. Additionally a switch echoes back the ID of its neighbor. If no echo frame with the switch's own ID has been seen from the peer for a certain amount of time, the port is suspected to be unidirectional. What happens next depends on UDLD mode of operation.

In "Normal" mode if the physical state of port (as reported by Layer 1) is still up UDLD marks this port as "Undetermined", but does NOT shut down or disable the port, and it continues to operate under its current STP status. This mode of operation is informational and potentially less disruptive (though it does not prevent STP loops).

If UDLD is set to "Aggressive" mode, once the switch loses its neighbor it actively tries to re-establish the relationship by sending a UDLD frames 8 times every 1 second. If the neighbor does not respond after that the port is considered to be unidirectional and sent to err-disable state.

In certain designs there are unidirectional links that Loop Guard can prevent, and UDLD can not, and likewise ones that UDLD can prevent, but Loop Guard cannot. For example if a loop occurs due to a physical wiring problem, i.e. someone mistakenly mixes up the send and receive pairs of a fiber link, UDLD can detect this, but Loop Guard cannot. Likewise if there is a unidirectional link due to a failure in the STP software itself, although much more rare, Loop Guard can detect this but UDLD cannot. Based on this the features can be configured at the same time to protect against all possible unidirectional link scenarios.

Copyright © 2010 Internetwork Expert

Although in this design UDLD is configured on copper UTP interfaces, this case is usually not needed in a real network design due to the Fast Link Pulse (FLP) signals that already track the interface status on wired interfaces. Instead UDLD is more commonly run on Fiber Optic interfaces.

```
Rack1SW1#show udld fa0/13

Interface Fa0/13
---
Port enable administrative configuration setting: Enabled / in
aggressive mode
Port enable operational state: Enabled / in aggressive mode
Current bidirectional state: Bidirectional
Current operational state: Advertisement - Single neighbor detected
Message interval: 7
Time out interval: 5

    Entry 1
    ---
    Expiration time: 45
    Device ID: 1
    Current neighbor state: Bidirectional
    Device name: FDO1118Z0P9
    Port ID: Fa0/13
    Neighbor echo 1 device: FDO1118Z0P6
    Neighbor echo 1 port: Fa0/13

    Message interval: 15
    Time out interval: 5
    CDP Device name: Rack1SW2
```

Copyright © 2010 Internetwork Expert

1.33 MST Root Bridge Election

- Configure the inter-switch links between SW1 & SW2, SW1 & SW3, SW2 & SW4, and SW3 & SW4 as 802.1q trunk links.
- Disable all other inter-switch links.
- Configure SW4 as a VTP server using the domain name CCIE with SW1, SW2, and SW3 as its clients.
- Configure VLAN assignments per the diagram.
- Configure Multiple Spanning-Tree on the switches.
- Instance 1 should service VLANs 1 - 100.
- Instance 2 should service VLANs 101 - 200.
- Instance 3 should service all other VLANs.
- Configure SW1 as the STP Root Bridge for instance 1.
- Configure SW4 as the STP Root Bridge for instance 2.
- If SW1 goes down SW2 should take over as the STP Root Bridge for instance 1.
- If SW4 goes down SW3 should take over as the STP Root Bridge for instance 2.

Configuration

```
R6:
interface FastEthernet0/0.67
 encapsulation dot1q 67
 ip address 155.1.67.6 255.255.255.0
!
interface FastEthernet0/0.146
 encapsulation dot1q 146
 ip address 155.1.146.6 255.255.255.0

SW1:
vtp domain CCIE
vtp mode client
!
spanning-tree mst configuration
 name MST1
 revision 1
 instance 1 vlan 1-100
 instance 2 vlan 101-200
 instance 3 vlan 201-4094
!
spanning-tree mst 1 priority 0
!
spanning-tree mode mst
!
interface FastEthernet0/1
 switchport access vlan 146
!
```

Copyright © 2010 Internetwork Expert

```
interface FastEthernet0/5
 switchport access vlan 58
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/14
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/15
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/17
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/18
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/19
 shutdown
!
interface FastEthernet0/20
 shutdown
!
interface FastEthernet0/21
 shutdown

SW2:
vtp domain CCIE
vtp mode client
!
spanning-tree mst configuration
 name MST1
 revision 1
 instance 1 vlan 1-100
 instance 2 vlan 101-200
 instance 3 vlan 201-4094
!
spanning-tree mst 1 priority 4096
!
spanning-tree mode mst
!
interface FastEthernet0/2
 switchport access vlan 22
!
interface FastEthernet0/4
 switchport access vlan 43
```

Copyright © 2010 Internetwork Expert

```
interface FastEthernet0/6
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/14
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/15
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/16
 shutdown
!
interface FastEthernet0/17
 shutdown
!
interface FastEthernet0/18
 shutdown
!
interface FastEthernet0/19
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/20
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/21
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/24
 switchport access vlan 22

SW3:
vtp domain CCIE
vtp mode client
!
spanning-tree mst configuration
 name MST1
 revision 1
 instance 1 vlan 1-100
 instance 2 vlan 101-200
 instance 3 vlan 201-4094
!
spanning-tree mst 2 priority 4096
!
```

Copyright © 2010 Internetwork Expert

```
spanning-tree mode mst
!
interface FastEthernet0/5
 switchport access vlan 5
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/14
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/15
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/16
 shutdown
!
interface FastEthernet0/17
 shutdown
!
interface FastEthernet0/18
 shutdown
!
interface FastEthernet0/19
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/20
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/21
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/24
switchport access vlan 43
```

Copyright © 2010 Internetwork Expert

```
SW4:
vtp domain CCIE
vlan 5,7,8,9,10,22,43,58,67,79,146
!
spanning-tree mst configuration
 name MST1
 revision 1
 instance 1 vlan 1-100
 instance 2 vlan 101-200
 instance 3 vlan 201-4094
!
spanning-tree mst 2 priority 0
!
spanning-tree mode mst
!
interface FastEthernet0/4
 switchport access vlan 146
!
interface FastEthernet0/13
 shutdown
!
interface FastEthernet0/14
 shutdown
!
interface FastEthernet0/15
 shutdown
!
interface FastEthernet0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/17
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/18
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/19
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/20
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/21
 switchport trunk encapsulation dot1q
 switchport mode trunk
```

Copyright © 2010 Internetwork Expert

Verification

Note

Multiple Spanning-Tree (MST) is an IEEE standard defined in 802.1s, and allows user-defined STP instances to be mapped to multiple VLANs. Unlike the Cisco proprietary Per-VLAN Spanning-Tree (PVST), MST can be used to eliminate the overhead of redundant STP instances in topologies where multiple VLANs, but not all VLANs, follow the same layer 2 forwarding path, while at the same time allowing for flexible failure domain separation and traffic engineering. MST essentially takes the best features of IEEE 802.1D Spanning-Tree, AKA Common Spanning-Tree, and the Cisco extensions to STP, PVST, PVST+, Rapid PVST+, and combines them.

For example in this design STP instances are created for VLANs 1 – 4094. In Common Spanning-Tree all 4094 VLANs would map to one instance. This has very little overhead but does not allow for detailed traffic engineering. With PVST there would be 4094 separate instances of STP, which allows for detailed traffic engineering but creates immense overhead. With MST three user-defined instances are created that map different portions of the VLAN space into separate instances with a similar forwarding path.

Like CST and PVST, MST uses the lowest Bridge-ID (BID) in the network to elect the Root Bridge. The BID is made up of the priority value and the MAC address. The lower priority wins the election, and if there is a tie in priority the lowest MAC address is the tie breaker. In PVST there is one root bridge election per VLAN, since there is one STP instance per VLAN, but in MST there is one election per user-defined instance.

From the **`show spanning-tree mst`** output we can see which VLANs are mapped to the particular MST instance, who the root bridge is, and how the root port election has occurred. In this case SW1 is the root for instance 1, while SW4 is the root for instance 2. SW1 is the root for instance 1 because it has a priority value of 1, which is made up of the configured priority of 0 plus the sytem-id extension of 1. In MST the sysid field is the instance number, where as in PVST the sysid is the VLAN number.

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show spanning-tree mst 1

##### MST1    vlans mapped:   1-100
Bridge        address 001b.d490.7c00  priority      1     (0 sysid 1)
Root          this switch for MST1

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/5            Desg FWD 200000    128.7    P2p
Fa0/13           Desg FWD 200000    128.15   P2p
Fa0/14           Desg FWD 200000    128.16   P2p
Fa0/15           Desg FWD 200000    128.17   P2p
Fa0/16           Desg FWD 200000    128.18   P2p
Fa0/17           Desg FWD 200000    128.19   P2p
Fa0/18           Desg FWD 200000    128.20   P2p

Rack1SW1#show spanning-tree mst 2

##### MST2    vlans mapped:   101-200
Bridge        address 001b.d490.7c00  priority      32770 (32768 sysid 2)
Root          address 000c.3045.d600  priority      2     (0 sysid 2)
              port    Fa0/16          cost          400000    rem hops 18

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1            Desg FWD 200000    128.3    P2p
Fa0/13           Altn BLK 200000    128.15   P2p
Fa0/14           Altn BLK 200000    128.16   P2p
Fa0/15           Altn BLK 200000    128.17   P2p
Fa0/16           Root FWD 200000    128.18   P2p
Fa0/17           Altn BLK 200000    128.19   P2p
Fa0/18           Altn BLK 200000    128.20   P2p
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show spanning-tree mst 1

##### MST1    vlans mapped:   1-100
Bridge        address 001b.d4df.ec80  priority      4097  (4096 sysid 1)
Root          address 001b.d490.7c00  priority      1     (0 sysid 1)
              port    Fa0/13          cost          200000    rem hops 19

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/2            Desg FWD 200000    128.4    P2p
Fa0/4            Desg FWD 200000    128.6    P2p
Fa0/6            Desg FWD 200000    128.8    P2p
Fa0/13           Root FWD 200000    128.15   P2p
Fa0/14           Altn BLK 200000    128.16   P2p
Fa0/15           Altn BLK 200000    128.17   P2p
Fa0/19           Desg FWD 200000    128.21   P2p
Fa0/20           Desg FWD 200000    128.22   P2p
Fa0/21           Desg FWD 200000    128.23   P2p
Fa0/24           Desg FWD 2000000   128.26   Shr

Rack1SW2#show spanning-tree mst 2

##### MST2    vlans mapped:   101-200
Bridge        address 001b.d4df.ec80  priority      32770 (32768 sysid 2)
Root          address 000c.3045.d600  priority      2     (0 sysid 2)
              port    Fa0/19          cost          200000    rem hops 19

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/6            Desg FWD 200000    128.8    P2p
Fa0/13           Desg FWD 200000    128.15   P2p
Fa0/14           Desg FWD 200000    128.16   P2p
Fa0/15           Desg FWD 200000    128.17   P2p
Fa0/19           Root FWD 200000    128.21   P2p
Fa0/20           Altn BLK 200000    128.22   P2p
Fa0/21           Altn BLK 200000    128.23   P2p

Rack1SW3#show spanning-tree mst 1

##### MST1    vlans mapped:   1-100
Bridge        address 000c.3045.4180  priority      32769 (32768 sysid 1)
Root          address 001b.d490.7c00  priority      1     (0 sysid 1)
              port    Fa0/13          cost          200000    rem hops 19

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/5            Desg FWD 200000    128.5    P2p
Fa0/13           Root FWD 200000    128.13   P2p
Fa0/14           Altn BLK 200000    128.14   P2p
Fa0/15           Altn BLK 200000    128.15   P2p
Fa0/19           Desg FWD 200000    128.19   P2p
Fa0/20           Desg FWD 200000    128.20   P2p
Fa0/21           Desg FWD 200000    128.21   P2p
Fa0/24           Desg FWD 2000000   128.24   Shr
```

Copyright © 2010 Internetwork Expert

```
Rack1SW3#show spanning-tree mst 2

##### MST2     vlans mapped:   101-200
Bridge         address 000c.3045.4180  priority      4098  (4096 sysid 2)
Root           address 000c.3045.d600  priority      2     (0 sysid 2)
               port    Fa0/19          cost          200000    rem hops 19

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/13           Desg FWD 200000    128.13   P2p
Fa0/14           Desg FWD 200000    128.14   P2p
Fa0/15           Desg FWD 200000    128.15   P2p
Fa0/19           Root FWD 200000    128.19   P2p
Fa0/20           Altn BLK 200000    128.20   P2p
Fa0/21           Altn BLK 200000    128.21   P2p

Rack1SW4#show spanning-tree mst 1

##### MST1     vlans mapped:   1-100
Bridge         address 000c.3045.d600  priority      32769 (32768 sysid 1)
Root           address 001b.d490.7c00  priority      1     (0 sysid 1)
               port    Fa0/16          cost          400000    rem hops 18

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/16           Root FWD 200000    128.16   P2p
Fa0/17           Altn BLK 200000    128.17   P2p
Fa0/18           Altn BLK 200000    128.18   P2p
Fa0/19           Altn BLK 200000    128.19   P2p
Fa0/20           Altn BLK 200000    128.20   P2p
Fa0/21           Altn BLK 200000    128.21   P2p

Rack1SW4#show spanning-tree mst 2

##### MST2     vlans mapped:   101-200
Bridge         address 000c.3045.d600  priority      2     (0 sysid 2)
Root           this switch for MST2

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/4            Desg FWD 200000    128.4    P2p
Fa0/16           Desg FWD 200000    128.16   P2p
Fa0/17           Desg FWD 200000    128.17   P2p
Fa0/18           Desg FWD 200000    128.18   P2p
Fa0/19           Desg FWD 200000    128.19   P2p
Fa0/20           Desg FWD 200000    128.20   P2p
Fa0/21           Desg FWD 200000    128.21   P2p
```

Copyright © 2010 Internetwork Expert

For MST instance 1 SW1 has a priority of 1, and SW2 is next in line with a priority of 4097. When connectivity to SW1 is lost SW2 is promoted to the root bridge status.

```
Rack1SW1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface range fa0/13 - 18
Rack1SW1(config-if-range)#shut
Rack1SW1(config-if-range)#

Rack1SW2#show spanning-tree mst 1

##### MST1    vlans mapped:   1-100
Bridge        address 001b.d4df.ec80  priority      4097  (4096 sysid 1)
Root          this switch for MST1

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/2            Desg BLK 200000    128.4    P2p
Fa0/4            Desg BLK 200000    128.6    P2p
Fa0/6            Desg BLK 200000    128.8    P2p
Fa0/19           Desg FWD 200000    128.21   P2p
Fa0/20           Desg FWD 200000    128.22   P2p
Fa0/21           Desg FWD 200000    128.23   P2p
Fa0/24           Desg BLK 2000000   128.26   Shr
```

Copyright © 2010 Internetwork Expert

1.34 MST Load Balancing with Port Cost

- Using Spanning-Tree cost modify the layer 2 transit network so that traffic for MST instance 1 from SW2 to SW1 uses the last link between SW2 and SW4.
- If this link goes down traffic should fall over to the second link between SW2 and SW4.

Configuration

```
SW2:
interface FastEthernet0/13
 spanning-tree mst 1 cost 500000
!
interface FastEthernet0/14
 spanning-tree mst 1 cost 500000
!
interface FastEthernet0/15
 spanning-tree mst 1 cost 500000
!
interface FastEthernet0/20
 spanning-tree mst 1 cost 2
!
interface FastEthernet0/21
 spanning-tree mst 1 cost 1
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Similar to CST and PVST, MST uses a cost value derived from the inverse bandwidth of the interface (higher bandwidth means lower cost). The root port is chosen based on the lowest end-to-end cost to the root bridge. The **show spanning-tree mst** command shows the local cost values of the outgoing ports on the local switch.

```
Rack1SW2#show spanning-tree mst 1

##### MST1    vlans mapped:   1-100
Bridge        address 001b.d4df.ec80  priority      4097  (4096 sysid 1)
Root          address 001b.d490.7c00  priority      1     (0 sysid 1)
              port    Fa0/21          cost          400001    rem hops 17

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/2            Desg FWD 200000    128.4    P2p
Fa0/4            Desg FWD 200000    128.6    P2p
Fa0/6            Desg FWD 200000    128.8    P2p
Fa0/13           Altn BLK 500000    128.15   P2p
Fa0/14           Altn BLK 500000    128.16   P2p
Fa0/15           Altn BLK 500000    128.17   P2p
Fa0/19           Altn BLK 200000    128.21   P2p
Fa0/20           Altn BLK 2         128.22   P2p
Fa0/21           Root FWD 1         128.23   P2p
Fa0/24           Desg FWD 2000000   128.26   Shr
```

Copyright © 2010 Internetwork Expert

To see the entire end-to-end cost of a path the **`show spanning-tree mst detail`** command should be used. The end-to-end cost is made up of the upstream (designated) cost, plus the local port cost.In this output the alternate ports Fa0/13 – Fa0/15 have a total cost of 500,000 due to the manual cost change. Fa0/20 has a total cost of 600,000, which is 200,000 to SW4, 200,000 from SW4 to SW3, and 200,000 from SW3 to SW1. Fa0/20 has a total cost of 400,002, which is 2 to SW4, 200,000 from SW4 to SW3, and 200,000 from SW3 to SW1. Fa0/21 wins the root port election since it has a total cost of 400,001.

```
Rack1SW2#show spanning-tree mst 1 detail

##### MST1    vlans mapped:   1-100
Bridge        address 001b.d4df.ec80  priority      4097  (4096 sysid 1)
Root          address 001b.d490.7c00  priority      1     (0 sysid 1)
              port    Fa0/21          cost          400001    rem hops 17

<output omitted>

FastEthernet0/13 of MST1 is alternate blocking
Port info             port id         128.15  priority    128  cost      500000
Designated root       address 001b.d490.7c00  priority      1  cost           0
Designated bridge     address 001b.d490.7c00  priority      1  port id   128.15
Timers: message expires in 5 sec, forward delay 0, forward transitions 3
Bpdus (MRecords) sent 1385, received 844

FastEthernet0/14 of MST1 is alternate blocking
Port info             port id         128.16  priority    128  cost      500000
Designated root       address 001b.d490.7c00  priority      1  cost           0
Designated bridge     address 001b.d490.7c00  priority      1  port id   128.16
Timers: message expires in 4 sec, forward delay 0, forward transitions 0
Bpdus (MRecords) sent 3965, received 4719

FastEthernet0/15 of MST1 is alternate blocking
Port info             port id         128.17  priority    128  cost      500000
Designated root       address 001b.d490.7c00  priority      1  cost           0
Designated bridge     address 001b.d490.7c00  priority      1  port id   128.17
Timers: message expires in 5 sec, forward delay 0, forward transitions 0
Bpdus (MRecords) sent 3971, received 4725

FastEthernet0/19 of MST1 is alternate blocking
Port info             port id         128.21  priority    128  cost      200000
Designated root       address 001b.d490.7c00  priority      1  cost      400000
Designated bridge     address 000c.3045.d600  priority  32769  port id   128.16
Timers: message expires in 5 sec, forward delay 0, forward transitions 5
Bpdus (MRecords) sent 960, received 1011

FastEthernet0/20 of MST1 is alternate blocking
Port info             port id         128.22  priority    128  cost           2
Designated root       address 001b.d490.7c00  priority      1  cost      400000
Designated bridge     address 000c.3045.d600  priority  32769  port id   128.17
Timers: message expires in 4 sec, forward delay 0, forward transitions 3
Bpdus (MRecords) sent 6085, received 6086

FastEthernet0/21 of MST1 is root forwarding
Port info             port id         128.23  priority    128  cost           1
Designated root       address 001b.d490.7c00  priority      1  cost      400000
Designated bridge     address 000c.3045.d600  priority  32769  port id   128.18
<output omitted>
```

Copyright © 2010 Internetwork Expert

When SW2's port Fa0/21 is down the next lowest cost path is 400,002 through Fa0/20.

```
Rack1SW2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW2(config)#interface fa0/21
Rack1SW2(config-if)#shut
Rack1SW2(config-if)#end
Rack1SW2#
07:22:04: %LINK-5-CHANGED: Interface FastEthernet0/21, changed state to
administratively down
07:22:04: %SYS-5-CONFIG_I: Configured from console by console
07:22:05: %LINEPROTO-5-UPDOWN: Line protocol on Interface
FastEthernet0/21, changed state to down

Rack1SW2#show spanning-tree mst 1

##### MST1    vlans mapped:   1-100
Bridge        address 001b.d4df.ec80  priority      4097  (4096 sysid 1)
Root          address 001b.d490.7c00  priority      1     (0 sysid 1)
              port    Fa0/20          cost          400002    rem hops 17

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/2            Desg BLK 200000    128.4    P2p
Fa0/4            Desg BLK 200000    128.6    P2p
Fa0/6            Desg BLK 200000    128.8    P2p
Fa0/13           Altn BLK 500000    128.15   P2p
Fa0/14           Altn BLK 500000    128.16   P2p
Fa0/15           Altn BLK 500000    128.17   P2p
Fa0/19           Altn BLK 200000    128.21   P2p
Fa0/20           Root FWD 2         128.22   P2p
```

Copyright © 2010 Internetwork Expert

1.35 MST Load Balancing with Port Priority

- Remove the previous STP cost modifications.
- Set the cost for MST instance 1 on SW3's links to SW1 to be 100,000.
- Using Spanning-Tree priority modify the layer 2 transit network so that traffic for MST instance 1 from SW4 to SW1 uses the last link between SW3 and SW4.
- If this link goes down traffic should fall over to the second link between SW3 and SW4.

Configuration

```
SW3:
interface FastEthernet0/13
 spanning-tree mst 1 cost 100000
!
interface FastEthernet0/14
 spanning-tree mst 1 cost 100000
!
interface FastEthernet0/15
 spanning-tree mst 1 cost 100000
!
interface FastEthernet0/20
 spanning-tree mst 1 port-priority 16
!
interface FastEthernet0/21
 spanning-tree mst 1 port-priority 0
```

Verification

> **Note**
>
> Like CST and PVST, MST uses the designated (upstream) port-priority as a tie breaker if the end-to-end cost is the same on multiple ports to the same upstream switch. The **`show spanning-tree mst`** only shows the local port-priority, so the below output doesn't tell us why Fa0/21 is chosen as the root port.

```
Rack1SW4#show spanning-tree mst 1

##### MST1    vlans mapped:   1-100
Bridge        address 000b.46bf.fd00  priority      32769 (32768 sysid 1)
Root          address 001a.a20f.6d00  priority      1     (0 sysid 1)
              port    Fa0/21          cost          300000    rem hops 18
Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/16           Altn BLK 200000    128.16   P2p
Fa0/17           Altn BLK 200000    128.17   P2p
Fa0/18           Altn BLK 200000    128.18   P2p
Fa0/19           Altn BLK 200000    128.19   P2p
Fa0/20           Altn BLK 200000    128.20   P2p
Fa0/21           Root FWD 200000    128.21   P2p
```

Copyright © 2010 Internetwork Expert

The show spanning-tree mst detail shows that the lowest end-to-end cost of 300,000 is equal on ports Fa0/19, Fa0/20, and Fa0/21. Since all three of these ports share the same designated bridge-id, the designated port-id is checked. The port-id is made of the port-priority and the internally assigned port number. Fa0/21 has the lowest designated port-id of 0.21, versus Fa0/20's 16.20 and Fa0/19's 128.19.

```
Rack1SW4#show spanning-tree mst 1 detail

##### MST1    vlans mapped:   1-100
Bridge        address 000b.46bf.fd00  priority      32769 (32768 sysid 1)
Root          address 001a.a20f.6d00  priority      1     (0 sysid 1)
              port    Fa0/21          cost          300000    rem hops 18

FastEthernet0/16 of MST1 is alternate blocking
Port info             port id         128.16  priority    128  cost      200000
Designated root       address 001a.a20f.6d00  priority      1  cost      200000
Designated bridge     address 001a.a256.7780  priority   4097  port id   128.21
Timers: message expires in 5 sec, forward delay 0, forward transitions 1
Bpdus (MRecords) sent 109, received 148

FastEthernet0/17 of MST1 is alternate blocking
Port info             port id         128.17  priority    128  cost      200000
Designated root       address 001a.a20f.6d00  priority      1  cost      200000
Designated bridge     address 001a.a256.7780  priority   4097  port id   128.22
Timers: message expires in 5 sec, forward delay 0, forward transitions 0
Bpdus (MRecords) sent 631, received 686

FastEthernet0/18 of MST1 is alternate blocking
Port info             port id         128.18  priority    128  cost      200000
Designated root       address 001a.a20f.6d00  priority      1  cost      200000
Designated bridge     address 001a.a256.7780  priority   4097  port id   128.23
Timers: message expires in 5 sec, forward delay 0, forward transitions 0
Bpdus (MRecords) sent 632, received 688

FastEthernet0/19 of MST1 is alternate blocking
Port info             port id         128.19  priority    128  cost      200000
Designated root       address 001a.a20f.6d00  priority      1  cost      100000
Designated bridge     address 000d.653a.2680  priority  32769  port id   128.19
Timers: message expires in 4 sec, forward delay 0, forward transitions 1
Bpdus (MRecords) sent 108, received 203

FastEthernet0/20 of MST1 is alternate blocking
Port info             port id         128.20  priority    128  cost      200000
Designated root       address 001a.a20f.6d00  priority      1  cost      100000
Designated bridge     address 000d.653a.2680  priority  32769  port id    16.20
Timers: message expires in 5 sec, forward delay 0, forward transitions 1
Bpdus (MRecords) sent 713, received 622

FastEthernet0/21 of MST1 is root forwarding
Port info             port id         128.21  priority    128  cost      200000
Designated root       address 001a.a20f.6d00  priority      1  cost      100000
Designated bridge     address 000d.653a.2680  priority  32769  port id     0.21
Timers: message expires in 5 sec, forward delay 0, forward transitions 1
Bpdus (MRecords) sent 599, received 507
```

Copyright © 2010 Internetwork Expert

When SW4 loses its connection to SW3 via Fa0/21 the next port in line is Fa0/20 with the designated port-id of 16.20.

```
Rack1SW4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW4(config)#interface fa0/21
Rack1SW4(config-if)#shut
Rack1SW4(config-if)#end
Rack1SW4#
00:20:12: %SYS-5-CONFIG_I: Configured from console by console
00:20:13: %LINK-5-CHANGED: Interface FastEthernet0/21, changed state to
administratively down
00:20:14: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/21,
changed state to down

Rack1SW4#show spanning-tree mst 1

##### MST1    vlans mapped:   1-100
Bridge        address 000b.46bf.fd00  priority      32769 (32768 sysid 1)
Root          address 001a.a20f.6d00  priority      1     (0 sysid 1)
              port    Fa0/20          cost          300000    rem hops 18

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/16           Altn BLK 200000    128.16   P2p
Fa0/17           Altn BLK 200000    128.17   P2p
Fa0/18           Altn BLK 200000    128.18   P2p
Fa0/19           Altn BLK 200000    128.19   P2p
Fa0/20           Root FWD 200000    128.20   P2p
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.36 MST and Rapid Spanning Tree

- Configure Rapid Spanning-Tree on the switches so that ports connected to the internal and external routers immediately begin forwarding when enabled.

Configuration

```
SW1:
interface FastEthernet0/1
 spanning-tree portfast
!
interface FastEthernet0/5
 spanning-tree portfast

SW2:
interface FastEthernet0/2
 spanning-tree portfast
!
interface FastEthernet0/4
 spanning-tree portfast
!
interface FastEthernet0/6
 spanning-tree portfast trunk
!
interface FastEthernet0/24
 spanning-tree portfast

SW3:
interface FastEthernet0/5
 spanning-tree portfast
!
interface FastEthernet0/24
 spanning-tree portfast

SW4:
interface FastEthernet0/4
 spanning-tree portfast
```

Copyright © 2010 Internetwork Expert

Verification

When MST is enabled, Rapid Spanning-Tree Protocol (RSTP) is automatically enabled. RSTP is an IEEE standard defined in 802.1w that speeds up convergence through a reliable handshaking process. RSTP defines new port "roles" to automatically allow for the functionality built into Cisco proprietary features such as PortFast and UplinkFast.

RSTP "edge" ports behave the same as PVST PortFast enabled ports. However, in order to maintain backwards compatible configurations Cisco's implementation of RSTP does not automatically elect edge ports as the standard suggests. Instead a port must be configured as an edge port with the **`spanning-tree portfast`** command.

```
Rack1SW1#show spanning-tree mst interface fa0/1

FastEthernet0/1 of MST0 is designated forwarding
Edge port: edge            (enable)          port guard : none        (default)
Link type: point-to-point (auto)             bpdu filter: disable     (default)
Boundary : internal                          bpdu guard : disable     (default)
Bpdus sent 260, received 0

Instance Role Sts Cost      Prio.Nbr Vlans mapped
-------- ---- --- --------- -------- --------------------------------
0        Desg FWD 200000    128.3    none
2        Desg FWD 200000    128.3    101-200

Rack1SW2#show spanning-tree mst interface fa0/6

FastEthernet0/6 of MST0 is designated forwarding
Edge port: edge            (trunk)           port guard : none        (default)
Link type: point-to-point (auto)             bpdu filter: disable     (default)
Boundary : internal                          bpdu guard : disable     (default)
Bpdus sent 30, received 0

Instance Role Sts Cost      Prio.Nbr Vlans mapped
-------- ---- --- --------- -------- --------------------------------
0        Desg FWD 200000    128.8    none
1        Desg FWD 200000    128.8    1-100
2        Desg FWD 200000    128.8    101-200
```

Copyright © 2010 Internetwork Expert

1.37 Protected Ports

- Create a new SVI for VLAN22 on SW2 and assign it the IP address 192.10.X.8/24, where X is your rack number.
- Configure port protection on SW2 so that R2 and BB2 cannot directly communicate with each other, but can communicate with SW2's VLAN22 interface.

Configuration

```
SW2:
interface FastEthernet0/2
 switchport protected
!
interface FastEthernet0/24
 switchport protected
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Protected ports are used to prevent traffic from being exchanged at layer 2 between two or more ports that are in the same VLAN. Traffic received in a protected port cannot be sent out another protected port, however traffic received in a protected port can be sent out a non-protected port. This feature is a much smaller subset of the Private VLAN feature, and cannot span between multiple physical switches.
>
> In this particular design the result of port protection is that R2 and SW2 can communicate, SW2 and BB2 can communicate, but R2 and BB2 cannot.

```
Rack1R2#ping 192.10.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.10.1.254, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)

Rack1R2#ping 192.10.1.8

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.10.1.8, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms

Rack1SW2#ping 192.10.1.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.10.1.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/8 ms

Rack1SW2#ping 192.10.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.10.1.254, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/8 ms

Rack1R2#show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  192.10.1.254            0   Incomplete      ARPA
Internet  192.10.1.8              0   001a.a256.77c3  ARPA   FastEthernet0/0
Internet  192.10.1.2              -   000d.65c2.f1c0  ARPA   FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

1.38 Storm Control

- Configure SW1 to limit unicast traffic received from R1 to 100 pps.
- Configure SW2 to limit broadcast traffic received from R6 to 10Mbps.
- Configure SW4 to limit broadcast traffic received from R4 to 1Mbps using a relative percentage of the interface bandwidth.

Configuration

```
SW1:
interface FastEthernet0/1
 storm-control unicast level pps 100

SW2:
interface FastEthernet0/6
 storm-control broadcast level bps 10m

SW4:
interface FastEthernet0/4
 storm-control broadcast level 10.0
```

Verification

> **Note**
>
> Storm control is used to limit the amount of unicast, multicast, or broadcast traffic received in a port. The most common application of this feature is to prevent broadcast storms, but it can also be used to police individual ports not to exceed a desired rate.
>
> Depending on the version of IOS the **storm-control** command may take units in percentage, packets per second, bits per second, or others. Make sure to use the question mark when implementing this command so that the units entered achieve the desired result.

```
Rack1SW2#show storm-control
Interface  Filter State   Upper         Lower         Current
---------  -------------  ------------  ------------  ----------
Fa0/6      Link Down         10m bps       10m bps        0 bps

Rack1SW4#show storm-control
Interface  Filter State   Upper         Lower         Current
---------  -------------  ------------  ------------  ----------
Fa0/4      Link Down         1.00%         1.00%         0.00%
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.39 MAC-Address Table Static Entries & Aging

- Ensure reachability on VLAN 146 between R1, R4, and R6.
- Configure a static CAM entry on SW4 so that frames destined to the MAC address of R4's interface connected to VLAN 146 are dropped; once complete R1 and R6 should have reachability to each other, but not R4.
- Configure static CAM entry for that MAC address of R6's connection to VLAN 146 to ensure that this address is not allowed to roam.

Configuration

```
SW2:
mac address-table static 000f.23f4.e640 vlan 146 interface
FastEthernet0/6

SW4:
mac address-table static 000a.f4b0.cfc2 vlan 146 drop
```

Verification

Note

Normally switches populate the CAM table, or MAC address table, by flooding unknown frames everywhere in the VLAN they were received in and by looking at the source MAC address of frames received in its ports. In certain circumstances this can be undesirable, such as when someone attempts to do a layer 2 MAC address spoofing attack. A simple way to prevent these types of attacks is to statically hard-code which MAC addresses are reachable via which ports.

Another static feature of the CAM table is the ability to Null route MAC addresses. Since static entries always override dynamically learned entries, if the drop keyword or an unused interface is used in the **`mac address-table static`** command traffic destined to that MAC address will be dropped.

Copyright © 2010 Internetwork Expert

In this particular design R1, R4, and R6 exchange traffic on VLAN 146. SW4, who is connected to R4's port Fa0/1, dynamically learns R4's MAC address 000a.f4b0.cfc2 in port Fa0/4.

```
Rack1R1#ping 155.1.146.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms

Rack1R1#ping 155.1.146.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1R4#ping 155.1.146.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms
Rack1R4#

Rack1SW4#show mac address-table dynamic interface fa0/4
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
 146    000a.f4b0.cfc2    DYNAMIC     Fa0/4
Total Mac Addresses for this criterion: 1
```

Copyright © 2010 Internetwork Expert

When SW4 is configured with a static entry that matches this address with the keyword drop at the end, the dynamically learned entry is overridden. The result is that any traffic going to R4, such as the ICMP PING from R1, is dropped in the layer 2 transit path.

```
Rack1SW4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW4(config)#mac address-table static 000a.f4b0.cfc2 vlan 146 drop
Rack1SW4(config)#

Rack1R1#ping 155.1.146.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)
```

Likewise traffic going to R6 uses the static entry as opposed to the dynamically learned entry.

```
Rack1R1#ping 155.1.146.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms

Rack1SW2#show mac address-table address 000f.23f4.e640
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    000f.23f4.e640    DYNAMIC     Fa0/6
 146    000f.23f4.e640    STATIC      Fa0/6
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.40 SPAN

- Configure SW1 so that all traffic transiting VLAN 146 is redirected to a host located on port Fa0/24.
- Configure SW4 so that all traffic coming from and going to R4's connection to VLAN 146 is redirected to a host located on port Fa0/24; Inbound traffic from the Linux host should be placed into VLAN 146.

Configuration

```
SW1:
monitor session 1 source vlan 146
monitor session 1 destination interface Fa0/24

SW4:
monitor session 1 source interface Fa0/4
monitor session 1 destination interface Fa0/24 ingress vlan 146
```

Copyright © 2010 Internetwork Expert

Verification

Note

The Switchport Analyzer (SPAN) feature is used to redirect traffic from a port or VLAN onto another port for analysis by devices such as a packet sniffer or Intrusion Prevention Sensor (IPS). There are two variations of SPAN, Local SPAN, or just SPAN, and Remote SPAN, or RSPAN.

With Local SPAN, as seen in this design SW4, traffic coming from or going to a particular port is redirect to another local port. The source of traffic can also be a VLAN, as seen on SW1.Normally when the SPAN feature is configured the switch drops all traffic coming back in the destination port.

The ingress keyword tells the switch which access VLAN inbound traffic on the destination port should belong to.

```
Rack1SW1#show monitor session 1
Session 1
---------
Type               : Local Session
Source VLANs       :
    Both           : 146
Destination Ports  : Fa0/24
    Encapsulation  : Native
          Ingress  : Disabled

Rack1SW4#show monitor session 1
Session 1
---------
Type               : Local Session
Source Ports       :
    Both           : Fa0/4
Destination Ports  : Fa0/24
    Encapsulation  : Native
          Ingress  : Enabled, default VLAN = 146
```

Copyright © 2010 Internetwork Expert

1.41 RSPAN

- Disable the trunk links between SW1 and SW2.
- Create VLAN 500 as an RSPAN VLAN on all switches in the topology.
- Configure SW2 so that traffic received from and sent to R4's connection to VLAN 43 is redirected to the RSPAN VLAN.
- Configure SW1 to receive traffic from the RSPAN VLAN and redirect it to a host connected to port Fa0/24.
- Inbound traffic on the link connected to this host should be placed in VLAN 146.

Configuration

```
SW1:
interface FastEthernet0/13
 shutdown
!
interface FastEthernet0/14
 shutdown
!
interface FastEthernet0/15
 shutdown
!
monitor session 2 destination interface Fa0/24 ingress vlan 146
monitor session 2 source remote vlan 500

SW2:
monitor session 2 source interface Fa0/4
monitor session 2 destination remote vlan 500

SW4:
vlan 500
 remote-span
```

Copyright © 2010 Internetwork Expert

Verification

Note

The Remote SPAN, or RSPAN, feature is used when the source port or VLAN that is being monitored is on a different physical switch than the destination sniffer or sensor.

The first step in configuring RSPAN is to ensure that the switches in the layer 2 transit path from the source port/VLAN to the destination port are trunking at layer 2, and know about the RSPAN VLAN that is used to encapsulate and transport the monitored traffic. In this case VTP is used, so only the VTP server SW4 needs to create the VLAN. Note the remote-span keyword under the VLAN, as this is a special attribute that affects how traffic is processed when it is received in this VLAN.

Next the switch attached to the source port or VLAN creates a SPAN session. The source of this span session, in the case of SW2, is all traffic coming in port Fa0/4. The destination of the session is the RSPAN VLAN 500 itself. This means that all traffic that comes in port Fa0/4 will receive a new trunking header with a VLAN 500 tag and be sent out the trunk network.

Lastly the switch attached to the sniffer/sensor creates a SPAN session with the source as the RSPAN VLAN, and the destination as the local port. This means that the switch wants to listen for all traffic received in the RSPAN VLAN, and redirect it out a local port. In this case SW1 says that the source of the session is the *remote* vlan 500. On SW1 therefore all traffic coming in a trunk link with a tag of 500 will be redirected out port Fa0/24. Since the *ingress* keyword is also used, any traffic that SW1 receives in port Fa0/24 will be treated as if it belongs to VLAN 146.

```
Rack1SW1#show monitor session 2
Session 2
---------
Type               : Remote Destination Session
Source RSPAN VLAN  : 500
Destination Ports  : Fa0/24
    Encapsulation  : Native
          Ingress  : Enabled, default VLAN = 146
    Ingress encap  : Untagged
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show monitor session 2
Session 2
---------
Type               : Remote Source Session
Source Ports       :
    Both           : Fa0/4
Dest RSPAN VLAN    : 500
```

```
Rack1SW2#show vlan

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/1, Fa0/3, Fa0/5, Fa0/7
                                                Fa0/8, Fa0/9, Fa0/10, Fa0/11
                                                Fa0/12, Fa0/13, Fa0/14, Fa0/15
                                                Fa0/16, Fa0/17, Fa0/18, Fa0/22
                                                Fa0/23, Gi0/1, Gi0/2
5    VLAN0005                         active
7    VLAN0007                         active
8    VLAN0008                         active
9    VLAN0009                         active
10   VLAN0010                         active
22   VLAN0022                         active    Fa0/2, Fa0/24
43   VLAN0043                         active    Fa0/4
58   VLAN0058                         active
67   VLAN0067                         active
79   VLAN0079                         active
146  VLAN0146                         active
500  VLAN0500                         active
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        0      0
5    enet  100005     1500  -      -      -        -    -        0      0
7    enet  100007     1500  -      -      -        -    -        0      0
8    enet  100008     1500  -      -      -        -    -        0      0
9    enet  100009     1500  -      -      -        -    -        0      0
10   enet  100010     1500  -      -      -        -    -        0      0
22   enet  100022     1500  -      -      -        -    -        0      0
43   enet  100043     1500  -      -      -        -    -        0      0
58   enet  100058     1500  -      -      -        -    -        0      0
67   enet  100067     1500  -      -      -        -    -        0      0
79   enet  100079     1500  -      -      -        -    -        0      0
146  enet  100146     1500  -      -      -        -    -        0      0
500  enet  100500     1500  -      -      -        -    -        0      0
1002 fddi  101002     1500  -      -      -        -    -        0      0
1003 tr    101003     1500  -      -      -        -    srb      0      0
1004 fdnet 101004     1500  -      -      -        ieee -        0      0
1005 trnet 101005     1500  -      -      -        ibm  -        0      0

Remote SPAN VLANs
------------------------------------------------------------------------------
500

Primary Secondary Type              Ports
------- --------- ----------------- ------------------------------------------
```

Copyright © 2010 Internetwork Expert

1.42 Voice VLAN

- Ports Fa0/2, Fa0/4, and Fa0/6 on SW1 will be connected to Cisco IP phones in the near future.
- Configure port Fa0/2 with an access VLAN assignment of 146 and a voice VLAN assignment of 600.
- Enable Spanning-Tree portfast on this link and ensure that CDP is enabled.
- Configure port Fa0/4 as an 802.1q trunk link.
- Configure SW1 so that only VLANs 146 and 600 are permitted on this switchport, so that STP BPDUs received on the port are filtered out, and so that the interface runs in STP portfast mode.
- Configure VLAN 146 as the native VLAN for this port and so that VLAN 600 is advertised as the voice VLAN via CDP.
- Configure port Fa0/6 with an access VLAN assignment of 146, and for voice VLAN frames to use dot1p tagging.

Configuration

```
SW1:
interface FastEthernet0/2
 switchport access vlan 146
 switchport voice vlan 600
 spanning-tree portfast
!
interface FastEthernet0/4
 switchport trunk encapsulation dot1q
 switchport trunk native vlan 146
 switchport trunk allowed vlan 146,600
 switchport mode trunk
 switchport voice vlan 600
 spanning-tree portfast trunk
 spanning-tree bpdufilter enable
!
interface FastEthernet0/6
 switchport access vlan 146
 switchport voice vlan dot1p

SW4:
vlan 600
```

Copyright © 2010 Internetwork Expert

Verification

Note

Many models of Cisco IP Phones have a built-in three-port switch, one port to connect to the upstream switch, one port for the IP Phone itself, and the last port to connect to a desktop PC. The built-in switch is capable of separating the IP Phone and the desktop PC traffic using different VLANs. Additionally the internal switch can also use different 802.1p markings in the Class of Service (CoS) field to distinguish the IP Phone and the desktop PC frames. Based on this there are three different options for connecting the IP Phone and the desktop PC to the Catalyst switches.

Option 1 is to separate the Data VLAN for the PC and the Voice VLAN for the IP Phone. The internal IP Phone switch will tag VoIP traffic with the respective VLAN number and apply a CoS value of 5. The data frames are sent untagged and received by the upstream switch on the configured access VLAN. The connection between the IP Phone and the upstream switch is an 802.1q trunk with the native VLAN equal to the Data VLAN.

Option 2 is to use a single VLAN for Data and Voice. The IP Phone's internal switch does not tag the frames and acts as a simple bridge. The connection between the IP Phone and the upstream switch is an access port.

Option 3 is to use a single VLAN for Data and Voice, but to add an 802.1p CoS tag. Data frames received from the PC on the phone, along with VoIP frames sent from the phone get a special 802.1q header that carries a VLAN ID equal to zero and has the CoS field set to 5 for VoIP and the value instructed from the switch for data frames. The Catalyst switch accepts the frames with VLAN zero as if they are in the access VLAN, but also honors the CoS bits to calculate the switch's internal QoS tag.

Copyright © 2010 Internetwork Expert

For all three options the IP Phone's built-in switch should be instructed which mode to use. The command **`switchport voice vlan`** configured on an access port will communicate with the IP Phone via CDP and tell its internal switch which VLAN should be used for voice traffic. The IP Phone's internal switch will then apply the instructed VLAN tag to the voice traffic and will send the PC's data untagged. Note that there is no need to configure the port as an 802.1q trunk via the **`switchport mode trunk`** command. The switchport ASIC will automatically convert the port into a rudimentary trunk.

If no **`switchport voice vlan`** command is configured, then Option 2 applies automatically. Both voice and data packets are received on the same VLAN (the access VLAN).

If the command **`switchport voice vlan dot1p`** is configure on a switchport then the connected IP Phone's switch is instructed to apply VLAN 0 to voice traffic along with the corresponding CoS bits. Both voice and data frames will share the same VLAN configured on the access port.

Note that as soon as the **`switchport voice vlan`** command is applied to the port, the **`spanning-tree portfast`** feature is automatically enabled.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.43 IP Phone Trust and CoS Extend

- Enable MLS QoS globally on SW1.
- Configure SW1 to trust the CoS of frames received on the ports connected to the IP phones.
- This trust should only occur if the Cisco IP phone is present and advertises itself via CDP.
- SW1 should enforce a CoS value of 1 to any appliance connected to the second port of the IP phone.

Configuration

```
SW1:
mls qos
!
interface FastEthernet0/2
 mls qos trust cos
 mls qos trust device cisco-phone
 switchport priority extend cos 1
!
interface FastEthernet0/4
 mls qos trust cos
 mls qos trust device cisco-phone
 switchport priority extend cos 1
!
interface FastEthernet0/6
 mls qos trust cos
 mls qos trust device cisco-phone
 switchport priority extend cos 1
```

Copyright © 2010 Internetwork Expert

Verification

Note

The QoS trust state of the port determines if frames with a CoS value are maintained or remarked as they are received. In this case these ports are configured to trust the QoS marking only if the presence of a Cisco IP Phone is sensed via CDP messages. This option is enabled with the command **`mls qos trust device cisco-phone`**. If no Cisco device is detected on the port then the QoS markings are not trusted, even if the port is configured for trust.

In addition to enforcing markings at the switchport boundary, the switch may also instruct the IP Phone's switch to apply specific CoS markings for frames received from the connected PC. The switch may either accept (trust) 802.1p bits received from the attached PC or enforce the instructed value. This feature particularly makes sense to be used with the dot1p Voice VLAN option.

```
Rack1SW1#show mls qos interface fa0/2
FastEthernet0/2
trust state: not trusted
trust mode: trust cos
trust enabled flag: dis
COS override: dis
default COS: 0
DSCP Mutation Map: Default DSCP Mutation Map
Trust device: cisco-phone
qos mode: port-based
```

Copyright © 2010 Internetwork Expert

1.44 Smartport Macros

- Configure a macro on SW1 named VLAN_146 that when applied to an interface will set it to be an access switchport, apply VLAN 146 as the access vlan, and filter Spanning-Tree BPDUs.
- Apply this macro to ports Fa0/7 and Fa0/8 on the switch.

Configuration

```
SW1:
macro name VLAN_146
switchport mode access
switchport access vlan 146
spanning-tree bpdufilter enable
@
```

Verification

> **Note**
>
> Smartport Macros are used to define a well known template of configuration to apply onto multiple interfaces. This feature is useful in large switching environments where general categories of ports can be defined, such as access, server, uplink, and have them share common configuration templates.
>
> In this particular design the macro is used to apply three attributes to the interface, the switchport mode, the access VLAN, and the BPDU Filter feature. The result seen from the show run output is identical to that which would be achieved by manually entering these commands on both interfaces, with the addition of the **`macro description`** telling us which macro was applied.

```
Rack1SW1#config t
Rack1SW1(config)#interface range fa0/7-8
Rack1SW1(config-if-range)#macro apply VLAN_146
Rack1SW1(config-if-range)#end
02:11:37: %SYS-5-CONFIG_I: Configured from console by console

Rack1SW1#show run interface fa0/7
Building configuration...

Current configuration : 146 bytes
!
interface FastEthernet0/7
 switchport access vlan 146
 switchport mode access
 macro description VLAN_146
 spanning-tree bpdufilter enable
end
```

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show run interface fa0/8
Building configuration...

Current configuration : 146 bytes
!
interface FastEthernet0/8
 switchport access vlan 146
 switchport mode access
 macro description VLAN_146
 spanning-tree bpdufilter enable
end
```

A number of default Smartport Macros exist in the switch, and can be seen by issuing the **show parser macro** command.

```
Rack1SW1#show parser macro
Total number of macros = 6
------------------------------------------------------------
Macro name : cisco-global
Macro type : default global
# Enable dynamic port error recovery for link state failures.
errdisable recovery cause link-flap
errdisable recovery interval 60

# Config Cos to DSCP mappings
mls qos map cos-dscp 0 8 16 26 32 46 46 56

# Enable aggressive mode UDLD on all fiber uplinks
udld aggressive

# Enable Rapid PVST+ and Loopguard
spanning-tree mode rapid-pvst
spanning-tree loopguard default
spanning-tree extend system-id
------------------------------------------------------------
Macro name : cisco-desktop
Macro type : default interface
# macro keywords $access_vlan
# Basic interface - Enable data VLAN only
# Recommended value for access vlan should not be 1
switchport access vlan $access_vlan
switchport mode access

# Enable port security limiting port to a single
# MAC address -- that of desktop
switchport port-security
switchport port-security maximum 1

# Ensure port-security age is greater than one minute
# and use inactivity timer
switchport port-security violation restrict
switchport port-security aging time 2
switchport port-security aging type inactivity
```

Copyright © 2010 Internetwork Expert

```
# Configure port as an edge network port
spanning-tree portfast
spanning-tree bpduguard enable
------------------------------------------------------------------
<output omitted>
```

A default macro can be applied as follows.

```
Rack1SW1#show run interface fa0/10
Building configuration...

Current configuration : 34 bytes
!
interface FastEthernet0/10
end

Rack1SW1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface fa0/10
Rack1SW1(config-if)#macro apply cisco-desktop $access_vlan 10
%Warning: portfast should only be enabled on ports connected to a single
 host. Connecting hubs, concentrators, switches, bridges, etc... to this
 interface  when portfast is enabled, can cause temporary bridging loops.
 Use with CAUTION

%Portfast has been configured on FastEthernet0/10 but will only
 have effect when the interface is in a non-trunking mode.
Rack1SW1(config-if)#end
Rack1SW1#show run interface fa0/10
Building configuration...

Current configuration : 332 bytes
!
interface FastEthernet0/10
 switchport access vlan 10
 switchport mode access
 switchport port-security
 switchport port-security aging time 2
 switchport port-security violation restrict
 switchport port-security aging type inactivity
 macro description cisco-desktop
 spanning-tree portfast
 spanning-tree bpduguard enable
end
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.45 Flex Links

- Configure links Fa0/16 between SW2 and SW3 as an 802.1q trunk.
- Configure link Fa0/16 on SW1 and Fa0/13 on SW3 as an 802.1q trunk.
- Configure links Fa0/13 & Fa0/14 between SW1 and SW2 as an 802.1q trunked EtherChannel.
- Disable all other inter-switch links.
- Configure R1's Ethernet interface with the IP address 10.0.0.1/24, R2's Ethernet interface with the IP address 10.0.0.2/24, and R3's second Ethernet interface with the IP address 10.0.0.3/24.
- Configure flex links on SW1 so that traffic from R1 to R3 uses the EtherChannel to SW2.
- If the EtherChannel goes down traffic should immediately switch over to use the link between SW1 and SW3.
- If the EtherChannel and all its members comes back up traffic should forward back over this link after 20 seconds.

Configuration

```
R1:
interface FastEthernet0/0
 ip address 10.0.0.1 255.255.255.0

R2:
interface FastEthernet0/0
 ip address 10.0.0.2 255.255.255.0

R3:
interface FastEthernet0/1
 ip address 10.0.0.3 255.255.255.0

SW1:
interface Port-channel1
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport backup interface Fa0/16
 switchport backup interface Fa0/16 preemption mode forced
 switchport backup interface Fa0/16 preemption delay 20
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
 channel-group 1 mode on
!
interface FastEthernet0/14
 switchport trunk encapsulation dot1q
 switchport mode trunk
 channel-group 1 mode on
!
```

Copyright © 2010 Internetwork Expert

```
interface FastEthernet0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk

SW2:
interface Port-channel1
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
 channel-group 1 mode on
!
interface FastEthernet0/14
 switchport trunk encapsulation dot1q
 switchport mode trunk
 channel-group 1 mode on
!
interface FastEthernet0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk

SW3:
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk
```

Copyright © 2010 Internetwork Expert

Verification

Note

The Flex Links feature is used as an alternative to Spanning-Tree Protocol in environments where physical loops occur in the layer 2 network. Flex Links work like the **backup interface** feature on the routers, in which a layer 2 physical interface or Port-Channel is configured as the "active" link, and another layer 2 link is configured as the "backup". STP is automatically disabled on both links when Flex Links are enabled.

The backup link operates in standby mode, and waits for the line protocol of the active link to go down. If the line protocol of the active link is down, the backup link becomes active and immediately starts forwarding. When the active link's line protocol status comes back up, the backup link goes back into standby state and stops forwarding traffic.

In this particular design SW1 has Port-Channel1 configured as the active link and FastEthernet0/16 configured as the backup.

```
Rack1SW1#show interfaces po1 switchport backup

Switch Backup Interface Pairs:

Active Interface        Backup Interface        State
------------------------------------------------------------------------
Port-channel1           FastEthernet0/16        Active Up/Backup Standby
```

```
Rack1R1#ping 10.0.0.3 repeat 5000

Type escape sequence to abort.
Sending 5000, 100-byte ICMP Echos to 10.0.0.3, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
<output omitted>
```

SW2's Port-Channel1 interface is shutdown, causing SW1's link to go down.

```
Rack1SW2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW2(config)#interface po1
Rack1SW2(config-if)#shut
Rack1SW2(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Port-channel1, changed
state to down
%LINK-5-CHANGED: Interface FastEthernet0/13, changed state to
administratively down
%LINK-5-CHANGED: Interface FastEthernet0/14, changed state to
administratively down
```

Copyright © 2010 Internetwork Expert

SW1 detects this and immediately activates port Fa0/16.

```
Rack1SW1#debug backup all
Switch Backup Interface all debugging is on
sw_backup_int: intf Po1, state 1, transition for event 0
sw_backup_int: Po1 is now Down
BACKUP_INT: idb Po1, peer Fa0/16, state Down
sw_backup_int: intf Fa0/16, state 2, transition for event 1
sw_backup_int: Fa0/16 is now Up
BACKUP_INT: intf Po1, updating vtp pruning join bits
BACKUP_INT: intf Fa0/16, updating vtp pruning join bits
BACKUP_INT: intf Po1, state up, bandwidth 100000 Kbps
BACKUP_INT: setting WB
BACKUP_INT: clearing WB
BACKUP_INT: Pair Po1 Fa0/16 mode bandwidth, delay 20 seconds,
Unscheduled
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/13,
changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/14,
changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Port-channel1, changed
state to down
BACKUP_INT: intf Po1, state down, bandwidth 100000 Kbps
BACKUP_INT: setting WB
BACKUP_INT: clearing WB
BACKUP_INT: Pair Po1 Fa0/16 mode bandwidth, delay 20 seconds,
Unscheduled
Rack1SW1#
%LINK-3-UPDOWN: Interface FastEthernet0/13, changed state to down
%LINK-3-UPDOWN: Interface Port-channel1, changed state to down
%LINK-3-UPDOWN: Interface FastEthernet0/14, changed state to down

Rack1SW1#show interfaces po1 switchport backup

Switch Backup Interface Pairs:

Active Interface        Backup Interface        State
------------------------------------------------------------------------
Port-channel1           FastEthernet0/16        Active Down/Backup Up
```

R1, who was sending traffic to R3 while the failure occurred, dropped one packet out of 5000. This implies that the network converged in less than four seconds, as the default timeout for a ping is two seconds.

```
[Resuming connection 1 to r1 ... ]
<output omitted>
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 99 percent (4999/5000), round-trip min/avg/max = 1/2/48
ms
Rack1R1#
```

Copyright © 2010 Internetwork Expert

When the Po1 interface of SW1 comes back up, a preemption delay counter starts, as configured with the **preemption delay 20** command. After the 20 second delay expires, the bandwidth of the Fa0/16 interface is compared with Po1 due to the **preemption mode bandwidth** command. Since Po1 has a higher bandwidth value it preempts Fa0/16, and Fa0/16 goes into the standby state.

```
Rack1SW2(config)#int po1
Rack1SW2(config-if)#no shut

Rack1SW1#
%LINK-3-UPDOWN: Interface FastEthernet0/13, changed state to up
%LINK-3-UPDOWN: Interface FastEthernet0/14, changed state to up
sw_backup_int: intf Po1, state 0, transition for event 2
sw_backup_int: Po1 is now Waiting to sync
BACKUP_INT: idb Po1, peer Fa0/16, state Waiting to sync
sw_backup_int: intf Po1, state 3, transition for event 6
sw_backup_int: Po1 is now Waiting for peer state
BACKUP_INT: idb Po1, peer Fa0/16, state Waiting for peer state
sw_backup_int: intf Fa0/16, state 1, transition for event 5
BACKUP_INT: idb Fa0/16, peer Po1, state Up
sw_backup_int: intf Po1, state 4, transition for event 3
sw_backup_int: Po1 is now Standby
BACKUP_INT: intf Po1, state up, bandwidth 200000 Kbps
BACKUP_INT: setting WB
BACKUP_INT: clearing WB
BACKUP_INT: AI Po1 ai_state 2 ai_bw 200000, BI Fa0/16 bi_state 1 bi_bw 100000
BACKUP_INT: Pair Po1 Fa0/16 mode bandwidth, delay 20 seconds, Scheduled
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/13, changed state
to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/14, changed state
to up
%LINK-3-UPDOWN: Interface Port-channel1, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Port-channel1, changed state to
up
BACKUP_INT: intf Po1, state up, bandwidth 200000 Kbps
BACKUP_INT: setting WB
BACKUP_INT: clearing WB
BACKUP_INT: AI Po1 ai_state 2 ai_bw 200000, BI Fa0/16 bi_state 1 bi_bw 100000
BACKUP_INT: Pair Po1 Fa0/16 mode bandwidth, delay 20 seconds, Scheduled
Rack1SW1#
BACKUP_INT: AI Po1 ai_state 2 ai_bw 200000, BI Fa0/16 bi_state 1 bi_bw 100000
%BACKUP_INTERFACE-5-PREEMPT: Preempting interface Fa0/16 in backup pair (Po1,
Fa0/16), preemption mode is bandwidth
<output omitted>

Rack1SW1#show interfaces po1 switchport backup

Switch Backup Interface Pairs:

Active Interface        Backup Interface        State
------------------------------------------------------------------------
Port-channel1           FastEthernet0/16        Active Up/Backup Standby
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.46 Fallback Bridging

- Configure R4's second Ethernet interface with the IP address 104.0.0.4/24, and with the IPv6 address 2001::4/24.
- Configure R6's second Ethernet interface with the IP address 106.0.0.6/24, and with the IPv6 address 2001::6/24.
- Configure interface VLAN104 on SW4 with the IP address 104.0.0.10/24, and configure interface Fa0/4 in VLAN 104.
- Configure interface Fa0/6 on SW4 with the IP address 106.0.0.10/24.
- Enable RIPv2 on all of these links.
- Configure fallback bridging on SW4 to bridge the IPv6 subnet of R4 and R6 together.

Configuration

```
R4:
interface FastEthernet0/1
 ip address 104.0.0.4 255.255.255.0
 ipv6 address 2001::4/64
!
router rip
 version 2
 no auto-summary
 network 104.0.0.0

R6:
interface FastEthernet0/1
 ip address 106.0.0.6 255.255.255.0
 ipv6 address 2001::6/64
!
router rip
 version 2
 no auto-summary
 network 106.0.0.0

SW4:
vlan 104
!
bridge 1 protocol vlan-bridge
!
interface FastEthernet0/4
 switchport access vlan 104
!
interface FastEthernet0/6
 no switchport
 ip address 106.0.0.10 255.255.255.0
 bridge-group 1
!
```

Copyright © 2010 Internetwork Expert

```
interface Vlan104
 ip address 104.0.0.10 255.255.255.0
 bridge-group 1
!
ip routing
!
router rip
 version 2
 no auto-summary
 network 104.0.0.0
 network 106.0.0.0
```

Verification

Note

The Fallback Bridging feature is used to bridge non-routed protocols between SVIs or native layer 3 routed interfaces. This feature is similar in theory to the Concurrent Routing and Bridging (CRB) and Integrated Routing and Bridging (IRB) features on the routers, where one protocol stack is routed on an interface while another protocol stack is bridged.

For example if a Catalyst switch has two layer 3 interfaces configured, VLAN 10 with the IP subnet 10.0.0.0/8, and VLAN 20 with the IP subnet 20.0.0.0/8, traffic from host 10.0.0.1 and 20.0.0.1 is routed at layer 3. If fallback bridging is configured on the SVI interfaces of VLAN 10 and VLAN 20, 10.0.0.1 and 20.0.0.1 are in different IPv4 subnets, but can be in the same IPX network and have their IPX traffic bridged together.

There are only two steps to implement this feature, create the fallback bridge group with the **`bridge [num] protocol vlan-bridge`** command, and apply it to the layer 3 interfaces with the **`bridge-group [num]`** command.

In this design the feature is tested by bridging IPv6 traffic between R4 and R6. This is done by configuring fallback bridging on SW4's SVI interface VLAN104 connecting to R4 and the native layer 3 routed interface FastEthernet0/6 connecting to R6. In this topology SW4 is a 3550 running IOS 12.2(25), which does not support IPv6 routing. This means that IPv4 will be routed, but IPv6 can be bridged through fallback bridging.

Copyright © 2010 Internetwork Expert

We can see the result of this design is that when R4 does a traceroute to R6 via IPv4, the traffic is routed to SW4, and then sent to R6. However when IPv6 traceroute is done between R4 and R6 they appear to be directly connected.

```
Rack1R4#traceroute 106.0.0.6
Translating "106.0.0.6"

Type escape sequence to abort.
Tracing the route to 106.0.0.6

  1 104.0.0.10 4 msec 0 msec 4 msec
  2 106.0.0.6 0 msec *  0 msec

Rack1R4#traceroute 2001::6

Type escape sequence to abort.
Tracing the route to 2001::6

  1 2001::6 4 msec 0 msec 0 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.47 Private VLANs

- Configure the first Ethernet interfaces of R1, R2, R3, R4, R5, and R6 with IP addresses 100.0.0.Y/24, where Y is the device number.
- Configure the first inter-switch link between SW1 and SW2 as a trunk.
- Configure the primary VLAN 100 to service private VLANs 1000, 2000, and 3000.
- VLANs 1000 and 2000 should be community VLANs, while VLAN 3000 should be an isolated VLAN.
- Assign VLAN 1000 to the links connecting to R2 & R3, VLAN 2000 to the links connecting to R4 & R5, and VLAN 3000 to R6.
- The link connecting to R1 should be a promiscuous port.
- Ensure that R1 can reach all devices, R2 can reach R3, and R4 can reach R5.
- No other connectivity should be allowed within this topology.

Configuration

```
R1:
interface FastEthernet0/0
 ip address 100.0.0.1 255.255.255.0

R2:
interface FastEthernet0/0
 ip address 100.0.0.2 255.255.255.0

R3:
interface FastEthernet0/0
 ip address 100.0.0.3 255.255.255.0

R4:
interface FastEthernet0/0
 ip address 100.0.0.4 255.255.255.0

R5:
interface FastEthernet0/0
 ip address 100.0.0.5 255.255.255.0

R6:
interface FastEthernet0/0
 ip address 100.0.0.6 255.255.255.0
```

Copyright © 2010 Internetwork Expert

```
SW1:
vtp domain PVLANS
vtp mode transparent
!
vlan 100
  private-vlan primary
  private-vlan association 1000,2000,3000
!
vlan 1000
  private-vlan community
!
vlan 2000
  private-vlan community
!
vlan 3000
  private-vlan isolated
!
interface FastEthernet0/1
 switchport private-vlan mapping 100 1000,2000,3000
 switchport mode private-vlan promiscuous
!
interface FastEthernet0/3
 switchport private-vlan host-association 100 1000
 switchport mode private-vlan host
!
interface FastEthernet0/5
 switchport private-vlan host-association 100 2000
 switchport mode private-vlan host
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk

SW2:
vtp domain PVLANS
vtp mode transparent
!
vlan 100
  private-vlan primary
  private-vlan association 1000,2000,3000
!
vlan 1000
  private-vlan community
!
vlan 2000
  private-vlan community
!
vlan 3000
  private-vlan isolated
!
interface FastEthernet0/2
 switchport private-vlan host-association 100 1000
 switchport mode private-vlan host
!
```

Copyright © 2010 Internetwork Expert

```
interface FastEthernet0/4
 switchport private-vlan host-association 100 2000
 switchport mode private-vlan host
!
interface FastEthernet0/6
 switchport private-vlan host-association 100 3000
 switchport mode private-vlan host
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
```

Verification

Note

The Private VLAN (PVLANs) feature is similar in theory to the Protected Ports feature, in which two or more ports can be in the same VLAN but cannot directly communicate at layer 2. Private VLANs expand this concept much further however, and allow very complex security policies that can span between multiple physical switches.

Private VLANs split a single broadcast domain, that is normally defined by a single VLAN, into multiple isolated broadcast subdomains, that are defined by primary VLAN and its secondary VLANs. In essence the feature allows us to configure VLANs inside a VLAN.

Design-wise this feature is commonly used in environments like shared ISP co-location, in which customers are on the same VLAN and same IP subnet, but should not communicate directly with each other, or in Multiple Dwelling Units (MDUs) such as hotels or office buildings, where two hotel rooms or offices may be in the same subnet and VLAN but should not communicate directly.

Pitfall

The Private VLAN feature requires VTP to run in transparent mode.

Copyright © 2010 Internetwork Expert

Note

While the theory of PVLANs is relatively straight forward, the implementation can be confusing due to the different terms that Cisco uses to describe VLANs and ports, and the syntax in which they are bound together.

First we must define the different port roles used in PVLANs. These are promiscuous ports, community ports, and isolated ports. Promiscuous ports are allowed to talk to all other ports within the VLAN. Isolated ports are only allowed to talk to promiscuous ports. Community ports are allowed to talk to other ports in their own community, but not ports in different communities, and can talk to any promiscuous ports.

Configuration-wise the port roles are defined by the interface's association to a primary VLAN, and one or more secondary VLANs. First the secondary VLANs are created, and defined as either community or isolated. Next the primary VLAN is defined, and the secondary VLANs are associated with the primary VLAN.

Next the command **`switchport mode private-vlan promiscuous`** or **`switchport mode private-vlan host`** is configured at the interface level. As you might guess the *promiscuous* option defines that the port role is promiscuous, while the *host* option defines that the port role is either community or isolated. Lastly the port is assigned to both the primary and secondary VLANs, which defines what other ports it can talk to.

In this case the link to R1 has the command **`switchport private-vlan mapping 100 1000,2000,3000`** configured, which means that it is a promiscuous port in the primary VLAN 100 and can talk to all ports in the secondary VLANs 1000, 2000, and 3000.

The link to R3 has the command **`switchport private-vlan host-association 100 1000`** configured, which means that is it a member of the primary VLAN 100 and the secondary VLAN 1000. Since VLAN 1000 was defined as a community VLAN, this implies that R3 can talk to all other ports in VLAN 1000 and any promiscuous ports belonging to VLAN 100.

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show vlan private-vlan

Primary Secondary Type              Ports
------- --------- ----------------- ------------------------------------
100     1000      community         Fa0/1, Fa0/3
100     2000      community         Fa0/1, Fa0/5
100     3000      isolated          Fa0/1

Rack1SW2#show vlan private-vlan

Primary Secondary Type              Ports
------- --------- ----------------- ------------------------------------
100     1000      community         Fa0/2
100     2000      community         Fa0/4
100     3000      isolated          Fa0/6
```

Final verification for this configuration can be obtained by sending traffic to the broadcast address of 255.255.255.255 from all devices. As defined in the requirements R1 can talk to all routers, since it is a promiscuous port.

```
Rack1R1#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 100.0.0.2, 4 ms
Reply to request 0 from 100.0.0.4, 4 ms
Reply to request 0 from 100.0.0.3, 4 ms
Reply to request 0 from 100.0.0.5, 4 ms
Reply to request 0 from 100.0.0.6, 4 ms
```

R2 can talk to R3, who is in the same community, and R1 who is a promiscuous port.

```
Rack1R2#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 100.0.0.1, 4 ms
Reply to request 0 from 100.0.0.3, 4 ms
```

Copyright © 2010 Internetwork Expert

R3 can talk to R2, who is in the same community, and R1 who is a promiscuous port.

```
Rack1R3#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 100.0.0.1, 4 ms
Reply to request 0 from 100.0.0.2, 4 ms
```

R4 can talk to R5, who is in the same community, and R1 who is a promiscuous port.

```
Rack1R4#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 100.0.0.1, 4 ms
Reply to request 0 from 100.0.0.5, 4 ms
```

R5 can talk to R4, who is in the same community, and R1 who is a promiscuous port.

```
Rack1R5#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 100.0.0.1, 4 ms
Reply to request 0 from 100.0.0.4, 4 ms
```

Since R6 is an isolated port it can only talk to the promiscuous port, R1.

```
Rack1R6#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 100.0.0.1, 4 ms
```

Copyright © 2010 Internetwork Expert

1.48 PPP

- Enable PPP encapsulation for the Serial link connecting R4 and R5 and use the IP subnet 155.X.45.0/24 for this link
- R4 should attempt to authenticate R5 using PAP and then CHAP. R5 should refuse PAP authentication and use CHAP.
- Make sure R4 uses an alternate CHAP hostname R4CHAP.
- Use the name R5CHAP and the password of CISCO to accomplish this.
- R5 should authenticate R4 using PAP only. R4 should use the name R4PPP and the password of CISCO.

Configuration

Note

PPP (point-to-point protocol) is the universal layer 2 encapsulation method for Serial (i.e. point to point) links. The framing format has been derived from simpler HDLC but they main feature of PPP are two protocols LCP (link control protocol) and NCP (network control protocol). These two allow for flexible link configuration and protocol extension. For example LCP could be used to negotiate an authentication scheme, callback, link compression procedure, encryption and so on. NCP allows for running any layer 3 protocol on top of PPP by embedding the protocol's parameter negotiation into PPP. Those two features made PPP a very popular protocol for dialup, serial and other types of WAN links. PPP is also a very widely adopted VPN solution, as it could be tunneled over any other L2 or L3 protocol (e.g. PPPoE, PPPoA and L2TPv3 or PPTP even allow for running PPP across an IP cloud) and provide rich auto-configuration services.

Within the scope of CCIE R&S lab you may have to deal with PPP being run on top of dedicated Serial links, Ethernet segments (PPPoE) or other Frame-Relay (PPPoFR). In any case, many PPP configuration steps remain the same. In this scenario, we'll discuss one important aspect of LCP – link authentication, which is an optional feature that occurs after the link has been established via LCP by no L3 protocols have been negotiated via NCP. There are two common methods of authenticating (i.e. verifying the identity) of a remote endpoint – PAP (password authentication protocol) and CHAP (challenge and response protocol). During the LCP phase, the routers should agree on the authentication protocol using configuration requests. Here is how PAP and CHAP work:

1) Using PAP, the authenticator simply waits for the remote node to send a username and password pair in cleartext. If the pair matches the locally configured credentials, the authenticator declares the link to be authenticated.

Copyright © 2010 Internetwork Expert

2) Using CHAP, the authenticator prompts the remote node with a "challenge" strings and its own hostname. This allows the remote end to pick up the proper authentication key, based on the incoming hostname, and respond to the authenticator using the hash function calculated over the challenge string and the authentication key matching the authenticator. Notice that the authenticating host also includes its own hostname in the response as well so the authenticator may find the right key as well. After that, the authenticator performs the same hash calculation (keyed with the password value) and compares it to the response. If they match, the remote node is declared to be authenticated. This allows for secure authentication over unencrypted links.

To configure the router as a PPP authenticator, enable PPP encapsulation on the interface and use the command ppp authentication:

R4:

```
interface Serial 0/1/0
 encapsulation ppp
 ppp authentication pap chap
```

The command above means that R4 will offer PAP authentication and if the remote end rejects it will offer CHAP authentication. You may configure an endpoint to reject a particular authentication method offered using the command **ppp {pap|chap} refuse**. This command should be configured on the "authenticating" end, e.g. the one that is being offered an authentication method. Notice that the above configuration should be performed on the authenticator only.

Next, the authenticating end should be configured for PAP or CHAP credentials using either of two methods:

1) Using the global **command username <AUTHEN_NAME> password <PASSWORD>** which specifies a name/password pair matching the PPP ID of the authenticator router (it's the hostname). This method only works for CHAP as it's the method that is being initiated by the authenticator (see above). Notice the reverse logic – the username database is being inspected for outgoing credentials, not to match the incoming. This is the only way to configure multiple CHAP credentials, matching multiple remote hostnames.

Copyright © 2010 Internetwork Expert

2) Using the interface-level command **ppp sent-username <NAME> password <PWD>** to send outgoing PAP credentials when PAP is negotiated. This is the only way to define outgoing PAP credentials for the local end of the connection. For responding to incoming CHAP challenges, you may use the interface-level command **ppp chap password <PWD>**. This command defines the *default* password matching any incoming CHAP challenge or sent when there is no password found in the list of local usernames. The similarly looking command **ppp chap hostname <NAME>** defines an alternate CHAP hostname to be used in CHAP challenges or responses instead of the regular hostname.

Here is an example of configuring R5 for outgoing PAP credentials and a name/password pair for CHAP request:

R5:
```
username R4PPP password CISCO
!
interface Serial 0/1/0
 ppp pap sent-username R5PPP password CISCO
```

Keep in mind that a PPP endpoint could be configured as both the authenticator and the authenticating endpoint. In our scenario, both R4 and R5 authenticate each other, with R4 using CHAP to authenticate R5 and R5 using PAP to authenticate R4.

R4:
```
default interface Serial 0/1/0
interface Serial 0/1/0
 clock rate 128000
 encapsulation ppp
 ip address 155.1.45.4 255.255.255.0
 ppp authentication pap chap
 ppp pap sent-username R4PPP password CISCO
 ppp chap hostname R4CHAP
!
username R5CHAP password CISCO
!
```
R5:
```
default interface Serial 0/1/0
interface Serial 0/1/0
 clock rate 128000
 encapsulation ppp
 ip address 155.1.45.5 255.255.255.0
 ppp authentication pap
 ppp pap refuse
 ppp chap hostname R5CHAP
!
username R4PPP password CISCO
username R4CHAP password CISCO
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Enable PPP debugging and bring the Serial link down then back up.

```
RSRack1R5#debug ppp negotiation
PPP protocol negotiation debugging is on

RSRack1R5#debug ppp authentication
PPP authentication debugging is on

RSRack1R5#debug aaa authentication
AAA Authentication debugging is on
RSRack1R5#conf t
RSRack1R5(config)#interface serial 0/1/0
RSRack1R5(config-if)#no shutdown
RSRack1R5(config-if)#
Se0/1/0 PPP: Outbound cdp packet dropped
RSRack1R5(config-if)#
%LINK-3-UPDOWN: Interface Serial0/1/0, changed state to up
Se0/1/0 PPP: Using default call direction
Se0/1/0 PPP: Treating connection as a dedicated line
Se0/1/0 PPP: Session handle[2B000026] Session id[77]
```

PPP starts LCP trying to establish the link. You may see a series of CONF REQUESTS and CONF ACK or NACKS flowing down, negotiating various link options, including authentication.

```
Se0/1/0 PPP: Phase is ESTABLISHING, Active Open
Se0/1/0 PPP: Authorization required
Se0/1/0 LCP: O CONFREQ [Closed] id 2 len 14
Se0/1/0 LCP:    AuthProto PAP (0x0302C041)
Se0/1/0 LCP:    MagicNumber 0x1854B341 (0x05061854B341)
```

Copyright © 2010 Internetwork Expert

Incoming request for PAP gets rejected by the local end. CHAP is suggested instead. The remote end then ofers CHAP and the local end agrees.

```
Se0/1/0 LCP: I CONFREQ [REQsent] id 11 len 14
Se0/1/0 LCP:    AuthProto PAP (0x0302C041)
Se0/1/0 LCP:    MagicNumber 0x1A9C8EC2 (0x05061A9C8EC2)
Se0/1/0 LCP: O CONFNAK [REQsent] id 11 len 9
Se0/1/0 LCP:    AuthProto CHAP (0x0305C22305)

Se0/1/0 LCP: I CONFACK [REQsent] id 2 len 14
Se0/1/0 LCP:    AuthProto PAP (0x0302C041)
Se0/1/0 LCP:    MagicNumber 0x1854B341 (0x05061854B341)
Se0/1/0 LCP: I CONFREQ [ACKrcvd] id 12 len 15
Se0/1/0 LCP:    AuthProto CHAP (0x0305C22305)
Se0/1/0 LCP:    MagicNumber 0x1A9C8EC2 (0x05061A9C8EC2)

Se0/1/0 LCP: O CONFACK [ACKrcvd] id 12 len 15
Se0/1/0 LCP:    AuthProto CHAP (0x0305C22305)
Se0/1/0 LCP:    MagicNumber 0x1A9C8EC2 (0x05061A9C8EC2)
```

Link has been established, not the LCP Authentication phase begins. Notice that both ends perform authentication (AUTHENTICATING, by both). Next you may see I AUTH-REQ from R4 which means R4 starts sending its name and password continuously:

```
Se0/1/0 LCP: State is Open
Se0/1/0 PPP: Phase is AUTHENTICATING, by both
Se0/1/0 PAP: I AUTH-REQ id 1 len 16 from "R4PPP"
Se0/1/0 PAP: Authenticating peer R4PPP
Se0/1/0 PPP: Phase is FORWARDING, Attempting Forward
Se0/1/0 PPP: Phase is AUTHENTICATING, Unauthenticated User
AAA/AUTHEN/PPP (0000004B): Pick method list 'Permanent Local'
Se0/1/0 PPP: Sent PAP LOGIN Request
```

Copyright © 2010 Internetwork Expert

At the same time, R4 attempts to authenticate R5 by sending a CHAP challenge with its alternate hostname. This allows R5 to select the local key from the list of the usernames:

```
Se0/1/0 CHAP: I CHALLENGE id 5 len 27 from "R4CHAP"
AAA/AUTHEN/PPP (0000004B): Pick method list 'Permanent Local'
Se0/1/0 PPP: Received LOGIN Response PASS
Se0/1/0 PPP: Phase is FORWARDING, Attempting Forward
Se0/1/0 PPP: Phase is AUTHENTICATING, Authenticated User
Se0/1/0 PPP: Sent LCP AUTHOR Request
Se0/1/0 PPP: Sent IPCP AUTHOR Request
```

R5 finds the local key and prepares a CHAP response. The hostname in the response is taken from the interface configuraiton command.

```
Se0/1/0 CHAP: Using hostname from interface CHAP
Se0/1/0 CHAP: Using password from AAA
Se0/1/0 CHAP: O RESPONSE id 5 len 27 from "R5CHAP"
Se0/1/0 LCP: Received AAA AUTHOR Response PASS
Se0/1/0 IPCP: Received AAA AUTHOR Response PASS
```

R5 confirms R4's PAP credentials and authenticates R4. At the same time, R4 responds back that teh response to CHAP challenge was correct.

```
Se0/1/0 PAP: O AUTH-ACK id 1 len 5
Se0/1/0 CHAP: I SUCCESS id 5 len 4
```

Copyright © 2010 Internetwork Expert

At this point the link is authenticated, and the NCP phase (IPCP) starts.

```
Se0/1/0 PPP: Phase is UP
Se0/1/0 IPCP: O CONFREQ [Closed] id 1 len 10
Se0/1/0 IPCP:    Address 155.1.45.5 (0x03069B012D05)
Se0/1/0 PPP: Sent CDPCP AUTHOR Request
Se0/1/0 PPP: Process pending ncp packets
Se0/1/0 IPCP: I CONFREQ [REQsent] id 1 len 10
Se0/1/0 IPCP:    Address 155.1.45.4 (0x03069B012D04)
Se0/1/0 AAA/AUTHOR/IPCP: Start.  Her address 155.1.45.4, we want
0.0.0.0
Se0/1/0 PPP: Sent IPCP AUTHOR Request
Se0/1/0 CDPCP: I CONFREQ [Closed] id 1 len 4
Se0/1/0 AAA/AUTHOR/IPCP: Reject 155.1.45.4, using 0.0.0.0
Se0/1/0 AAA/AUTHOR/IPCP: Done.  Her address 155.1.45.4, we want 0.0.0.0
Se0/1/0 IPCP: O CONFACK [REQsent] id 1 len 10
Se0/1/0 IPCP:    Address 155.1.45.4 (0x03069B012D04)
Se0/1/0 IPCP: I CONFACK [ACKsent] id 1 len 10
Se0/1/0 IPCP:    Address 155.1.45.5 (0x03069B012D05)
Se0/1/0 IPCP: State is Open
Se0/1/0 CDPCP: Received AAA AUTHOR Response PASS
Se0/1/0 CDPCP: O CONFREQ [Closed] id 1 len 4
Se0/1/0 IPCP: Install route to 155.1.45.4
Se0/1/0 CDPCP: I CONFACK [REQsent] id 1 len 4
Se0/1/0 CDPCP: I CONFREQ [ACKrcvd] id 2 len 4
Se0/1/0 CDPCP: O CONFACK [ACKrcvd] id 2 len 4
Se0/1/0 CDPCP: State is Open
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

1.49 PPP AAA Authentication

- Reconfigure R4 so that it attempts to authenticate R5 against the RADIUS server at the IP address 155.X.146.100.
- Reconfigure R5 so that it attempts to authenticate R4 against the TACACS+ server at the IP address 155.X.146.200.
- In case if the remote AAA servers fail both R4 and R5 should use the local user database for authentication.
- Use the password value of CISCO to authenticate with the remote servers.
- Do not use global configuration commands to define a RADIUS server in R4.

Configuration

Note

Using remote AAA servers for PPP authentication is common in large-scale deployments. We are not going to cover AAA framework in depth here, as you can read about it referring to the **Security** section of this workbook. Configure AAA authentication for PPP is no different than configuring any other service authentication or authorization. The only significant difference is the fact that there is no real RADIUS or TACACS+ server in the CCIE R&S lab exam, so you have to deal with imaginary situation. Therefore, every method you configured for PPP should be configured to fall back to local authentication or any other backup method.

This particular scenario calls for creating of an AAA group, as it mentions that the AAA server should not be configured globally. To configure an AAA group use the command **aaa group server {radius|tacacs+} <NAME>**. You may then reference the group when creating an AAA list using the command similar to **aaa authentication ppp <AAA_GROUP> group <SERVER_GROUP>**. When populating the group, use the command **server** to refer a globally configured RADIUS or TACACS+ server or the command **server-private** to refer to the locally configured server, accessible only under the particular group.The other router, R5, is configured using the standard global RADIUS server configuration.

Copyright © 2010 Internetwork Expert

In order to configure a particular Serial interface to use a non-default PPP authentication or authorization group, use the interface-level command **ppp authentication {pap|chap} <AAA_GROUP>**. Notice that in most situations you may just resort to using the default AAA list for PPP configured using the command **aaa authentication ppp default {<SERVER_GROUP> |radius|tacacs+}**. When you specify just the **radius** or **tacacs+** keywords, the globally configured servers will be used in sequence they have been configured. You may want to use name lists when you have to use different AAA servers for multiple interfaces connected to the same router.

Pay attention to the special configuration needed to make sure the console line is not authenticated. By default enabling AAA will switch the console authentication mode to local authentication and might lock you out of the router.

R4:

```
aaa new-model
aaa authentication login CONSOLE none
aaa authentication ppp PPPAUTH group MYRADIUS local
!
aaa group server radius MYRADIUS
 server-private 155.1.146.100 key CISCO
!
interface Serial 0/1/0
 ppp authentication pap chap PPPAUTH
!
line console 0
 login authentication CONSOLE
```

R5:

```
aaa new-model
aaa authentication login CONSOLE none
aaa authentication ppp default group tacacs local
!
tacacs-server host 155.1.146.200 key CISCO
!
line console 0
 login authentication CONSOLE
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Enable RADIUS debugging in R4 in addition to the regular PPP debugging commands.

```
RSRack1R4#debug ppp negotiation
PPP protocol negotiation debugging is on

RSRack1R4#debug ppp authentication
PPP authentication debugging is on

RSRack1R4#debug radius
Radius protocol debugging is on
Radius protocol brief debugging is off
Radius protocol verbose debugging is off
Radius packet hex dump debugging is off
Radius packet protocol debugging is on
Radius elog debugging debugging is off
Radius packet retransmission debugging is off
Radius server fail-over debugging is off
Radius elog debugging debugging is off
```

Bounce R4's Serial interface and observe the flow of the debugging messages:

```
Se0/1/0 PPP: Sending Acct Event[Down] id[4C]
Se0/1/0 CDPCP: State is Closed
Se0/1/0 IPCP: State is Closed
Se0/1/0 PPP: Phase is TERMINATING
Se0/1/0 LCP: State is Closed
Se0/1/0 PPP: Phase is DOWN
Se0/1/0 IPCP: Remove route to 155.1.45.5
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0, changed
state to down
RSRack1R4(config-if)#no shutdown
RSRack1R4(config-if)#
Se0/1/0 PPP: Outbound cdp packet dropped
RSRack1R4(config-if)#
```

Copyright © 2010 Internetwork Expert

```
%LINK-3-UPDOWN: Interface Serial0/1/0, changed state to up
Se0/1/0 PPP: Using default call direction
Se0/1/0 PPP: Treating connection as a dedicated line
Se0/1/0 PPP: Session handle[B400003C] Session id[78]
Se0/1/0 PPP: Phase is ESTABLISHING, Active Open
Se0/1/0 PPP: Authorization NOT required
Se0/1/0 LCP: O CONFREQ [Closed] id 15 len 14
Se0/1/0 LCP:    AuthProto PAP (0x0302C041)
Se0/1/0 LCP:    MagicNumber 0x1AE0862A (0x05061AE0862A)
Se0/1/0 LCP: I CONFREQ [REQsent] id 4 len 14
Se0/1/0 LCP:    AuthProto PAP (0x0302C041)
Se0/1/0 LCP:    MagicNumber 0x1898AA5B (0x05061898AA5B)
Se0/1/0 LCP: O CONFACK [REQsent] id 4 len 14
Se0/1/0 LCP:    AuthProto PAP (0x0302C041)
Se0/1/0 LCP:    MagicNumber 0x1898AA5B (0x05061898AA5B)
Se0/1/0 LCP: I CONFNAK [ACKsent] id 15 len 9
Se0/1/0 LCP:    AuthProto CHAP (0x0305C22305)
Se0/1/0 LCP: O CONFREQ [ACKsent] id 16 len 15
Se0/1/0 LCP:    AuthProto CHAP (0x0305C22305)
Se0/1/0 LCP:    MagicNumber 0x1AE0862A (0x05061AE0862A)
Se0/1/0 LCP: I CONFACK [ACKsent] id 16 len 15
Se0/1/0 LCP:    AuthProto CHAP (0x0305C22305)
Se0/1/0 LCP:    MagicNumber 0x1AE0862A (0x05061AE0862A)
Se0/1/0 LCP: State is Open
```

This is where authentication phase starts.

```
Se0/1/0 PPP: Phase is AUTHENTICATING, by both
Se0/1/0 PAP: Using hostname from interface PAP
Se0/1/0 PAP: Using password from interface PAP
Se0/1/0 PAP: O AUTH-REQ id 3 len 16 from "R4PPP"
Se0/1/0 CHAP: O CHALLENGE id 7 len 27 from "R4CHAP"
Se0/1/0 PAP: I AUTH-ACK id 3 len 5
Se0/1/0 CHAP: I RESPONSE id 7 len 27 from "R5CHAP"
Se0/1/0 PPP: Phase is FORWARDING, Attempting Forward
Se0/1/0 PPP: Phase is AUTHENTICATING, Unauthenticated User
Se0/1/0 PPP: Sent CHAP LOGIN Request
```

Copyright © 2010 Internetwork Expert

From the above output we can see that R4 sends CHAP challenge and receives CHAP response. Next we see RADIUS packets being sent to the server.

```
RADIUS/ENCODE(0000004D):Orig. component type = PPP
RADIUS:  AAA Unsupported Attr: interface         [175] 11
RADIUS:   53 65 72 69 61 6C 30 2F 31                        [Serial0/1]
RADIUS(0000004D): Config NAS IP: 0.0.0.0
RADIUS/ENCODE(0000004D): acct_session_id: 74
RADIUS(0000004D): sending
RADIUS/ENCODE: Best Local IP-Address 155.1.45.4 for Radius-Server
155.1.146.100
RADIUS(0000004D): Send Access-Request to 155.1.146.100:1645 id 1645/1,
len 90
RADIUS:  authenticator F5 24 28 26 F5 46 99 F6 - 2B B5 BD 5A 64 B8 7C
F9
RADIUS:  Framed-Protocol      [7]   6   PPP                       [1]
RADIUS:  User-Name            [1]   8   "R5CHAP"
RADIUS:  CHAP-Password        [3]   19  *
RADIUS:  NAS-Port             [5]   6   10000
RADIUS:  NAS-Port-Id          [87]  13  "Serial0/1/0"
RADIUS:  NAS-Port-Type        [61]  6   Sync                      [1]
RADIUS:  Service-Type         [6]   6   Framed                    [2]
RADIUS:  NAS-IP-Address       [4]   6   155.1.45.4
```

The router keeps retransmitting the packets until it declares the server dead and switches back to local authentication.

```
RADIUS: Retransmit to (155.1.146.100:1645,1646) for id 1645/1
RADIUS: Retransmit to (155.1.146.100:1645,1646) for id 1645/1
Se0/1/0 CHAP: I RESPONSE id 7 len 27 from "R5CHAP"
Se0/1/0 CHAP: Ignoring Additional Response
Se0/1/0 AUTH: Timeout 1

RADIUS: Retransmit to (155.1.146.100:1645,1646) for id 1645/1
RADIUS: No response from (155.1.146.100:1645,1646) for id 1645/1
RADIUS/DECODE: No response from radius-server; parse response; FAIL
RADIUS/DECODE: Case error(no response/ bad packet/ op decode);parse
response; FAIL
```

Copyright © 2010 Internetwork Expert

```
Se0/1/0 PPP: Received LOGIN Response PASS
Se0/1/0 PPP: Phase is FORWARDING, Attempting Forward
Se0/1/0 PPP: Phase is AUTHENTICATING, Authenticated User
Se0/1/0 CHAP: O SUCCESS id 7 len 4
Se0/1/0 PPP: Phase is UP
Se0/1/0 IPCP: O CONFREQ [Closed] id 1 len 10
Se0/1/0 IPCP:    Address 155.1.45.4 (0x03069B012D04)
Se0/1/0 CDPCP: O CONFREQ [Closed] id 1 len 4
Se0/1/0 PPP: Process pending ncp packets
Se0/1/0 IPCP: I CONFREQ [REQsent] id 1 len 10
Se0/1/0 IPCP:    Address 155.1.45.5 (0x03069B012D05)
Se0/1/0 IPCP: O CONFACK [REQsent] id 1 len 10
Se0/1/0 IPCP:    Address 155.1.45.5 (0x03069B012D05)
Se0/1/0 CDPCP: I CONFREQ [REQsent] id 1 len 4
Se0/1/0 CDPCP: O CONFACK [REQsent] id 1 len 4
Se0/1/0 IPCP: I CONFACK [ACKsent] id 1 len 10
Se0/1/0 IPCP:    Address 155.1.45.4 (0x03069B012D04)
Se0/1/0 IPCP: State is Open
Se0/1/0 IPCP: Install route to 155.1.45.5
Se0/1/0 CDPCP: I CONFACK [ACKsent] id 1 len 4
Se0/1/0 CDPCP: State is Open
```

Copyright © 2010 Internetwork Expert

Now observe the fallback from TACACS+ authentication to the local database in R5:

```
RSRack1R5#debug ppp authentication
PPP authentication debugging is on

RSRack1R5#debug ppp negotiation
PPP protocol negotiation debugging is on

RSRack1R5#debug aaa authentication
AAA Authentication debugging is on

RSRack1R5#debug tacacs
TACACS access control debugging is on

RSRack1R5#debug tacacs packet
TACACS+ packets debugging is on

AAA/BIND(0000004E): Bind i/f Serial0/1/0
Se0/1/0 PPP: Using default call direction
Se0/1/0 PPP: Treating connection as a dedicated line
Se0/1/0 PPP: Session handle[FD000029] Session id[80]
Se0/1/0 PPP: Phase is ESTABLISHING, Active Open
Se0/1/0 PPP: Authorization NOT required
Se0/1/0 LCP: O CONFREQ [Closed] id 5 len 14
Se0/1/0 LCP:    AuthProto PAP (0x0302C041)
Se0/1/0 LCP:    MagicNumber 0x18A58BB5 (0x050618A58BB5)
Se0/1/0 LCP: I CONFREQ [REQsent] id 17 len 14
Se0/1/0 LCP:    AuthProto PAP (0x0302C041)
Se0/1/0 LCP:    MagicNumber 0x1AED6797 (0x05061AED6797)
Se0/1/0 LCP: O CONFNAK [REQsent] id 17 len 9
Se0/1/0 LCP:    AuthProto CHAP (0x0305C22305)
Se0/1/0 LCP: I CONFACK [REQsent] id 5 len 14
Se0/1/0 LCP:    AuthProto PAP (0x0302C041)
Se0/1/0 LCP:    MagicNumber 0x18A58BB5 (0x050618A58BB5)
Se0/1/0 LCP: I CONFREQ [ACKrcvd] id 18 len 15
Se0/1/0 LCP:    AuthProto CHAP (0x0305C22305)
Se0/1/0 LCP:    MagicNumber 0x1AED6797 (0x05061AED6797)
Se0/1/0 LCP: O CONFACK [ACKrcvd] id 18 len 15
Se0/1/0 LCP:    AuthProto CHAP (0x0305C22305)
Se0/1/0 LCP:    MagicNumber 0x1AED6797 (0x05061AED6797)
Se0/1/0 LCP: State is Open
```

Authentication starts here. R4 sends loging credentials for PAP and R5 attempts to contact the TACACS+ server. The server is not accessible and thus R5 falls back to local authentication.

Copyright © 2010 Internetwork Expert

```
Se0/1/0 PPP: Phase is AUTHENTICATING, by both
Se0/1/0 PAP: I AUTH-REQ id 4 len 16 from "R4PPP"
Se0/1/0 PAP: Authenticating peer R4PPP
Se0/1/0 PPP: Phase is FORWARDING, Attempting Forward
Se0/1/0 PPP: Phase is AUTHENTICATING, Unauthenticated User
AAA/AUTHEN/PPP (0000004E): Pick method list 'default'
Se0/1/0 PPP: Sent PAP LOGIN Request
TPLUS: Queuing AAA Authentication request 78 for processing
Se0/1/0 CHAP: I CHALLENGE id 8 len 27 from "R4CHAP"

AAA/AUTHEN/PPP (0000004E): Pick method list 'default'
TPLUS: processing authentication start request id 78
TPLUS: Authentication start packet created for 78(R4PPP)
TPLUS: Using server 155.1.146.200
TPLUS(0000004E)/0: Connect Error No route to host
TPLUS: Queuing AAA Authentication request 78 for processing
TPLUS: processing authentication start request id 78
TPLUS: Authentication start packet created for 78(R4CHAP)
TPLUS: Using server 155.1.146.200
TPLUS(0000004E)/0: Connect Error No route to host

Se0/1/0 PPP: Received LOGIN Response PASS
Se0/1/0 PPP: Phase is FORWARDING, Attempting Forward
Se0/1/0 CHAP: Using hostname from interface CHAP
Se0/1/0 CHAP: Using password from AAA
Se0/1/0 CHAP: O RESPONSE id 8 len 27 from "R5CHAP"
Se0/1/0 PPP: Phase is AUTHENTICATING, Authenticated User
Se0/1/0 PAP: O AUTH-ACK id 4 len 5
Se0/1/0 AUTH: Timeout 1
Se0/1/0 CHAP: Using hostname from interface CHAP
Se0/1/0 CHAP: Using password from AAA
Se0/1/0 CHAP: O RESPONSE id 8 len 27 from "R5CHAP"
Se0/1/0 CHAP: I SUCCESS id 8 len 4
Se0/1/0 PPP: Phase is UP
Se0/1/0 IPCP: O CONFREQ [Closed] id 1 len 10
Se0/1/0 IPCP:    Address 155.1.45.5 (0x03069B012D05)
Se0/1/0 CDPCP: O CONFREQ [Closed] id 1 len 4
Se0/1/0 PPP: Process pending ncp packets
Se0/1/0 IPCP: I CONFREQ [REQsent] id 1 len 10
Se0/1/0 IPCP:    Address 155.1.45.4 (0x03069B012D04)
Se0/1/0 IPCP: O CONFACK [REQsent] id 1 len 10
Se0/1/0 IPCP:    Address 155.1.45.4 (0x03069B012D04)
Se0/1/0 CDPCP: I CONFREQ [REQsent] id 1 len 4
Se0/1/0 CDPCP: O CONFACK [REQsent] id 1 len 4
Se0/1/0 IPCP: I CONFACK [ACKsent] id 1 len 10
Se0/1/0 IPCP:    Address 155.1.45.5 (0x03069B012D05)
Se0/1/0 IPCP: State is Open
Se0/1/0 IPCP: Install route to 155.1.45.4
Se0/1/0 CDPCP: I CONFACK [ACKsent] id 1 len 4
Se0/1/0 CDPCP: State is Open
```

Copyright © 2010 Internetwork Expert

1.50 PPPoE

- Configure R3 as a PPPoE client and R5 as the PPPoE server.
- Use R3's interface Fa0/1 and R5's sub-interface Fa0/1.54 to accomplish this.
- R5 should block any host attempting to initiate more than 10 sessions in a minute for 5 minutes.
- You are allowed to create an additional VLAN 54 for this task.
- R5 should allocate an IP address to R3 from the local DHCP pool for the subnet 155.X.35.0.
- Make sure R3 is assigned the IP address 155.X.35.3 from the pool.
- R5 should authenticate R3 using CHAP and the password value of CISCO.

Configuration

Note

One of the widest uses for PPP is broadcast access, used in DSL and cable networks. PPP allows for replicating the single-user/single-session paradigm on multiple-access networks, which greatly simplifies deployment as it allows for reusing the existing billing systems and expertise.

The most common broadband PPP implementation is PPP over Ethernet or PPPoE. Essentially it's just a PPP session being run on top of Ethernet segment inside the regular frames. Since PPP as a point-to-point protocol and Ethernet is a multiple-access segment, an extension is needed to discover the server willing to terminate the PPP link. When a PPPoE client starts, it enters the discovery phase by sending out a broadcast PADI (PPPoE Active Discovery Initiation) packet. If there is a PPPoE server on the segment it responds with a PADO (PPPoE Active Discovery Offer Packet) to the MAC address of the client initiating the session. If the client accepts the offer, it sends out a Request packet (much like during DHCP negotiation) to the server requesting the PPP service. If the server agrees to the request, it responds with a PADS (Session-confirmation) packet and the client and server start exchanging PPP packets inside Ethernet frames performing normal PPP session negotiation.

Copyright © 2010 Internetwork Expert

In the older IOS versions configuring PPPoE server would require using **vpdn** (virtual prviaten dialin network) commands. The recent IOS versions introduced new syntax just for PPPoE configuration using the command mode **bba-group** which stands for broadband aggregation group. You may configure the default BBA group – the global one which applies to all interfaces configured for PPPoE termination or created named groups tuned for various settings and applied to different interfaces. You may limit the amount of incoming PPPoE sessions per interface/sub-interface using the command **pppoe max-sessions**. Aside from the optional features, there is probably just one mandatory configuration settings – configuring a virtual-template interface and associating it with the bba-group. The virtual-template is used to clone the actual virtual-access interfaces for every connecting user. You may define PPP authentication settings and other native PPP options under the virtual-template interface.

The PPPoE client configuration in the IOS router is straightforward – you define a dialer interface, configure it for PPP and addressing and bind it to a physical interface using dialer pool. You may see the example in the solution below.

Pay attention to the special "trick" we're using to allocate the IP address to the connecting client via DHCP. The DHCP pool is configured with an exclusion to allocate only a single desired IP address. Notice that only the client is configured with the statement **ip address dhcp** – there is no need to configure any **peer default ip address** settings in the server interface, as this engages IPCP address allocation mechanism, which is not needed with DHCP.

Since PPP is being run over Ethernet, MTU limitations arise. The default Ethernet MTU is 1500 and the PPPoE header adds extra 8 bytes to PPP frames encapsulated inside Ethernet frames. Normally, a client host like PC would account for the overhead and properly set the MSS option in outgoing TCP connections. However, if a router is used as a PPPoE client, the following issues may arise:

1) Some IGP protocols e.g. OSPF may fail to negotiate across the virtual link due to MTU mismatch. This could be fixed using the command **ip ospf mtu-ignore** with OSPF or by setting the MTU value manually at both endpoints.

2) PMTU discovery may fail due to firewall filtering and TCP sessions may not work properly. This could be fixed using the command **ip tcp adjust-mss 1452** (or lower) on the virtual interface (e.g. on the dialer) in the client router.

Copyright © 2010 Internetwork Expert

R3:

```
interface Dialer1
 ip address dhcp
 encapsulation ppp
 dialer pool 1
 ppp chap hostname R3PPP
 ppp chap password CISCO
!
interface FastEthernet0/1
 no ip address
 no shutdown
 pppoe enable
 pppoe-client dial-pool-number 1
```

R5:

```
aaa authentication ppp PPPOE local
username R3PPP password CISCO
!
ip dhcp excluded-address 155.1.35.1 155.1.35.2
ip dhcp excluded-address 155.1.35.4 155.1.35.254
!
ip dhcp pool PPPOE
   network 155.1.35.0 255.255.255.0
!
interface FastEthernet 0/1
 no shutdown
!
interface FastEthernet0/1.35
 encapsulation dot1Q 35
 pppoe enable group PPPOE

!
! Define per-MAC throttling settings under the BBA group
!
bba-group pppoe PPPOE
 virtual-template 1
 sessions per-mac throttle 10 60 300
!
interface Virtual-Template 1
 encapsulation ppp
 ip address 155.1.35.1 255.255.255.0
 ppp authentication chap PPPOE
```

SW3:

```
vlan 35
!
interface fastEthernet 0/3
 switchport host
 switchport access vlan 35
!
interface FastEthernet 0/5
 switchport trunk encapsulation dot1q
 switchport mode trunk
 spanning-tree portfast trunk
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Debug PPPoE packets exchange along with the PPP session packets in R4. After this, clear the active PPPoE sessions.

```
RSRack1R3#debug pppoe packets
PPPoE control packets debugging is on

RSRack1R3#debug pppoe events
PPPoE protocol events debugging is on

RSRack1R3#debug ppp negotiation
PPP protocol negotiation debugging is on

RSRack1R3#clear pppoe all
RSRack1R3#
```

The session is being terminated using PADT packet:

```
Vi1 PPP: Outbound cdp packet dropped
[0]PPPoE 16: O PADT  R:0013.1a68.b04d L:0011.2093.e561 Fa0/1
         00 13 1A 68 B0 4D 00 11 20 93 E5 61 88 63 11 A7
         00 10 00 00 00 00 00 00 00 00 00 00 00 00 00 ...
Vi1 IPCP: Remove link info for cef entry 155.1.35.1

%DIALER-6-UNBIND: Interface Vi1 unbound from profile Di1

[0]PPPoE 16: Destroying  R:0013.1a68.b04d L:0011.2093.e561 Fa0/1
PPPoE: Returning Vaccess Virtual-Access1
Vi1 PPP: Block vaccess from being freed [0x10]
Di1 IPCP: Remove route to 155.1.35.1
```

The router sends initial PADI packet out:

```
 Sending PADI: Interface = FastEthernet0/1
pppoe_send_padi:
         FF FF FF FF FF FF 00 11 20 93 E5 61 88 63 11 09
         00 00 00 0C 01 01 00 00 01 03 00 04 84 4F 7E 70 ...
%LINK-3-UPDOWN: Interface Virtual-Access1, changed state to down
RSRack1R3#
```

Copyright © 2010 Internetwork Expert

```
Vi1 PPP: Sending Acct Event[Down] id[3]
Vi1 CDPCP: State is Closed
Vi1 IPCP: State is Closed
Vi1 PPP: Unlocked by [0x10] Still Locked by [0x2]
Vi1 PPP: Phase is TERMINATING
Vi1 LCP: State is Closed
Vi1 PPP: Phase is DOWN
Vi1 PPP SSS: Send DISCONNECT to mgr_hdl[BA000001]
Vi1 PPP: Unlocked by [0x2] Still Locked by [0x0]
Vi1 PPP: Free previously blocked vaccess
```

The server responds with an offer and the client requests a session. The server finally confirms the session:

```
PPPoE 0: I PADO  R:0013.1a68.b04d L:0011.2093.e561 Fa0/1
         00 11 20 93 E5 61 00 13 1A 68 B0 4D 88 63 11 07
         00 00 00 2D 01 01 00 00 01 03 00 04 84 4F 7E 70 ...
RSRack1R3#

%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access1,
changed state to down

OUT PADR from PPPoE Session
         00 13 1A 68 B0 4D 00 11 20 93 E5 61 88 63 11 19
         00 00 00 2D 01 01 00 00 01 03 00 04 84 4F 7E 70 ...
PPPoE 17: I PADS  R:0013.1a68.b04d L:0011.2093.e561 Fa0/1
         00 11 20 93 E5 61 00 13 1A 68 B0 4D 88 63 11 65
         00 11 00 2D 01 01 00 00 01 03 00 04 84 4F 7E 70 ...

IN PADS from PPPoE Session

%DIALER-6-BIND: Interface Vi1 bound to profile Di1
PPPoE: Virtual Access interface obtained.
PPPoE : encap string prepared
[0]PPPoE 17: data path set to Virtual Acess
Vi1 PPP: Phase is DOWN, Setup
Vi1 PPP: Using dialer call direction
Vi1 PPP: Treating connection as a callout
Vi1 PPP: Session handle[8E000008] Session id[0]
```

Now the regular PPP session starts. Notice one special configuration request of MRU (maximum receive unit) of 1492 which is because of the PPPoE header of 8 bytes. The server still suggests MRU of 1500 because it does not account for PPPoE overhead.

Copyright © 2010 Internetwork Expert

```
Vi1 PPP: Phase is ESTABLISHING, Active Open
Vi1 PPP: No remote authentication for call-out
Vi1 LCP: O CONFREQ [Closed] id 1 len 10
Vi1 LCP:    MagicNumber 0x170E8640 (0x0506170E8640)
%LINK-3-UPDOWN: Interface Virtual-Access1, changed state to up
Vi1 LCP: I CONFREQ [REQsent] id 1 len 19
Vi1 LCP:    MRU 1492 (0x010405D4)
Vi1 LCP:    AuthProto CHAP (0x0305C22305)
Vi1 LCP:    MagicNumber 0x190C2819 (0x0506190C2819)
Vi1 LCP: O CONFNAK [REQsent] id 1 len 8
Vi1 LCP:    MRU 1500 (0x010405DC)
Vi1 LCP: I CONFACK [REQsent] id 1 len 10
Vi1 LCP:    MagicNumber 0x170E8640 (0x0506170E8640)
Vi1 LCP: I CONFREQ [ACKrcvd] id 2 len 19
Vi1 LCP:    MRU 1500 (0x010405DC)
Vi1 LCP:    AuthProto CHAP (0x0305C22305)
Vi1 LCP:    MagicNumber 0x190C2819 (0x0506190C2819)
Vi1 LCP: O CONFACK [ACKrcvd] id 2 len 19
Vi1 LCP:    MRU 1500 (0x010405DC)
Vi1 LCP:    AuthProto CHAP (0x0305C22305)
Vi1 LCP:    MagicNumber 0x190C2819 (0x0506190C2819)
Vi1 LCP: State is Open
```

This is where R5 authenticates R3 using CHAP:

```
Vi1 PPP: Phase is AUTHENTICATING, by the peer
Vi1 CHAP: I CHALLENGE id 1 len 30 from "RSRack1R5"
Vi1 CHAP: Using hostname from interface CHAP
Vi1 CHAP: Using password from interface CHAP
Vi1 CHAP: O RESPONSE id 1 len 26 from "R3PPP"
Vi1 CHAP: I SUCCESS id 1 len 4
Vi1 PPP: Phase is FORWARDING, Attempting Forward
Vi1 PPP: Queue IPCP code[1] id[1]
Vi1 PPP SSS: Receive SSS-Mgr Connect-Local
Vi1 PPP: Phase is ESTABLISHING, Finish LCP
Vi1 PPP: Phase is UP
Vi1 IPCP: O CONFREQ [Closed] id 1 len 10
Vi1 IPCP:    Address 155.1.35.3 (0x03069B012303)
Vi1 CDPCP: O CONFREQ [Closed] id 1 len 4
Vi1 PPP: Process pending ncp packets
Vi1 IPCP: Redirect packet to Vi1
Vi1 IPCP: I CONFREQ [REQsent] id 1 len 10
Vi1 IPCP:    Address 155.1.35.1 (0x03069B012301)
Vi1 IPCP: O CONFACK [REQsent] id 1 len 10
Vi1 IPCP:    Address 155.1.35.1 (0x03069B012301)
Vi1 LCP: I PROTREJ [Open] id 3 len 10 protocol CDPCP (0x820701010004)
Vi1 CDPCP: State is Closed
Vi1 CDPCP: State is Listen
Vi1 IPCP: I CONFACK [ACKsent] id 1 len 10
Vi1 IPCP:    Address 155.1.35.3 (0x03069B012303)
Vi1 IPCP: State is Open
Di1 IPCP: Install route to 155.1.35.1
Vi1 IPCP: Add link info for cef entry 155.1.35.1
[0]PPPoE 17: Vi1 O FS
```

Copyright © 2010 Internetwork Expert

Check the session status in R3 and R5 after everything has been established:

```
RSRack1R3#show pppoe session
     1 client session

Uniq ID  PPPoE  RemMAC          Port                 Source   VA
State
           SID  LocMAC                                       VA-st
    N/A     17  0013.1a68.b04d  Fa0/1                Di1      Vi1
UP
               0011.2093.e561                                 UP


RSRack1R5#show pppoe session
     1 session  in LOCALLY_TERMINATED (PTA) State
     1 session  total

Uniq ID  PPPoE  RemMAC          Port                 Source   VA
State
           SID  LocMAC                                       VA-st
     98     17  0011.2093.e561  Fa0/1.35             Vt1      Vi1.1
PTA
               0013.1a68.b04d  VLAN :35                    UP

RSRack1R5#ping 155.1.35.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.35.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

Copyright © 2010 Internetwork Expert

Frame Relay Solutions

> **Note**
>
> Erase and reload all devices to a blank configuration prior to starting.

2.1 Inverse-ARP

- Configure the IP address of R6's Serial interface using the information in the diagram.
- Enable Frame Relay on this link.
- Ensure R6 learns an IP to DLCI mapping dynamically for PVC 101 connected to BB1, and that these devices have unicast, multicast, and broadcast reachability to each other.

Configuration

```
R6:
interface Serial0/0
 encapsulation frame-relay
 ip address 54.1.1.6 255.255.255.0
```

Verification

> **Note**
>
> Frame-Relay is a non-broadcast multi-access (NBMA) media, which means that there is no layer 2 address that maps to all devices in the network, like FFFF:FFFF:FFFF does in Ethernet. This means that for interfaces that can have multiple layer 2 destinations, i.e. the multipoint main interface or multipoint subinterface, layer 3 to layer 2 resolution must be performed either manually with the **`frame-relay map command`** or automatically through Frame Relay Inverse-ARP.

Copyright © 2010 Internetwork Expert

Once the **encapsulation frame-relay** command is issued at the interface level the router will learn all DLCIs provisioned by the service provider through LMI. Without other configuration all DLCIs will be automatically assigned to the main interface when learned. The first verification for this is to issue the **show frame-relay pvc** command. If the PVC status is *active*, then frames can be sent and received over it.

```
Rack1R6#show frame-relay pvc 101

PVC Statistics for interface Serial0/0 (Frame Relay DTE)

DLCI = 101, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0

  input pkts 218           output pkts 25           in bytes 24901
  out bytes 1802           dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 3         out bcast bytes 102
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 00:10:21, last time pvc status changed 00:01:10
```

Once a layer 3 protocol is configured on the interface, such as IPv4, Inverse-ARP requests will automatically be sent out all DLCIs learned via LMI. If the device on the other end of the link is running that protocol stack, a reply will be received indicating their layer 3 address that is assigned on that circuit. Verification for this can be performed with the **show frame-relay map** command. Note that the *broadcast* keyword appears in the below show output. This indicates that Inverse-ARP automatically supports the sending of broadcast and multicast frames as replicated unicasts out the mapped circuit.

```
Rack1R6#show frame-relay map
Serial0/0 (up): ip 54.1.1.254 dlci 101(0x65,0x1850), dynamic,
             broadcast,, status defined, active
Serial0/0 (up): ip 54.1.2.254 dlci 100(0x64,0x1840), dynamic,
             broadcast,, status defined, active
Serial0/0 (up): ip 54.1.3.254 dlci 51(0x33,0xC30), dynamic,
             broadcast,, status defined, active
```

Copyright © 2010 Internetwork Expert

To verify that transport is achieved it's a good idea at this point to test both unicast and multicast/broadcast reachability. Unicast can be tested with a simple ICMP ping, while multicast/broadcast transport can be tested by pinging the all hosts broadcast address of 255.255.255.255.

```
Rack1R6#ping 54.1.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 54.1.1.254, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/36 ms

Rack1R6#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 54.1.2.254, 76 ms
Reply to request 0 from 54.1.1.254, 108 ms
Reply to request 0 from 54.1.3.254, 92 ms
```

Pitfall

Note that R6 learns dynamic mappings for IP addresses not on the same subnet as configured on the interface. This is because multiple PVCs are learned from the Frame Relay network through LMI, and Inverse-ARP requests are sent out all of them. The result of this is seen when broadcast traffic is sent out the interface. Even though R6 does not have the subnet 54.1.3.0/24 assigned it received a reply from the broadcast request sent out to that circuit. To fix this behavior Inverse-ARP should be disabled on the other circuits (not PVC 101), or PVC 101 should be assigned to a subinterface. The ideal design solution would be to use a subinterface to avoid this problem.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

2.2 Static Mappings

- Configure the IP addresses of R1 and R5's Serial interfaces connected to the Frame Relay cloud using the information in the diagram.
- Enable Frame Relay on this link and configure static IP to DLCI mappings so that R1 and R5 have unicast, multicast, and broadcast reachability to each other.

Configuration

```
R1:
interface Serial0/0
 encapsulation frame-relay
 ip address 155.1.0.1 255.255.255.0
 frame-relay map ip 155.1.0.5 105 broadcast

R5:
interface Serial0/0
 encapsulation frame-relay
 ip address 155.1.0.5 255.255.255.0
 frame-relay map ip 155.1.0.1 501 broadcast
```

Verification

Note

Static Frame Relay mappings override any dynamically learned Inverse-ARP mappings. If multipoint interfaces are used in your configuration, i.e. the main interface or a multipoint subinterface, static **`frame-relay map`** commands would be the ideal design solution, as sometimes the Inverse-ARP protocol is unreliable and dynamic mappings disappear. Note that broadcast and multicast transport is not automatically enabled unless the *broadcast* keyword is added at the end of the **`frame-relay map`** command.

```
Rack1R1#show frame-relay map
Serial0/0 (up): ip 155.1.0.5 dlci 105(0x69,0x1890), static,
              broadcast,
              CISCO, status defined, active

Rack1R5#show frame-relay map
Serial0/0 (up): ip 155.1.0.1 dlci 501(0x1F5,0x7C50), static,
              broadcast,
              CISCO, status defined, active
```

Copyright © 2010 Internetwork Expert

Just like using dynamic mappings it's a good idea at this point to test both unicast and multicast/broadcast reachability. Unicast can be tested with a simple ICMP ping, while multicast/broadcast transport can be tested by pinging the all hosts broadcast address of 255.255.255.255.

```
Rack1R1#ping 155.1.0.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/60 ms

Rack1R1#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2 seconds:

Reply to request 0 from 155.1.0.5, 116 ms

Rack1R5#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2 seconds:

Reply to request 0 from 155.1.0.1, 60 ms
```

Copyright © 2010 Internetwork Expert

2.3 Point-to-Point Subinterfaces

- Enable Frame Relay on R2's link connecting to the Frame Relay cloud.
- Create a point-to-point subinterface numbered 205 on R2 with the IP address 155.X.25.2/24 and assign DLCI 205 to it.
- Create a point-to-point subinterface numbered 502 on R5 with the IP address 155.X.25.5/24 and assign DLCI 502 to it.
- Ensure that these devices have unicast, multicast, and broadcast reachability to each other on this segment.

Configuration

```
R2:
interface Serial0/0
 encapsulation frame-relay
!
interface Serial0/0.205 point-to-point
 ip address 155.1.25.2 255.255.255.0
 frame-relay interface-dlci 205

R5:
interface Serial0/0
 encapsulation frame-relay
!
interface Serial0/0.502 point-to-point
 ip address 155.1.25.5 255.255.255.0
 frame-relay interface-dlci 502
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Point-to-point Frame Relay interfaces, such as point-to-point subinterfaces or PPP over Frame Relay interfaces, do not require layer 3 to layer 2 resolution because there is only one possible layer 2 destination reachable out the link. As long as the circuit is assigned with the **`frame-relay interface-dlci`** command all unicast, multicast, or broadcast traffic for the configured layer 3 protocol can be encapsulated.
>
> This type of configuration would be the ideal design solution for running Frame Relay, as complex dynamic or static Frame Relay mappings are not required. Using point-to-point interfaces will also solve many other layer 3 design issues with protocols such as OSPF, Multicast, and IPv6. These issues are covered in depth in those particular protocol chapters of the workbook.

```
Rack1R2#show frame-relay map
Serial0/0.205 (up): point-to-point dlci, dlci 205(0xCD,0x30D0),
broadcast
          status defined, active

Rack1R5#show frame-relay map
Serial0/0 (up): ip 155.1.0.1 dlci 501(0x1F5,0x7C50), static,
              broadcast,
              CISCO, status defined, active
Serial0/0.502 (up): point-to-point dlci, dlci 502(0x1F6,0x7C60),
broadcast
          status defined, active

Rack1R2#ping 155.1.25.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.25.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/60 ms

Rack1R2#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 155.1.25.5, 128 ms
```

Copyright © 2010 Internetwork Expert

2.4 Multipoint Subinterfaces & Inverse-ARP

- Enable Frame Relay on R3 and R4's connections to the Frame Relay cloud.
- Create a multipoint subinterface numbered 100 on R3 with the IP address 155.X.100.3/24 and assign DLCIs 304 and 305 to it.
- Create a multipoint subinterface numbered 100 on R4 with the IP address 155.X.100.4/24 and assign DLCIs 403 and 405 to it.
- Create a multipoint subinterface numbered 100 on R5 with the IP address 155.X.100.5/24 and assign DLCIs 503 and 504 to it.
- Ensure that R3, R4, and R5 all learn dynamic IP to DLCI mappings for each other on this segment and have unicast, multicast, and broadcast reachability to each other.

Configuration

```
R3:
interface Serial1/0
 encapsulation frame-relay
!
interface Serial1/0.100 multipoint
 ip address 155.1.100.3 255.255.255.0
 frame-relay interface-dlci 304
 frame-relay interface-dlci 305

R4:
interface Serial0/0
 encapsulation frame-relay
!
interface Serial0/0.100 multipoint
 ip address 155.1.100.4 255.255.255.0
 frame-relay interface-dlci 403
 frame-relay interface-dlci 405

R5:
interface Serial0/0
 encapsulation frame-relay
!
interface Serial0/0.100 multipoint
 ip address 155.1.100.5 255.255.255.0
 frame-relay interface-dlci 503
 frame-relay interface-dlci 504
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Like the main interface, multipoint subinterfaces require layer 3 to layer 2 resolution since there are multiple possible layer 2 destinations. Typically this configuration is done with the static **`frame-relay map`** commands, however it can also be done dynamically with Inverse-ARP like in this example. By assigning the DLCI to the interface with the **`frame-relay interface-dlci`** command, Inverse-ARP request will automatically be sent out the PVCs for the configured protocol stack, IPv4.

```
Rack1R3#show frame-relay map
Serial1/0.100 (up): ip 155.1.100.4 dlci 304(0x130,0x4C00), dynamic,
              broadcast,, status defined, active
Serial1/0.100 (up): ip 155.1.100.5 dlci 305(0x131,0x4C10), dynamic,
              broadcast,, status defined, active

Rack1R4#show frame-relay map
Serial0/0.100 (up): ip 155.1.100.3 dlci 403(0x193,0x6430), dynamic,
              broadcast,, status defined, active
Serial0/0.100 (up): ip 155.1.100.5 dlci 405(0x195,0x6450), dynamic,
              broadcast,, status defined, active

Rack1R5#show frame-relay map
Serial0/0.100 (up): ip 155.1.100.3 dlci 503(0x1F7,0x7C70), dynamic,
              broadcast,, status defined, active
Serial0/0.100 (up): ip 155.1.100.4 dlci 504(0x1F8,0x7C80), dynamic,
              broadcast,, status defined, active
Serial0/0 (up): ip 155.1.0.1 dlci 501(0x1F5,0x7C50), static,
              broadcast,
              CISCO, status defined, active
Serial0/0.502 (up): point-to-point dlci, dlci 502(0x1F6,0x7C60),
broadcast
          status defined, active
```

Copyright © 2010 Internetwork Expert

Once the mappings appear testing unicast and broadcast reachability can eliminate and higher layer problems, such as incorrect IP address assignments.

```
Rack1R3#ping 155.1.100.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.100.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/59/60 ms

Rack1R3#ping 155.1.100.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.100.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/59/60 ms

Rack1R3#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 155.1.100.5, 60 ms
Reply to request 0 from 155.1.100.4, 72 ms
```

Note

Erase and reload all devices to a blank configuration before continuing.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

2.5 Multipoint Subinterfaces & Static Mappings

- Modify the previous configuration between R3, R4, and R5 to use static IP to DLCI mappings as opposed to dynamic mappings.
- Ensure that these devices have unicast, multicast, and broadcast reachability to each other on this segment.

Configuration

```
R3:
interface Serial1/0
 encapsulation frame-relay
!
interface Serial1/0.100 multipoint
 ip address 155.1.100.3 255.255.255.0
 frame-relay map ip 155.1.100.4 304 broadcast
 frame-relay map ip 155.1.100.5 305 broadcast

R4:
interface Serial0/0
 encapsulation frame-relay
!
interface Serial0/0.100 multipoint
 ip address 155.1.100.4 255.255.255.0
 frame-relay map ip 155.1.100.3 403 broadcast
 frame-relay map ip 155.1.100.5 405 broadcast

R5:
interface Serial0/0
 encapsulation frame-relay
!
interface Serial0/0.100 multipoint
 ip address 155.1.100.5 255.255.255.0
 frame-relay map ip 155.1.100.3 503 broadcast
 frame-relay map ip 155.1.100.4 504 broadcast
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Identical in operation to the previous task, using static **`frame-relay map`** commands on the multipoint subinterface eliminates any possibility of errors in dynamic negotiation. Ensure to add the *broadcast* keyword at the end of the mapping to ensure that broadcast and multicast packets can be encapsulated.

```
Rack1R3#show frame-relay map
Serial1/0.100 (up): ip 155.1.100.4 dlci 304(0x130,0x4C00), static,
              broadcast,
              CISCO, status defined, active
Serial1/0.100 (up): ip 155.1.100.5 dlci 305(0x131,0x4C10), static,
              broadcast,
              CISCO, status defined, active

Rack1R4#show frame-relay map
Serial0/0.100 (up): ip 155.1.100.3 dlci 403(0x193,0x6430), static,
              broadcast,
              CISCO, status defined, active
Serial0/0.100 (up): ip 155.1.100.5 dlci 405(0x195,0x6450), static,
              broadcast,
              CISCO, status defined, active

Rack1R5#show frame-relay map
Serial0/0.100 (up): ip 155.1.100.3 dlci 503(0x1F7,0x7C70), static,
              broadcast,
              CISCO, status defined, active
Serial0/0.100 (up): ip 155.1.100.4 dlci 504(0x1F8,0x7C80), static,
              broadcast,
              CISCO, status defined, active
Serial0/0 (up): ip 155.1.0.1 dlci 501(0x1F5,0x7C50), static,
              broadcast,
              CISCO, status defined, active
Serial0/0.502 (up): point-to-point dlci, dlci 502(0x1F6,0x7C60),
broadcast
          status defined, active
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#ping 155.1.100.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.100.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 57/59/60 ms

Rack1R3#ping 155.1.100.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.100.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 60/60/60 ms

Rack1R3#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 155.1.100.5, 72 ms
Reply to request 0 from 155.1.100.4, 88 ms
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

2.6 Hub-and-Spoke & Static Mappings

- Enable Frame Relay on the Serial interfaces of R1, R3, and R5 connected to the Frame Relay switch.
- Assign IP addresses to these interfaces per the diagram.
- Configure static IP to DLCI mappings on these devices to gain reachability to each other using only the DLCI assignments shown in the diagram.
- Ensure that R1 & R5 and R3 & R5 have unicast, multicast, and broadcast reachability to each other, and that unicast traffic between R1 & R3 transits through R5.

Configuration

```
R1:
interface Serial0/0
 encapsulation frame-relay
 ip address 155.1.0.1 255.255.255.0
 frame-relay map ip 155.1.0.5 105 broadcast
 frame-relay map ip 155.1.0.3 105

R3:
interface Serial1/0
 encapsulation frame-relay
 ip address 155.1.0.3 255.255.255.0
 frame-relay map ip 155.1.0.5 305 broadcast
 frame-relay map ip 155.1.0.1 305

R5:
interface Serial0/0
 encapsulation frame-relay
 ip address 155.1.0.5 255.255.255.0
 frame-relay map ip 155.1.0.1 501 broadcast
 frame-relay map ip 155.1.0.3 503 broadcast
```

Copyright © 2010 Internetwork Expert

Verification

Note

Hub-and-spoke, or partial mesh networks, implies that the layer 3 network does not map directly to a full mesh of layer 2 circuits. In this particular design R1, R3, and R5 are all in the subnet 155.1.0.0/24, but there is no layer 2 DLCI directly between R1 and R3. This implies that traffic from R1 to R3 must first transit R5. To accomplish this R1 has a static **`frame-relay map`** command for the IP addresses of R3 and R5, both reachable through the DLCI to R5. Likewise R3's mappings to both R1 and R5 use the circuit address to R5.

Note that the *broadcast* keyword on R1 and R3 is only associated with one of the mappings. This is to prevent excess broadcast or multicast traffic from being replicated multiple times at layer 2 and going out the same circuit. For example if the broadcast keyword was listed on all mappings in this design, every broadcast packet R1 sends to the interface to be delivered out DLCI 105 would be copied twice, and R5 would receive two packets. The excess traffic will be dropped on R5, and should not cause any reachability problems, but it will result in excess utilization on the link between the devices.

```
Rack1R1#show frame-relay map
Serial0/0 (up): ip 155.1.0.3 dlci 105(0x69,0x1890), static,
              CISCO, status defined, active
Serial0/0 (up): ip 155.1.0.5 dlci 105(0x69,0x1890), static,
              broadcast,
              CISCO, status defined, active

Rack1R3#show frame-relay map
Serial1/0 (up): ip 155.1.0.1 dlci 305(0x131,0x4C10), static,
              CISCO, status defined, active
Serial1/0 (up): ip 155.1.0.5 dlci 305(0x131,0x4C10), static,
              broadcast,
              CISCO, status defined, active

Rack1R5#show frame-relay map
Serial0/0 (up): ip 155.1.0.1 dlci 501(0x1F5,0x7C50), static,
              broadcast,
              CISCO, status defined, active
Serial0/0 (up): ip 155.1.0.3 dlci 503(0x1F7,0x7C70), static,
              broadcast,
              CISCO, status defined, active
```

Copyright © 2010 Internetwork Expert

At this point R1, R3, and R5 should all have unicast reachability to each other, but R1 and R3 will not have broadcast or multicast reachability. This is due to the fact that R5 cannot route broadcasts or link local multicasts between interfaces, which is the desired behavior.

```
Rack1R1#ping 155.1.0.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/116/124 ms

Rack1R1#ping 155.1.0.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/61 ms

Rack1R1#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 155.1.0.5, 61 ms

Rack1R3#ping 155.1.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 116/117/124 ms

Rack1R3#ping 155.1.0.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 57/59/60 ms

Rack1R3#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 155.1.0.5, 104 ms
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#ping 155.1.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/60 ms
Rack1R5#ping 155.1.0.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/59/61 ms

Rack1R5#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 155.1.0.3, 92 ms
Reply to request 0 from 155.1.0.1, 108 ms
```

Copyright © 2010 Internetwork Expert

2.7 Hub-and-Spoke & Inverse-ARP

- Reset the Frame Relay interfaces of R1, R3, and R5 to its default configuration.
- Enable Frame Relay on the Serial interfaces of R1, R3, and R5 connected to the Frame Relay switch.
- Assign IP addresses to these interfaces per the diagram.
- Configure the network in such a way that R1 & R5 and R3 & R5 resolve each other's IP addresses dynamically through Inverse-ARP, while R1 and R3 resolve each other's addresses statically.
- Ensure that R1 & R5 and R3 & R5 have unicast, multicast, and broadcast reachability to each other, that unicast traffic between R1 & R3 transits through R5.

Configuration

```
R1:
interface Serial0/0
 encapsulation frame-relay
 ip address 155.1.0.1 255.255.255.0
 frame-relay map ip 155.1.0.3 105

R3:
interface Serial1/0
 encapsulation frame-relay
 ip address 155.1.0.3 255.255.255.0
 frame-relay map ip 155.1.0.1 305

R5:
interface Serial0/0
 encapsulation frame-relay
 ip address 155.1.0.5 255.255.255.0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Frame Relay Inverse-ARP packets are link-local to the layer 2 process, and cannot be forwarded between different circuits. This means that when R1 sends an Inverse-ARP request out to R5 on DLCI 105, R5 cannot turn around and forward this request to R3 on DLCI 503. The result of this is that R5 can resolve R1 and R3 via Inverse-ARP, R1 and R3 can resolve R5 via Inverse-ARP, but R1 and R3 cannot resolve each other through Inverse-ARP. To gain reachability between these devices a static **`frame-relay map`** command must be used to populate the layer 3 to layer 2 resolution table.

```
Rack1R1#show frame-relay map
Serial0/0 (up): ip 155.1.0.3 dlci 105(0x69,0x1890), static,
              CISCO, status defined, active
Serial0/0 (up): ip 155.1.0.5 dlci 105(0x69,0x1890), dynamic,
              broadcast,, status defined, active

Rack1R3#show frame-relay map
Serial1/0 (up): ip 155.1.0.1 dlci 305(0x131,0x4C10), static,
              CISCO, status defined, active
Serial1/0 (up): ip 155.1.0.5 dlci 305(0x131,0x4C10), dynamic,
              broadcast,, status defined, active

Rack1R5#show frame-relay map
Serial0/0 (up): ip 155.1.0.1 dlci 501(0x1F5,0x7C50), dynamic,
              broadcast,, status defined, active
Serial0/0 (up): ip 155.1.0.3 dlci 503(0x1F7,0x7C70), dynamic,
              broadcast,, status defined, active

Rack1R1#ping 155.1.0.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 113/117/124 ms

Rack1R1#ping 155.1.0.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/61 ms

Rack1R1#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2 seconds:

Reply to request 0 from 155.1.0.5, 100 ms
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#ping 155.1.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 116/117/125 ms

Rack1R3#ping 155.1.0.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms

Rack1R3#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2 seconds:

Reply to request 0 from 155.1.0.5, 117 ms

Rack1R5#ping 155.1.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/61 ms

Rack1R5#ping 155.1.0.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 60/60/61 ms

Rack1R5#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2 seconds:

Reply to request 0 from 155.1.0.3, 120 ms
Reply to request 0 from 155.1.0.1, 136 ms
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

2.8 Hub-and-Spoke & Point-to-Point Subinterfaces

- Reset the Frame Relay interfaces of R1, R3, and R5 to its default configuration.
- Enable Frame Relay on the Serial interfaces of R1, R3, and R5 connected to the Frame Relay switch.
- Configure a point-to-point subinterface numbered 105 on R1 and assign it the IP address 155.X.0.1/24 & DLCI 105.
- Configure a point-to-point subinterface numbered 305 on R3 and assign it the IP address 155.X.0.3/24 & DLCI 305.
- Assign the IP address 155.X.0.5 to R5's main Serial interface and configure static IP to DLCI mappings for R1 and R3.
- Ensure that R1 & R5 and R3 & R5 have unicast, multicast, and broadcast reachability to each other, that unicast traffic between R1 & R3 transits through R5.

Configuration

```
R1:
interface Serial0/0
 encapsulation frame-relay
!
interface Serial0/0.105 point-to-point
 ip address 155.1.0.1 255.255.255.0
 frame-relay interface-dlci 105

R3:
interface Serial1/0
 encapsulation frame-relay
!
interface Serial1/0.305 point-to-point
 ip address 155.1.0.3 255.255.255.0
 frame-relay interface-dlci 305

R5:
interface Serial0/0
 encapsulation frame-relay
 ip address 155.1.0.5 255.255.255.0
 frame-relay map ip 155.1.0.1 501 broadcast
 frame-relay map ip 155.1.0.3 503 broadcast
```

Copyright © 2010 Internetwork Expert

Verification

Note

Point-to-point interfaces simplify the previous design by eliminating the need for R1 and R3 to maintain the layer 3 to layer 2 mapping table. Since both R1 and R3 only have one possible layer 2 address to use in the network, 105 on R1 and 305 on R3, point-to-point subinterfaces can be used. However, since R5 has two layer 2 addresses, 501 to R1 and 503 to R3, it must use a multipoint interface and maintain a layer 3 to layer 2 mapping table to differentiate traffic going to R1 vs. R3.

Note that the interface type and resolution method are only locally significant on the routers. This means that R5 does not know that R1 and R3 are using point-to-point subinterfaces from a layer 2 perspective. However other higher layer protocol designs, such as running OSPF over this network, can be affected by the underlying layer 2 network. Strictly from a layer 2 point of view though this scenario has the same exact reachability results as the last two scenarios.

```
Rack1R1#show frame-relay map
Serial0/0.105 (up): point-to-point dlci, dlci 105(0x69,0x1890), broadcast
          status defined, active

Rack1R3#show frame-relay map
Serial1/0.305 (up): point-to-point dlci, dlci 305(0x131,0x4C10), broadcast
          status defined, active

Rack1R5#show frame-relay map
Serial0/0 (up): ip 155.1.0.1 dlci 501(0x1F5,0x7C50), static,
              broadcast,
              CISCO, status defined, active
Serial0/0 (up): ip 155.1.0.3 dlci 503(0x1F7,0x7C70), static,
              broadcast,
              CISCO, status defined, active

Rack1R1#ping 155.1.0.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/116/125 ms

Rack1R1#ping 155.1.0.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/60 ms
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2 seconds:

Reply to request 0 from 155.1.0.5, 76 ms
```

```
Rack1R3#ping 155.1.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/116/124 ms
```

```
Rack1R3#ping 155.1.0.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/59/64 ms
```

```
Rack1R3#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2 seconds:

Reply to request 0 from 155.1.0.5, 60 ms
```

```
Rack1R5#ping 155.1.0.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/60 ms
```

```
Rack1R5#ping 155.1.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms
```

```
Rack1R5#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2 seconds:

Reply to request 0 from 155.1.0.3, 64 ms
Reply to request 0 from 155.1.0.1, 80 ms
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

> **Note**
>
> Erase and reload all devices to a blank configuration before continuing.

2.9 Disabling Inverse-ARP

- Enable Frame Relay on R1, R2, R3, R4, and R5's interfaces connected to the Frame Relay cloud.
- Configure R1's interface with the IP address 155.X.100.1/24.
- Configure R2's interface with the IP address 155.X.100.2/24.
- Configure R3's interface with the IP address 155.X.200.3/24.
- Configure R4's interface with the IP address 155.X.200.4/24.
- Create a multipoint subinterface numbered 100 on R5 with the IP address 155.X.100.5/24 and the DLCIs 501 & 502.
- Create a multipoint subinterface numbered 200 on R5 with the IP address 155.X.200.5/24 and the DLCIs 503 & 504.
- Configure R1 and R2 so that they will not send Inverse-ARP requests for IP out any DLCIs learned from the Frame Relay cloud other than those in the diagram.
- Configure R3 and R4 to disable Inverse-ARP on DLCIs learned from the Frame Relay cloud that are not listed in the diagram by assigning them to the subinterface 999.
- Ensure that R5 has unicast, multicast, and broadcast reachability to R1, R2, R3, and R4.
- R1 and R2 should be able to reach R5, but not each other.
- R3 and R4 should be able to reach R5, but not each other.
- R1, R2, R3, and R4 should not be able to reach each other.

Copyright © 2010 Internetwork Expert

Configuration

```
R1:
interface Serial0/0
 encapsulation frame-relay
 ip address 155.1.100.1 255.255.255.0
 no frame-relay inverse-arp ip 102
 no frame-relay inverse-arp ip 103
 no frame-relay inverse-arp ip 104
 no frame-relay inverse-arp ip 113

R2:
interface Serial0/0
 encapsulation frame-relay
 ip address 155.1.100.2 255.255.255.0
 no frame-relay inverse-arp ip 201
 no frame-relay inverse-arp ip 203
 no frame-relay inverse-arp ip 204
 no frame-relay inverse-arp ip 213

R3:
interface Serial1/0
 encapsulation frame-relay
 ip address 155.1.200.3 255.255.255.0
!
interface Serial1/0.999 multipoint
 frame-relay interface-dlci 301
 frame-relay interface-dlci 302
 frame-relay interface-dlci 304

R4:
interface Serial0/0
 encapsulation frame-relay
 ip address 155.1.200.3 255.255.255.0
!
interface Serial0/0.999 multipoint
 frame-relay interface-dlci 401
 frame-relay interface-dlci 402
 frame-relay interface-dlci 403
 frame-relay interface-dlci 413

R5:
interface Serial0/0
 encapsulation frame-relay
!
interface Serial0/0.100 multipoint
 ip address 155.1.100.5 255.255.255.0
 frame-relay interface-dlci 501
 frame-relay interface-dlci 502
!
interface Serial0/0.200 multipoint
 ip address 155.1.200.5 255.255.255.0
 frame-relay interface-dlci 503
 frame-relay interface-dlci 504
```

Copyright © 2010 Internetwork Expert

Verification

Note

When a layer 3 protocol such as IPv4 is configured on a multipoint Frame Relay interface, Inverse-ARP requests are automatically be sent out all PVCs assigned to that interface that are learned from LMI. In this case there is a full mesh of layer 2 circuits in the underlying Frame Relay cloud, however only a portion of these DLCIs are used in the network topology. If Inverse-ARP were not disabled, i.e. just issuing the **`encapsulation frame-relay`** and the **`ip address`** commands, R1, R2, R3, and R4 would have formed a full mesh of Frame Relay mappings, even though they are not all in the same IP subnet. From a layer 2 perspective this may not be an issue, but when higher layer protocols such as EIGRP or Multicast are enabled problems will arise.

In this design R1 and R2 prevent the problem by disabling the Inverse-ARP request on the circuits with the no frame-relay inverse-arp command. On R3 and R4 this problem is prevented by moving the DLCIs learned via LMI to a multipoint subinterface that does not have an IP address assigned. Since the subinterface is not running IPv4, an Inverse-ARP request cannot be sent out the PVCs assigned to that interface.

```
Rack1R1#show frame-relay map
Serial0/0 (up): ip 155.1.100.5 dlci 105(0x69,0x1890), dynamic,
              broadcast,, status defined, active

Rack1R2#show frame-relay map
Serial0/0 (up): ip 155.1.100.5 dlci 205(0xCD,0x30D0), dynamic,
              broadcast,, status defined, active

Rack1R3#show frame-relay map
Serial1/0 (up): ip 155.1.200.5 dlci 305(0x131,0x4C10), dynamic,
              broadcast,, status defined, active

Rack1R4#show frame-relay map
Serial0/0 (up): ip 155.1.200.5 dlci 405(0x195,0x6450), dynamic,
              broadcast,
              CISCO, status defined, active
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show frame-relay map
Serial0/0.100 (up): ip 155.1.100.1 dlci 501(0x1F5,0x7C50), dynamic,
              broadcast,, status defined, active
Serial0/0.100 (up): ip 155.1.100.2 dlci 502(0x1F6,0x7C60), dynamic,
              broadcast,, status defined, active
Serial0/0.200 (up): ip 155.1.200.3 dlci 503(0x1F7,0x7C70), dynamic,
              broadcast,
              CISCO, status defined, active
Serial0/0.200 (up): ip 155.1.200.4 dlci 504(0x1F8,0x7C80), dynamic,
              broadcast,
              CISCO, status defined, active

Rack1R5#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 155.1.100.2, 96 ms
Reply to request 0 from 155.1.200.3, 196 ms
Reply to request 0 from 155.1.200.4, 124 ms
Reply to request 0 from 155.1.100.1, 112 ms
```

Copyright © 2010 Internetwork Expert

2.10 Back-to-Back Frame Relay

- Configure IP addresses on the directly connected Serial link between R4 and R5 using the information in the diagram.
- Configure Frame Relay PVC 514 on this link without using LMI.
- Configure static IP to DLCI mappings on R4 and R5 to obtain reachability on this segment.

Configuration

```
R4:
interface Serial0/1
 ip address 155.1.45.4 255.255.255.0
 encapsulation frame-relay
 no keepalive
 frame-relay map ip 155.1.45.5 514 broadcast

R5:
interface Serial0/1
 ip address 155.1.45.5 255.255.255.0
 encapsulation frame-relay
 no keepalive
 clock rate 64000
 frame-relay map ip 155.1.45.4 514 broadcast
```

Verification

Note

Back-to-Back Frame Relay can be used between routers connected directly by a Serial link. Since there is no Frame Relay service provider that is advertising PVCs though LMI, the **no keepalive** command is used to disable LMI. The result of this is seen in the **show frame-relay pvc** output with the PVC status of *STATIC*. STATIC means that the router is not receiving LMI updates about the circuit, and it can never move to the INACTIVE state due to an LMI message.

```
Rack1R4#show frame-relay pvc 514

PVC Statistics for interface Serial0/1 (Frame Relay DTE)

DLCI = 514, DLCI USAGE = LOCAL, PVC STATUS = STATIC, INTERFACE = Serial0/1

  input pkts 66             output pkts 66           in bytes 6864
  out bytes 6864            dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                  out bytes dropped 0
  in FECN pkts 0            in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0           in DE pkts 0             out DE pkts 0
  out bcast pkts 0          out bcast bytes 0
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 00:01:52, last time pvc status changed 00:01:52
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show frame-relay map
Serial0/0 (up): ip 155.1.200.5 dlci 405(0x195,0x6450), dynamic,
              broadcast,
              CISCO, status defined, active
Serial0/1 (up): ip 155.1.45.5 dlci 514(0x202,0x8020), static,
              broadcast,
              CISCO

Rack1R4#ping 155.1.45.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.45.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/32 ms

Rack1R4#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2 seconds:

Reply to request 0 from 155.1.45.5, 32 ms
Reply to request 0 from 155.1.200.5, 60 ms

Rack1R5#show frame-relay pvc 514

PVC Statistics for interface Serial0/1 (Frame Relay DTE)

DLCI = 514, DLCI USAGE = LOCAL, PVC STATUS = STATIC, INTERFACE = Serial0/1

  input pkts 73             output pkts 101          in bytes 7592
  out bytes 10504           dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                  out bytes dropped 0
  in FECN pkts 0            in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0           in DE pkts 0             out DE pkts 0
  out bcast pkts 0          out bcast bytes 0
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 00:05:40, last time pvc status changed 00:05:30

Rack1R5#show frame-relay map
Serial0/0.100 (up): ip 155.1.100.1 dlci 501(0x1F5,0x7C50), dynamic,
              broadcast,, status defined, active
Serial0/0.100 (up): ip 155.1.100.2 dlci 502(0x1F6,0x7C60), dynamic,
              broadcast,, status defined, active
Serial0/0.200 (up): ip 155.1.200.3 dlci 503(0x1F7,0x7C70), dynamic,
              broadcast,
              CISCO, status defined, active
Serial0/0.200 (up): ip 155.1.200.4 dlci 504(0x1F8,0x7C80), dynamic,
              broadcast,
              CISCO, status defined, active
Serial0/1 (up): ip 155.1.45.4 dlci 514(0x202,0x8020), static,
              broadcast,
              CISCO
```

2.11 Frame Relay End-to-End Keepalives

- Create a Frame Relay map-class called FREEK on R4 and R5.
- Configure R4 to respond to Frame Relay keepalives, and R5 to request them.
- Set R5 to poll every 5 seconds with an event window of 10 and an error threshold of 8.
- R4 should apply this configuration to its main Serial0/0 interface, while R5 should apply this just to the DLCI connecting to R4.

Configuration

```
R4:
map-class frame-relay FREEK
 frame-relay end-to-end keepalive mode reply
 frame-relay end-to-end keepalive timer recv 5
 frame-relay end-to-end keepalive event-window recv 10
 frame-relay end-to-end keepalive error-threshold recv 8
!
interface Serial0/0
 frame-relay class FREEK

R5:
map-class frame-relay FREEK
 frame-relay end-to-end keepalive mode request
 frame-relay end-to-end keepalive timer send 5
 frame-relay end-to-end keepalive event-window send 10
 frame-relay end-to-end keepalive error-threshold send 8
!
interface Serial0/0.200 multipoint
 frame-relay interface-dlci 504
  class FREEK
```

Copyright © 2010 Internetwork Expert

Verification

Note

In true end-to-end Frame Relay networks the LMI status that one router receives indicates the end-to-end status of the circuit that the other router receives. This consists of the status from the router to the switch via the User-to-Network interface (UNI), the status of all the switch to switch links via the Network-to-Network (NNI) interface, and the remote UNI. However in the majority of real world service provider networks today Frame Relay only exists on the edge, and is transported in a carrier protocol such as MPLS in the transit. The result of this is that the LMI status of one end of the link does not update the LMI status of the remote end.

The Frame Relay End-to-End Keepalive (EEK) feature avoids design problems like this by adding an additional keepalive layer that is exchanged directly between the router endpoints. When a router is configured to "send" EEK, it implies that it is waiting to "receive" EEK. Likewise if a router is configured to "receive" EEK it is expecting the "send" to come in. This means that if the receive side does not hear the request come in after the event timers expire, or the send side does not hear the reply come in, the PVC status changes to EEK DOWN, and traffic cannot be sent out the link.

```
Rack1R4#show frame-relay end-to-end keepalive interface seria0/0 405

End-to-end Keepalive Statistics for interface Serial0/0 (Frame Relay DTE)

DLCI = 405, DLCI USAGE = LOCAL, VC STATUS = ACTIVE (EEK UP)

RECEIVE SIDE STATISTICS

Send Sequence Number: 18,       Receive Sequence Number: 17
Configured Event Window: 10,    Configured Error Threshold: 8
Total Observed Events: 30,      Total Observed Errors: 10
Monitored Events: 10,           Monitored Errors: 0
Successive Successes: 10,       End-to-end VC Status: UP

Rack1R5#show frame-relay end-to-end keepalive interface serial0/0

End-to-end Keepalive Statistics for Interface Serial0/0 (Frame Relay DTE)

DLCI = 504, DLCI USAGE = LOCAL, VC STATUS = ACTIVE (EEK UP)

SEND SIDE STATISTICS

Send Sequence Number: 22,       Receive Sequence Number: 23
Configured Event Window: 10,    Configured Error Threshold: 8
Total Observed Events: 25,      Total Observed Errors: 0
Monitored Events: 10,           Monitored Errors: 0
Successive Successes: 10,       End-to-end VC Status: UP
```

Copyright © 2010 Internetwork Expert

2.12 Frame Relay Broadcast Queue

- Tune the Frame Relay broadcast queue size between R4 and R5 to 100 packets.

Configuration

```
R4:
interface Serial0/0
 frame-relay broadcast-queue 100 256000 36

R5:
interface Serial0/0
 frame-relay broadcast-queue 100 256000 36
```

2.13 Frame Relay TCP & RTP Header Compression

- Enable RTP header compression for DLCIs 503 and 305 on R5 and R3 respectively.
- Enable TCP header compression for DLCIs 501 and 105 on R5 and R1 respectively.

Configuration

```
R1:
interface Serial0/0
 frame-relay map ip 155.1.100.5 105 broadcast tcp header-compression

R3:
interface Serial1/0
 frame-relay map ip 155.1.200.5 305 broadcast rtp header-compression

R5:
interface Serial0/0.100 multipoint
 frame-relay map ip 155.1.100.1 501 broadcast tcp header-compression
!
interface Serial0/0.200 multipoint
 frame-relay map ip 155.1.200.3 503 broadcast rtp header-compression
```

Copyright © 2010 Internetwork Expert

Verification

```
Rack1R5#show frame-relay ip tcp header-compression
 DLCI 501        Link/Destination info: ip 155.1.100.1
  Interface Serial0/0.100 DLCI 501 (compression on, VJ)
    Rcvd:    67 total, 65 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    46 total, 43 compressed, 0 status msgs, 0 not predicted
             1473 bytes saved, 435 bytes sent
             4.38 efficiency improvement factor
    Connect: 256 rx slots, 256 tx slots,
             2 misses, 0 collisions, 0 negative cache hits, 255 free contexts
             95% hit ratio, five minute miss rate 0 misses/sec, 0 max

Rack1R5#show frame-relay ip rtp header-compression
 DLCI 503        Link/Destination info: ip 155.1.200.3
  Interface Serial0/0.200 DLCI 503 (compression on, Cisco, RTP)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 256 rx slots, 256 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 256 free contexts
```

Note

Erase and reload all devices to a blank configuration before continuing.

Copyright © 2010 Internetwork Expert

2.14 PPP over Frame Relay

- Create Virtual-Template interfaces on R1 and R5 using the IP addresses 155.X.0.1/24 and 155.X.0.5/24 respectively.
- Enable Frame Relay on R1 and R5's Serial interfaces attached to the Frame Relay cloud, and attach the Virtual-Templates to the DLCIs connecting these devices.
- Ensure that R1 and R5 have unicast, multicast, and broadcast reachability to each other over this segment.

Configuration

```
R1:
interface Serial0/0
 encapsulation frame-relay
 frame-relay interface-dlci 105 ppp Virtual-Template1
!
interface Virtual-Template1
 ip address 155.1.0.1 255.255.255.0

R5:
interface Serial0/0
 encapsulation frame-relay
 frame-relay interface-dlci 501 ppp Virtual-Template1
!
interface Virtual-Template1
 ip address 155.1.0.5 255.255.255.0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Running PPP over Frame Relay (PPPoFR) adds additional features such as authentication that are not available in the Frame Relay header itself. Most issues that relate to PPPoFR come from the order of operations. The first step in this configuration should be to configure the Virtual-Template interface. On this interface all logical options such as PPP authentication or IP addressing should be configured. Next the Virtual-Template should be bound to the Frame Relay DLCI with the **`frame-relay interface-dlci xxx ppp Virtual-Template`** command. If bound properly a log message should appear that the *Virtual-Access* interface's line protocol changed state to up. This means that the logical instance of the PPP link is actually active. When viewing the **`show ip interface brief`** or **`show ip route`** output, this is the interface that should be in the UP/UP state.

```
Rack1R1#show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
FastEthernet0/0        unassigned      YES manual administratively down down
Serial0/0              unassigned      YES TFTP   up                    up
Serial0/0.105          unassigned      YES manual deleted               down
Serial0/1              unassigned      YES manual administratively down down
NVI0                   unassigned      NO  unset  up                    up
Virtual-Access1        unassigned      YES unset  down                  down
Virtual-Template1      155.1.0.1       YES manual down                  down
Virtual-Access2        155.1.0.1       YES TFTP   up                    up
```

```
Rack1R1#show interface virtual-access2
Virtual-Access2 is up, line protocol is up
  Hardware is Virtual Access interface
  Internet address is 155.1.0.1/24
  MTU 1500 bytes, BW 100000 Kbit, DLY 100000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation PPP, LCP Open
  Open: IPCP
  PPPoFR vaccess, cloned from Virtual-Template1
  Vaccess status 0x44
  Bound to Serial0/0 DLCI 105, Cloned from Virtual-Template1, loopback not set
  Keepalive set (10 sec)
  DTR is pulsed for 5 seconds on reset
  Last input 00:02:00, output never, output hang never
  Last clearing of "show interface" counters 00:02:17
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
<output omitted>
```

Copyright © 2010 Internetwork Expert

By default PPP uses the peer-neighbor route in order to solve any routing problems due to unnumbered links or different IP address pools in DHCP. If the PPP link is up and functional the peer-neighbor route, which is the remote end's IP address, should appear in the routing table as a connected route.

```
Rack1R1#show ip route connected
     155.1.0.0/16 is variably subnetted, 2 subnets, 2 masks
C       155.1.0.0/24 is directly connected, Virtual-Access2
C       155.1.0.5/32 is directly connected, Virtual-Access2
```

Since PPP is encapsulated between the Frame Relay and the normal layer 3 header, the Frame Relay process does not require layer 3 to layer 2 resolution for any upper layer protocols that are configured on top of PPP. This means that both unicast and multicast/broadcast transmission are successful for IPv4 without the need for a **`frame-relay map`** command, even though this configuration uses the main Serial interfaces, which are usually multipoint. This is due to the fact that the Virtual-Access is a point-to-point link, since by definition PPP is a point-to-point protocol.

```
Rack1R1#ping 155.1.0.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/61 ms

Rack1R1#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2 seconds:

Reply to request 0 from 155.1.0.5, 64 ms
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

2.15 Bridging over Frame Relay

- Remove the previous PPPoFR configuration on R5.
- Configure ports Fa0/2 and Fa0/24 on SW2 in VLAN 100.
- Configure SW1's interface Fa0/5 with the IP address 192.10.X.7/24.
- Enable Frame Relay on the Serial interfaces connecting to the Frame Relay cloud on R2 and R5.
- Disable IP routing on R2 and R5 and remove any IP addresses from their interfaces.
- Configure an IEEE STP bridge group numbered 1 on R2 & R5, and apply this to their Frame Relay connections and their FastEthernet0/0 interfaces.
- Ensure that SW1 and BB2 have unicast, multicast, and broadcast reachability to each other.

Configuration

```
R2:
interface FastEthernet0/0
 bridge-group 1
!
interface Serial0/0
 encapsulation frame-relay
 frame-relay map bridge 205 broadcast
 bridge-group 1
!
bridge 1 protocol ieee

R5:
interface FastEthernet0/0
 bridge-group 1
!
interface Serial0/0
 encapsulation frame-relay
 frame-relay map bridge 502 broadcast
 bridge-group 1
!
bridge 1 protocol ieee

SW1:
interface FastEthernet0/5
 no switchport
 ip address 192.10.1.7 255.255.255.0

SW2:
interface FastEthernet0/2
 switchport access vlan 100
!
interface FastEthernet0/24
 switchport access vlan 100
```

Copyright © 2010 Internetwork Expert

Verification

Note

When bridging over multipoint Frame Relay interfaces the **frame-relay map bridge** command is required to tell the router which PVC to encapsulate Spanning-Tree Protocol and the other bridged packets on the interface.

Once completed the below three verifications all imply that the bridge domain is built properly. From the **show spanning-tree** output on R2 and R5 we can see that they both agree on who the root bridge is. This implies that they see each other's STP BPDUs, and are in the same broadcast domain.

```
Rack1R2#show spanning-tree 1

 Bridge group 1 is executing the ieee compatible Spanning Tree protocol
  Bridge Identifier has priority 32768, address 0012.0120.3680
  Configured hello time 2, max age 20, forward delay 15
  Current root has priority 32768, address 0012.001c.daa0
  Root port is 5 (Serial0/0), cost of root path is 647
  Topology change flag not set, detected flag not set
  Number of topology changes 7 last change occurred 00:01:15 ago
          from Serial0/0
  Times:  hold 1, topology change 35, notification 2
          hello 2, max age 20, forward delay 15
  Timers: hello 0, topology change 0, notification 0, aging 300

 Port 4 (FastEthernet0/0) of Bridge group 1 is forwarding
    Port path cost 19, Port priority 128, Port Identifier 128.4.
    Designated root has priority 32768, address 0012.001c.daa0
    Designated bridge has priority 32768, address 0012.0120.3680
    Designated port id is 128.4, designated path cost 647
    Timers: message age 0, forward delay 0, hold 0
    Number of transitions to forwarding state: 1
    BPDU: sent 357, received 5

 Port 5 (Serial0/0) of Bridge group 1 is forwarding
    Port path cost 647, Port priority 128, Port Identifier 128.5.
    Designated root has priority 32768, address 0012.001c.daa0
    Designated bridge has priority 32768, address 0012.001c.daa0
    Designated port id is 128.6, designated path cost 0
    Timers: message age 1, forward delay 0, hold 0
    Number of transitions to forwarding state: 1
    BPDU: sent 2, received 55
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show spanning-tree 1

 Bridge group 1 is executing the ieee compatible Spanning Tree protocol
  Bridge Identifier has priority 32768, address 0012.001c.daa0
  Configured hello time 2, max age 20, forward delay 15
  We are the root of the spanning tree
  Topology change flag not set, detected flag not set
  Number of topology changes 4 last change occurred 00:01:58 ago
          from Serial0/0
  Times:  hold 1, topology change 35, notification 2
          hello 2, max age 20, forward delay 15
  Timers: hello 0, topology change 0, notification 0, aging 300

 Port 4 (FastEthernet0/0) of Bridge group 1 is forwarding
    Port path cost 19, Port priority 128, Port Identifier 128.4.
    Designated root has priority 32768, address 0012.001c.daa0
    Designated bridge has priority 32768, address 0012.001c.daa0
    Designated port id is 128.4, designated path cost 0
    Timers: message age 0, forward delay 0, hold 0
    Number of transitions to forwarding state: 1
    BPDU: sent 364, received 3

 Port 6 (Serial0/0) of Bridge group 1 is forwarding
    Port path cost 647, Port priority 128, Port Identifier 128.6.
    Designated root has priority 32768, address 0012.001c.daa0
    Designated bridge has priority 32768, address 0012.001c.daa0
    Designated port id is 128.6, designated path cost 0
    Timers: message age 0, forward delay 0, hold 0
    Number of transitions to forwarding state: 1
    BPDU: sent 368, received 8
```

A traceroute on SW1 shows only one hop to the destination. This implies that there are no layer 3 routing devices in the transit path, only layer 2 bridges.

```
Rack1SW1#traceroute 192.10.1.254

Type escape sequence to abort.
Tracing the route to 192.10.1.254

  1 192.10.1.254 42 msec *  42 msec
```

By pinging the all subnet broadcast address all devices that are directly connected in the broadcast domain should respond. Since SW1 gets a response from BB2, via 192.10.1.254, this implies that SW1's interface 192.10.1.7 is in the same broadcast domain as BB2.

```
Rack1SW1#ping 255.255.255.255 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 255.255.255.255, timeout is 2
seconds:

Reply to request 0 from 192.10.1.254, 67 ms
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

2.16 Frame-Relay Switching

- Configure R3 to provision a switched Frame-Relay link between R1 and R2.
- Use the Serial interfaces connecting R1 and R2 to R3 to accomplish this.
- Use the DLCI numbers 132 and 231 on the links connecting R1 to R3 and R2 to R3 respectively.
- Do not use the **`frame-relay route`** command to accomplish this.
- Do not use Inverse-ARP on R2 to establish IP connectivity.
- Use the IP subnet 155.X.12.0/24 for IP addressing.

Configuration

Note

Frame-Relay switching topic is a new (Oct 18 2009) addition to CCIE R&S exam. The Frame-Relay switch is no longer a backbone device but a part of the student's pod topology. However, as far as the content of the R&S track goes, configuring FR switching is relatively straight-forward. Switching VCs are not supported by IOS and this fact only leaves room for static PVCs configuration.

In short, the router acting as a FR switch should be enabled for Frame-Relay switching using the global command **`frame-relay switching`**. Next, the interfaces facing the "client" devices should be configured as Frame-Relay DCE interfaces using the command **`frame-relay intf-type dce`** (the default type is FR DTE). Notice that in case of back-to-back serial connections you may have to be configured the physical DCE side of the cable for clocking using the command clock rate. It might be that the FR DCE and physical DCE do not match – this is normal as those are two completely different concepts.

Configuring a Frame-Relay interface as FR DCE allows this interface to respond to LMI queries and publish the list of locally active DLCIs. Notice that you may have to adjust the LMI type according to the connected device, even though IOS is supposed to auto-sense LMI type. Remember that FR DLCIs might be either switched or terminated locally at the DCE interface. If the DLCI terminates locally on the DCE interface, you configure it just as you would do with the regular DTE interface – e.g. create a subinterface and use the command **`frame-relay interface-dlci`**. If the DLCI is to be switched to another Frame-Relay port in the switch, you may need to configure either of the two commands:

Copyright © 2010 Internetwork Expert

1) Interface-level **frame-relay route** command that specifies the local DLCI, the other end of the connection – another local interface and the translated DLCI. This is a legacy command that still perfectly works in modern IOS releases. Notice that it is enough to configure this command only for one interface – the FR switch will automatically create a "reverse" entry to switch the FR packets coming to the other end of the connection.

2) Global command "**connect**" that creates a named connection, linking two local interfaces and two DLCIs. This is a unified syntax that is used for many other types of L2 connections. By configuring this command you will enter the connection configuration mode where you can apply switched features, e.g. Frame-Relay traffic-policing. As with the **frame-relay route** command it is enough to create just a single instance of this command – the router will automatically creates the "reverse" connection.

Both commands will populate the Frame-Relay switching table with the switching entries. Remember that FR DLCIs have local semantics and may change from switch to switch. After you have the connection entries created you may use the **show frame-relay pvc, show frame-relay route** and **show connection** commands to check your configuration.

R3:

```
frame-relay switching
!
interface Serial 1/2
 clock rate 64000
 encapsulation frame-relay
 frame-relay intf-type dce
 no shutdown
!
interface Serial 1/3
 clock rate 64000
 encapsulation frame-relay
 frame-relay intf-type dce
 no shutdown
!
connect R1_R2 Serial 1/2 132 Serial 1/3 231
```

R1:

```
interface Serial 0/1
 encapsulation frame-relay
 ip address 155.1.12.1 255.255.255.0
 no shutdown
```

Copyright © 2010 Internetwork Expert

```
R2:
interface Serial 0/1
 encapsulation frame-relay
 no frame-relay inverse-arp
 ip address 155.1.12.2 255.255.255.0
 frame-relay map ip 155.1.12.1 231
 no shutdown
```

Verification

> **Note**
>
> Use the following commands to identify the switched DLCIs and list their statistics. Notice that the PVC status must be ACTIVE, otherwise there is a problem – either misconfiguration or physical/logical interface problem (e.g. LMI mismatch and missed LMI keepalives).

```
RSRack1R3#show frame-relay pvc | inc SWI
DLCI = 132, DLCI USAGE = SWITCHED, PVC STATUS = ACTIVE, INTERFACE =
Serial1/2
DLCI = 231, DLCI USAGE = SWITCHED, PVC STATUS = ACTIVE, INTERFACE =
Serial1/3

RSRack1R3#show frame-relay pvc 132

PVC Statistics for interface Serial1/2 (Frame Relay DCE)

DLCI = 132, DLCI USAGE = SWITCHED, PVC STATUS = ACTIVE, INTERFACE =
Serial1/2

  input pkts 10             output pkts 10           in bytes 1040
  out bytes 1040            dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                  out bytes dropped 0
  in FECN pkts 0            in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0           in DE pkts 0             out DE pkts 0
  out bcast pkts 0          out bcast bytes 0
  30 second input rate 0 bits/sec, 0 packets/sec
  30 second output rate 0 bits/sec, 0 packets/sec
  switched pkts 10
  Detailed packet drop counters:
  no out intf 0             out intf down 0          no out PVC 0
  in PVC down 0             out PVC down 0           pkt too big 0
  shaping Q full 0          pkt above DE 0           policing drop 0
  pvc create time 21:52:48, last time pvc status changed 21:52:35
```

Copyright © 2010 Internetwork Expert

RSRack1R3#show frame-relay pvc 231

```
PVC Statistics for interface Serial1/3 (Frame Relay DCE)

DLCI = 231, DLCI USAGE = SWITCHED, PVC STATUS = ACTIVE, INTERFACE =
Serial1/3

  input pkts 10            output pkts 10           in bytes 1040
  out bytes 1040           dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 0         out bcast bytes 0
  30 second input rate 0 bits/sec, 0 packets/sec
  30 second output rate 0 bits/sec, 0 packets/sec
  switched pkts 10
  Detailed packet drop counters:
  no out intf 0            out intf down 0          no out PVC 0
  in PVC down 0            out PVC down 0           pkt too big 0
  shaping Q full 0         pkt above DE 0           policing drop 0
  pvc create time 21:52:52, last time pvc status changed 21:52:33
RSRack1R3#
```

RSRack1R3#show connection all

```
ID   Name                Segment 1            Segment 2           State
=====================================================================
=
1    R1_R2              Se1/2 132            Se1/3 231            UP
```

RSRack1R1#show frame-relay pvc 132

```
PVC Statistics for interface Serial0/1 (Frame Relay DTE)

DLCI = 132, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/1

  input pkts 10            output pkts 10           in bytes 1040
  out bytes 1040           dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 0         out bcast bytes 0
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 21:59:56, last time pvc status changed 21:58:08
```

Verify L3 connectivity:

```
RSRack1R2#sh frame-relay map
<snip>
Serial0/1 (up): ip 155.1.12.1 dlci 231(0x84,0x2040), static,
              CISCO, status defined, active
RSRack1R2#

RSRack1R1#ping 149.1.12.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 149.1.12.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/59/60 ms
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

IP Routing Solutions

3.1 Routing to Multipoint Broadcast Interfaces

- Configure R1 with a static route for R4's Loopback 0 network pointing to the IP address of R4.
- Configure R1 with a static route for R6's Loopback 0 network pointing to R1's Ethernet interface connecting to VLAN 146.
- Ensure that R1 can ping the Loopback 0 interfaces of R4 and R6.
- Verify that R6 is running Proxy ARP by checking the ARP table of R1.
- Disable Proxy ARP on R6's connections to VLAN 146.
- Modify R1's ARP table so that it still has IP reachability to the Loopback 0 interface of R6.

Configuration

```
R1:
ip route 150.1.4.0 255.255.255.0 155.1.146.4
ip route 150.1.6.0 255.255.255.0 FastEthernet0/0
arp 150.1.6.6 0011.93da.bf40 arpa

R6:
interface FastEthernet0/0.146
 no ip proxy-arp
```

Verification

> **Note**
>
> When routing to a next-hop value the router performs layer 2 to layer 3 resolution on the next-hop address. When routing to an interface the router performs layer 2 to layer 3 resolution on the final destination. In this particular case this means that R1 will use the MAC address of 155.1.146.4 to reach 150.1.4.4, but directly ARP for the address 150.1.6.6. Since R6 has Proxy ARP enabled by default on its interface connected to this segment it will respond to R1's ARP request with its own MAC address.

```
Rack1R1#show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  155.1.146.1            -   000e.83f8.0d00  ARPA   FastEthernet0/0
Internet  155.1.146.4            5   0011.20d2.4641  ARPA   FastEthernet0/0
Internet  155.1.146.6            5   0011.93da.bf40  ARPA   FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#ping 150.1.4.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1R1#ping 150.1.6.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.6.6, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 1/2/4 ms

Rack1R1#show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  155.1.146.1             -   000e.83f8.0d00  ARPA   FastEthernet0/0
Internet  155.1.146.4             5   0011.20d2.4641  ARPA   FastEthernet0/0
Internet  155.1.146.6             5   0011.93da.bf40  ARPA   FastEthernet0/0
Internet  150.1.6.6               0   0011.93da.bf40  ARPA   FastEthernet0/0
```

Once Proxy ARP is disabled on R6, R1 cannot resolve the destination 150.1.6.6. This is seen from the *encapsulation failed* message R1 generates in the debug ip packet output. Encapsulation failed means that the router does not know the correct layer 2 address to use when building the layer 2 frame.

```
Rack1R1#clear arp

Rack1R1#debug arp
ARP packet debugging is on

Rack1R1#debug ip packet
IP packet debugging is on

Rack1R1#ping 150.1.6.6 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 150.1.6.6, timeout is 2 seconds:
.
Success rate is 0 percent (0/1)
IP: tableid=0, s=155.1.146.1 (local), d=150.1.6.6 (FastEthernet0/0),
routed via RIB
IP: s=155.1.146.1 (local), d=150.1.6.6 (FastEthernet0/0), len 100,
sending
IP ARP: creating incomplete entry for IP address: 150.1.6.6 interface
FastEthernet0/0
IP ARP: sent req src 155.1.146.1 000e.83f8.0d00,
                 dst 150.1.6.6 0000.0000.0000 FastEthernet0/0
IP: s=155.1.146.1 (local), d=150.1.6.6 (FastEthernet0/0), len 100,
encapsulation failed
```

Copyright © 2010 Internetwork Expert

There are two ways to resolve this problem, either change the IP routing on R1 so that it does not ARP for the final destination, or populate the ARP cache of R1 statically so that it knows which MAC address to use when it sends the packet to 150.1.6.6.

```
Rack1R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#arp 150.1.6.6 0011.93da.bf40 arpa
Rack1R1(config)#end

Rack1R1#show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  155.1.146.1              -   000e.83f8.0d00  ARPA   FastEthernet0/0
Internet  155.1.146.4              4   0011.20d2.4641  ARPA   FastEthernet0/0
Internet  155.1.146.6              4   0011.93da.bf40  ARPA   FastEthernet0/0
Internet  150.1.6.6                -   0011.93da.bf40  ARPA

Rack1R1#ping 150.1.6.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.6.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms
```

The ideal solution for this problem design-wise is to never configure a static route to point out a multipoint interface. Static routes should either point to the next-hop value of the neighbor on the multipoint interface, or point to an interface only if it is point-to-point, such as a GRE tunnel.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

3.2 Routing to NBMA Interfaces

- Configure R1 and R2 with default routes pointing out their point-to-point Frame Relay interfaces connecting to R5.
- Configure R5 with a static route for R1's Loopback 0 network pointing to the IP address of R1.
- Configure R5 with a static route for R2's Loopback 0 network pointing out its main Frame Relay interface connecting to R2.
- Ensure that R1, R2, and R5 can all ping each other's Loopback 0 interfaces.

Configuration

```
R1:
ip route 0.0.0.0 0.0.0.0 Serial0/0.1

R2:
ip route 0.0.0.0 0.0.0.0 Serial0/0.1

R5:
interface Serial0/0/0
 frame-relay map ip 150.1.2.2 502
!
ip route 150.1.1.0 255.255.255.0 155.1.0.1
ip route 150.1.2.0 255.255.255.0 Serial0/0/0
```

Copyright © 2010 Internetwork Expert

Verification

Note

Similar to the previous routing example over Ethernet, when a static route points at an interface, layer 3 to layer 2 resolution is performed for the final destination, but if the route points at the intermediary next-hop, layer 3 to layer 2 resolution is performed for the next-hop. Since point-to-point interfaces do not require layer 3 to layer 2 resolution there is no functional difference between using the interface versus the next-hop value.

In this particular case R1 and R2 point routes directly out their point-to-point Frame Relay interfaces. Since these links do not require address resolution there is no problem with encapsulation. On R5 the route to R1's Loopback uses the next-hop value of R1, while the route to R2's Loopback uses the connected interface. This means that when R5 sends traffic to 150.1.1.1 it will use the DLCI that is associated with 155.1.0.1 (DLCI 501), but when it sends traffic to 150.1.2.2 it will look for the DLCI associated directly with 150.1.2.2. Since there is a Frame Relay mapping for R1's next-hop, but not R2's Loopback, only transport to R1's Loopback is successful.

```
Rack1R1#ping 150.1.5.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.5.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/64/88 ms

Rack1R2#ping 150.1.5.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.5.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/59/60 ms

Rack1R5#ping 150.1.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/60 ms
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#debug ip packet
IP packet debugging is on

Rack1R5#debug frame-relay packet
Frame Relay packet debugging is on

Rack1R5#ping 150.1.2.2 repeat 1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 150.1.2.2, timeout is 2 seconds:

IP: tableid=0, s=155.1.0.5 (local), d=150.1.2.2 (Serial0/0/0), routed
via RIB
IP: s=155.1.0.5 (local), d=150.1.2.2 (Serial0/0/0), len 100, sending
Serial0/0:Encaps failed--no map entry link 7(IP)
IP: s=155.1.0.5 (local), d=150.1.2.2 (Serial0/0/0), len 100,
encapsulation failed.
Success rate is 0 percent (0/1)
```

The output *no map entry link* means that R5 does not know which PVC to use to get to 150.1.2.2 because there is no associated **frame-relay map** statement. Once the Frame Relay mapping table includes this resolution, encapsulation is successful.

```
Rack1R5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#frame-relay map ip 150.1.2.2 502
Rack1R5(config-if)#end

Rack1R5#ping 150.1.2.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/61 ms
```

This is another design example as to why static routes should not point out multipoint interfaces. If R5 were to configure a default route pointing directly to Serial0/0/0 it would mean that there would need to be a matching **frame-relay map** statement for every possible destination. On the other hand if the default route pointed towards R1's next-hop value, R5 would only need a **frame-relay map** statement for R1's next-hop.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

3.3 Longest Match Routing

- Configure R4 and R5 with routes to each other's Loopback0 interfaces via the point-to-point Serial interface between them.
- Configure R4 and R5 to route the rest of the 150.X.0.0/16 network via the Frame Relay connection between them.
- Ensure that traffic between R4 and R5's Loopback0 networks transits the point-to-point link between them, but is rerouted out the Frame Relay connection if the point-to-point link is down.

Configuration

```
R4:
ip route 150.1.5.0 255.255.255.0 Serial0/1/0
ip route 150.1.0.0 255.255.0.0 155.1.0.5

R5:
ip route 150.1.4.0 255.255.255.0 Serial0/1/0
ip route 150.1.0.0 255.255.0.0 155.1.0.4
```

Verification

Note

IPv4 routing logic uses *longest match* routing to determine which entry to use from the routing table for forwarding. This principle can be used both to achieve redundancy and traffic engineering.

In this particular example traffic engineering is accomplished for traffic routed between the Loopback 0 networks of R4 and R5. When R5 does a lookup on the final destination 150.1.4.4, the longest match is 150.1.4.0/24 out Serial0/1/0. While technically there are multiple routes to this destination, 150.1.4.0/24 and 150.1.0.0/16, the /24 route wins. For any other 150.1.0.0 destinations, such as 150.1.3.3, the longest match will be 150.1.0.0 via 155.1.0.4. This allows traffic for a portion of the IP address space to be routed one way, and traffic for a different portion of the IP address space to be routed another way.

```
Rack1R5#show ip route 150.1.4.4
Routing entry for 150.1.4.0/24
  Known via "static", distance 1, metric 0 (connected)
  Routing Descriptor Blocks:
  * directly connected, via Serial0/1/0
      Route metric is 0, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.45.4 16 msec *  12 msec
```

Redundancy is accomplished in this example when the point-to-point link between R4 and R5 fails. As long as this link is up and installed in the routing table, the route 150.1.4.0/24 can be installed in the routing table. However if the Serial0/1/0 link is up/down or down/down it cannot be installed in the routing table, and any routes that recurse to Serial0/1/0 cannot be installed in the routing table. The result of this is that when the link is down traffic between the two Loopbacks of R4 and R5 is rerouted out the Frame Relay link using the prefix 150.1.0.0/16 as the longest match.

```
Rack1R5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/1/0
Rack1R5(config-if)#shutdown
Rack1R5(config-if)#end
Rack1R5#

Rack1R5#show ip route 150.1.4.4
Routing entry for 150.1.0.0/16
  Known via "static", distance 1, metric 0
  Routing Descriptor Blocks:
  * 155.1.0.4
      Route metric is 0, traffic share count is 1

Rack1R5#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.0.4 32 msec *  28 msec
```

Note

Be sure to "no shut" R5's S0/1/0 interface before proceeding to the next lab.

Copyright © 2010 Internetwork Expert

3.4 Floating Static Routes

- Remove the previous static routes on R4 and R5.
- Configure R4 and R5 with identical static routes for each others' Loopback 0 networks out both their Frame Relay connections and the point-to-point connection between them.
- Using administrative distance ensure that traffic between R4 and R5's Loopback 0 networks transits the point-to-point link between them, but is rerouted out the Frame Relay connection if the point-to-point link is down.

Configuration

```
R4:
ip route 150.1.5.0 255.255.255.0 Serial0/1/0 10
ip route 150.1.5.0 255.255.255.0 155.1.0.5 20

R5:
ip route 150.1.4.0 255.255.255.0 Serial0/1/0 10
ip route 150.1.4.0 255.255.255.0 155.1.0.4 20
```

Verification

Note

When the router does a lookup for a destination and there are multiple routes with the same longest match, the administrative distance is compared. The route with the lower administrative distance is installed in the routing table. With static routing this principle can be used for simple redundancy by configuring a backup route with a higher administrative distance than the primary route.

In this example R5 installs the route to 150.1.4.0/24 via Serial0/1/0 with an administrative distance of 10. When the link Serial0/1/0 is down the route with the next lowest administrative distance, 150.1.4.0/24 via 155.1.0.4 with a distance of 20, is installed. The result is that traffic is routed out the point-to-point link unless it is down, in which case traffic is rerouted out the Frame Relay link.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip route 150.1.4.4
Routing entry for 150.1.4.0/24
  Known via "static", distance 10, metric 0 (connected)
  Routing Descriptor Blocks:
  * directly connected, via Serial0/1/0
      Route metric is 0, traffic share count is 1

Rack1R5#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.45.4 12 msec *  12 msec

Rack1R5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/1/0
Rack1R5(config-if)#shutdown
Rack1R5(config-if)#end
Rack1R5#show ip route 150.1.4.4
Routing entry for 150.1.4.0/24
  Known via "static", distance 20, metric 0
  Routing Descriptor Blocks:
  * 155.1.0.4
      Route metric is 0, traffic share count is 1

Rack1R5#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.0.4 28 msec *  28 msec
```

Note

Be sure to “no shut” R5’s S0/1/0 interface before proceeding to the next lab.

Copyright © 2010 Internetwork Expert

3.5 Backup Interface

- Modify the administrative distance of R4's static routes to R5's Loopback 0 interface to be identical out the Frame Relay and point-to-point interfaces.
- Configure the backup interface feature on R4 to route traffic for R5's Loopback 0 out the Frame Relay network unless the PVC to R5 is down.
- If the PVC to R5 is down this traffic should be rerouted out the point-to-point link.
- Ensure the backup link is activate 3 seconds after the main link fails, and deactivated once the main link is active for 60 seconds.

Configuration

```
R4:
ip route 150.1.5.0 255.255.255.0 Serial0/1/0
ip route 150.1.5.0 255.255.255.0 155.1.0.5
!
interface Serial0/0/0.1 point-to-point
 backup delay 3 60
 backup interface Serial0/1/0
```

Verification

> **Note**
>
> In this example R4 uses the backup interface feature along with duplicate routing information to perform both traffic engineering and redundancy. With the backup interface configured on R4's point-to-point subinterface to R5, R4 waits for the line protocol of interface Serial0/0/0.1 to go down before interface Serial0/1/0 is activated. Since the line protocol status of Serial0/0/0.1 is based on the PVC status of DLCI 405, any failure in the end-to-end LMI path will notify R4 of a problem, and activate the backup link. Once the backup link is up, the route to 150.1.5.0/24 can be installed via Serial0/1/0.

```
Rack1R4#show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
FastEthernet0/0        204.12.1.4      YES manual up                    up
Serial0/0/0            unassigned      YES unset  up                    up
Serial0/0/0.1          155.1.0.4       YES manual up                    up
FastEthernet0/1        155.1.146.4     YES manual up                    up
Serial0/1/0            155.1.45.4      YES manual standby mode          down
Loopback0              150.1.4.4       YES manual up                    up

Rack1R4#show backup
Primary Interface   Secondary Interface   Status
-----------------   -------------------   ------
Serial0/0/0.1       Serial0/1/0           normal operation
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ip route 150.1.5.5
Routing entry for 150.1.5.0/24
  Known via "static", distance 1, metric 0
  Routing Descriptor Blocks:
  * 155.1.0.5
      Route metric is 0, traffic share count is 1

Rack1R4#traceroute 150.1.5.5

Type escape sequence to abort.
Tracing the route to 150.1.5.5

  1 155.1.0.5 28 msec *  28 msec
```

Once the circuit status for PVC 405 on R4 changes to INACTIVE, the line protocol state of Serial0/0/0.1 is down. Three seconds later the backup interface is taken out of standby mode, and the new routing information is installed in the routing table.

```
Rack1R4#debug backup
Backup events debugging is on

Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#shutdown
Rack1R5(config-if)#

Rack1R4#
Jun 30 11:59:20: BACKUP(Serial0/0/0.1): event = primary interface went down
Jun 30 11:59:20: BACKUP(Serial0/0/0.1): changed state to "waiting to backup"
Jun 30 11:59:23: BACKUP(Serial0/0/0.1): event = timer expired on primary
Jun 30 11:59:23: BACKUP(Serial0/0/0.1): secondary interface (Serial0/1/0) made
active
Jun 30 11:59:23: BACKUP(Serial0/0/0.1): changed state to "backup mode"
%LINK-3-UPDOWN: Interface Serial0/1/0, changed state to up
Jun 30 11:59:25: BACKUP(Serial0/1/0): event = secondary interface came up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0, changed state to
up
Jun 30 11:59:26: BACKUP(Serial0/1/0): event = secondary interface came up

Rack1R4#show ip route 150.1.5.5
Routing entry for 150.1.5.0/24
  Known via "static", distance 1, metric 0 (connected)
  Routing Descriptor Blocks:
  * directly connected, via Serial0/1/0
      Route metric is 0, traffic share count is 1

Rack1R4#traceroute 150.1.5.5

Type escape sequence to abort.
Tracing the route to 150.1.5.5

 1 155.1.45.5 16 msec *  12 msec
```

Copyright © 2010 Internetwork Expert

Once the circuit status for PVC 405 on R4 changes to ACTIVE, the line protocol state of Serial0/0/0.1 returns to up. Sixty seconds later the backup interface is sent back into standby state. If dynamic routing were configured in this topology the backup delay would allow time for the routing domain to reconverge before failing back over to the primary link.

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#no shutdown
Rack1R5(config-if)#

Rack1R4#
Jun 30 12:00:10: BACKUP(Serial0/0/0.1): event = primary interface came up
Jun 30 12:00:10: BACKUP(Serial0/0/0.1): changed state to "waiting to revert"
Jun 30 12:01:10: BACKUP(Serial0/0/0.1): event = timer expired on primary
Jun 30 12:01:10: BACKUP(Serial0/0/0.1): secondary interface (Serial0/1/0) moved
to standby
Jun 30 12:01:10: BACKUP(Serial0/0/0.1): changed state to "normal operation"
%LINK-5-CHANGED: Interface Serial0/1/0, changed state to standby mode
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0, changed state to
down
Jun 30 12:01:13: BACKUP(Serial0/1/0): event = secondary interface went down

Rack1R4#traceroute 150.1.5.5

Type escape sequence to abort.
Tracing the route to 150.1.5.5

  1 155.1.0.5 32 msec *  28 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

3.6 Reliable Static Routing with Enhanced Object Tracking

- Remove all previous routing configurations.
- Configure static routes on R5 so that it has reachability to the Loopback 0 networks of R1 and R4 via the Frame Relay network.
- Configure R1 with a static route to R4's Loopback 0 network via the Frame Relay connection to R5.
- Configure R4 with a static route to R1's Loopback 0 network via the Ethernet connection to VLAN 146.
- Configure a second static route on R4 for R1's Loopback 0 network via the Frame Relay connection to R5 with an administrative distance of 2.
- Configure an IP SLA instance on R4 to ping R1's Ethernet connection to VLAN 146 every 5 seconds.
- Create an enhanced object to track this SLA instance.
- If the SLA instance receives no response in a period of 2 seconds the instance should report itself as being "DOWN".
- The IP SLA should start immediately and never expire.
- Tie this enhanced object to the static route R4 has for R1's Loopback 0 out the connection to VLAN 146.
- Ensure that R4 maintains IP reachability to R1's Loopback 0 in the case that R1's Ethernet interface is down.

Configuration

```
R1:
ip route 150.1.4.0 255.255.255.0 155.1.0.4

R4:
ip sla 1
 icmp-echo 155.1.146.1 source-interface FastEthernet0/1
 timeout 2000
 frequency 5
ip sla schedule 1 life forever start-time now
!
track 1 rtr 1
!
ip route 150.1.1.0 255.255.255.0 155.1.146.1 track 1
ip route 150.1.1.0 255.255.255.0 155.1.0.5 2

R5:
ip route 150.1.1.0 255.255.255.0 155.1.0.1
ip route 150.1.4.0 255.255.255.0 155.1.0.4
```

Copyright © 2010 Internetwork Expert

Verification

Note

Although R1 and R4 are on the same layer 2 segment for VLAN 146, their physical Ethernet interfaces are not on the same layer 1 network. This means that the link status of R1's connection to VLAN 146 is independent of R4's connection. From a redundancy design point of view the possible problem with this is that R4 has no way to know whether or not R1's link is actually up. In point-to-point designs, like the previous examples with R4 and R5's directly connected Serial link, this is not an issue because the line protocol status of one side of the link dictates the line protocols status of the other side.

In the example below we can see that even though R1's link to VLAN 146 is down, R4 still installs the route to 150.1.1.0/24 via 155.1.146.1, since R4 has no way to know if 155.1.146.1 is available. The result of this situation is that if R1's link to VLAN 146 goes down traffic between the Loopbacks of R1 and R4 will dropped.

We can verify this problem before configuring the "ip sla" instance and enhanced object tracking features on R4:

```
Rack1R1#ping 150.1.4.4 source 150.1.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
Packet sent with a source address of 150.1.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/60 ms

Rack1SW1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface Fa0/1
Rack1SW1(config-if)#shutdown
Rack1SW1(config-if)#

Rack1R4#show ip route 150.1.1.1
Routing entry for 150.1.1.0/24
  Known via "static", distance 1, metric 0
  Routing Descriptor Blocks:
  * 155.1.146.1
      Route metric is 0, traffic share count is 1

Rack1R1#ping 150.1.4.4 source 150.1.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
Packet sent with a source address of 150.1.1.1
 ...
```

Copyright © 2010 Internetwork Expert

In order to fix this design the IP Service Level Agreement (SLA) and Enhanced Object Tracking features are introduced. First, R4 creates an SLA instance that sends an ICMP ping to R1 via VLAN 146 every five seconds. If a response is not received within 2000 milliseconds the SLA instance reports its status as down.

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#ip sla 1
Rack1R4(config-sla-monitor)#icmp-echo 155.1.146.1 source-interface
FastEthernet0/1
Rack1R4(config-sla-monitor-echo)#frequency 5
Rack1R4(config-sla-monitor-echo)#timeout 2000
Rack1R4(config-sla-monitor-echo)#exit
Rack1R4(config)#ip sla schedule 1 life forever start-time now

Rack1R4(config)#end

Rack1R4#show ip sla statistics 1
IPSLAs Latest Operation Statistics

IPSLA operation id: 1
Type of operation: icmp-echo
      Latest RTT: 1 milliseconds
Latest operation start time: 08:54:30.191 UTC Tue Feb 9 2010
Latest operation return code: OK
Number of successes: 174
Number of failures: 143
Operation time to live: Forever
```

Next an Enhanced Object Tracking is created that watches the IP SLA instance. This tracking says that if SLA instance 1 is down, tracking instance 1 is down, and the static route to R1's Loopback cannot be installed in the routing table.

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#track 1 rtr 1
Rack1R4(config-track)#exit
Rack1R4(config)#no ip route 150.1.1.0 255.255.255.0 155.1.146.1
Rack1R4(config)#ip route 150.1.1.0 255.255.255.0 155.1.146.1 track 1
Rack1R4(config)#end
Rack1R4#show track 1
Track 1
  IP SLA 1 state
  State is Up
    3 changes, last change 00:01:14
  Latest operation return code: OK
  Latest RTT (millisecs) 1
  Tracked by:
    STATIC-IP-ROUTING 0
```

Copyright © 2010 Internetwork Expert

The same failure scenario is re-introduced, however with the reliable static route R4 can now reroute around this network failure.

```
Rack1R4#traceroute 150.1.1.1 source 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.1.1

  1 155.1.146.1 36 msec *  36 msec

Rack1R4#debug track

Rack1R4#debug ip routing
IP routing debugging is on

Rack1SW1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface Fa0/1
Rack1SW1(config-if)#shutdown
Rack1SW1(config-if)#

Rack1R4#
Track: 1 Change #2 rtr 1, state Up->Down
RT: del 150.1.1.0/24 via 155.1.146.1, static metric [1/0]
RT: delete subnet route to 150.1.1.0/24
RT: NET-RED 150.1.1.0/24
RT: SET_LAST_RDB for 150.1.1.0/24
  NEW rdb: via 155.1.0.5

RT: add 150.1.1.0/24 via 155.1.0.5, static metric [2/0]
RT: NET-RED 150.1.1.0/24

Rack1R4#traceroute 150.1.1.1 source 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.1.1

  1 155.1.0.5 28 msec 28 msec 28 msec
  2 155.1.0.1 56 msec *  56 msec

Rack1R4#show ip route 150.1.1.1
Routing entry for 150.1.1.0/24
  Known via "static", distance 2, metric 0
  Routing Descriptor Blocks:
  * 155.1.0.5
      Route metric is 0, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

Once R1's connection to VLAN 146 comes back, the SLA instance reports itself as back up, the tracking instance reports itself as back up, and the static route with the lower administrative distance is reinstalled in the routing table.

```
Rack1SW1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface Fa0/1
Rack1SW1(config-if)#no shutdown
Rack1SW1(config-if)#

Rack1R4#
Track: 1 Change #3 rtr 1, state Down->Up
RT: closer admin distance for 150.1.1.0, flushing 1 routes
RT: SET_LAST_RDB for 150.1.1.0/24
  NEW rdb: via 155.1.146.1

RT: add 150.1.1.0/24 via 155.1.146.1, static metric [1/0]
RT: NET-RED 150.1.1.0/24

Rack1R4#show ip route 150.1.1.1
Routing entry for 150.1.1.0/24
  Known via "static", distance 1, metric 0
  Routing Descriptor Blocks:
  * 155.1.146.1
      Route metric is 0, traffic share count is 1

Rack1R4#traceroute 150.1.1.1 source 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.1.1

  1 155.1.146.1 40 msec *  36 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

3.7 Policy Routing

- Remove all previous routing configurations.
- Configure default routes on R4 and R6 pointing to the VLAN 146 connection of R1.
- Configure a default route on R3 pointing to the point-to-point link connecting to R1.
- Configure a default route on R5 pointing to R1 out the Frame Relay network.
- Configure a static route on R3 for R5's Loopback 0 network and a static route on R5 for R3's Loopback 0 network out the Frame Relay network.
- Create two extended access-lists on R1, one named FROM_R4 and the other named FROM_R6.
- Access-list FROM_R4 should match all IP packets coming from R4, while access-list FROM_R6 should match all IP packets coming from R6.
- Configure policy-routing on R1 so that traffic from R4 is routed out the serial link between R1 and R3, and traffic from R6 is routed out the Frame Relay link to R5.
- Use traceroute on R4 and R6 for R3 and R5's Loopback 0 networks to verify that this configuration is functional.

Configuration

```
R1:
interface FastEthernet0/0
 ip policy route-map POLICY_ROUTING
!
ip access-list extended FROM_R4
 permit ip host 155.1.146.4 any
ip access-list extended FROM_R6
 permit ip host 155.1.146.6 any
!
route-map POLICY_ROUTING permit 10
 match ip address FROM_R4
 set ip next-hop 155.1.13.3
!
route-map POLICY_ROUTING permit 20
 match ip address FROM_R6
 set ip next-hop 155.1.0.5

R3:
ip route 0.0.0.0 0.0.0.0 155.1.13.1
ip route 150.1.5.0 255.255.255.0 155.1.0.5

R4:
ip route 0.0.0.0 0.0.0.0 155.1.146.1
```

Copyright © 2010 Internetwork Expert

```
R5:
ip route 0.0.0.0 0.0.0.0 155.1.0.1
ip route 150.1.3.0 255.255.255.0 155.1.0.3

R6:
ip route 0.0.0.0 0.0.0.0 155.1.146.1
```

Verification

> **Note**
>
> Policy routing allows the router to forward traffic based on user-defined criteria before the normal IP routing table is consulted. In this example we can see that R1 does not have routing information for either of the Loopbacks of R3 and R5, so it cannot send traffic there when it is locally generated.

```
Rack1R1#show ip route 150.1.3.3
% Subnet not in table

Rack1R1#show ip route 150.1.5.5
% Subnet not in table

Rack1R1#ping 150.1.3.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.3.3, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)

Rack1R1#ping 150.1.5.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.5.5, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)
```

Copyright © 2010 Internetwork Expert

If traffic comes in from R4 or R6's IP addresses attached to VLAN 146 it is routed according to the route-map attached to the incoming Fa0/0 interface.

```
Rack1R1#debug ip policy
Policy routing debugging is on

Rack1R4#traceroute 150.1.3.3

Type escape sequence to abort.
Tracing the route to 150.1.3.3

  1 155.1.146.1 4 msec 4 msec 0 msec
  2 155.1.13.3 16 msec *  16 msec

Rack1R1#
IP: s=155.1.146.4 (FastEthernet0/0), d=150.1.3.3, len 28, FIB policy match
IP: s=155.1.146.4 (FastEthernet0/0), d=150.1.3.3, g=155.1.13.3, len 28, FIB
policy routed

Rack1R4#traceroute 150.1.5.5

Type escape sequence to abort.
Tracing the route to 150.1.5.5

  1 155.1.146.1 0 msec 0 msec 4 msec
  2 155.1.13.3 16 msec 20 msec 16 msec
  3 155.1.0.5 36 msec *  36 msec

Rack1R1#
IP: s=155.1.146.4 (FastEthernet0/0), d=150.1.5.5, len 28, FIB policy match
IP: s=155.1.146.4 (FastEthernet0/0), d=150.1.5.5, g=155.1.13.3, len 28, FIB
policy routed

Rack1R6#traceroute 150.1.3.3

Type escape sequence to abort.
Tracing the route to 150.1.3.3

  1 155.1.146.1 0 msec 4 msec 0 msec
  2 155.1.0.5 28 msec 32 msec 33 msec
  3 155.1.0.3 32 msec *  32 msec

Rack1R1#
IP: s=155.1.146.6 (FastEthernet0/0), d=150.1.3.3, len 28, FIB policy match
IP: s=155.1.146.6 (FastEthernet0/0), d=150.1.3.3, g=155.1.0.5, len 28, FIB
policy routed

Rack1R6#traceroute 150.1.5.5

Type escape sequence to abort.
Tracing the route to 150.1.5.5

  1 155.1.146.1 0 msec 4 msec 0 msec
  2 155.1.0.5 28 msec *  28 msec

Rack1R1#
IP: s=155.1.146.6 (FastEthernet0/0), d=150.1.5.5, len 28, FIB policy match
IP: s=155.1.146.6 (FastEthernet0/0), d=150.1.5.5, g=155.1.0.5, len 28, FIB
policy routed
```

Copyright © 2010 Internetwork Expert

R1's route-map used for policy routing does not match traffic sourced from R6's Loopback 0 interface, so this traffic is dropped when it is received in the link to VLAN 146.

```
Rack1R6#ping 150.1.5.5 source 150.1.6.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.5.5, timeout is 2 seconds:
Packet sent with a source address of 150.1.6.6
....
Success rate is 0 percent (0/5)

Rack1R1#
IP: s=150.1.6.6 (FastEthernet0/0), d=150.1.5.5, len 28, FIB policy rejected(no
match) - normal forwarding
IP: s=150.1.6.6 (FastEthernet0/0), d=150.1.5.5, len 28, policy rejected --
normal forwarding
```

3.8 Reliable Policy Routing

- Remove the previous static and policy routing configurations.
- Remove the point-to-point Frame Relay subinterface on R1 and replace this with a static mapping to R5 on R1's main Serial0/0 interface.
- Configure a default route on R4 pointing to the VLAN 146 connection of R1.
- Configure a default route on R3 pointing to the point-to-point link connecting to R1.
- Configure a default route on R5 pointing to R1 out the Frame Relay network.
- Configure a route for R4's Loopback 0 network on R5 via their point-to-point Serial link.
- Configure a route for R5's Loopback 0 network on R4 via their point-to-point Serial link.
- Configure R1 and R5 to run CDP over the Frame Relay network with each other.
- Configure an IP SLA instance on R1 that pings R4's connection to VLAN 146 every five seconds.
- Configure policy routing on R1 so that traffic from R3 going to R4's Loopback0 network is sent to R5 over the Frame Relay link, while traffic to R5's Loopback0 network is sent to R4 over VLAN 146.
- Use traceroute to verify that this configuration is functional.
- Modify the R1's policy routing so that if R1 loses R5 as a CDP neighbor traffic from R3 to R4's Loopback0 network is rerouted directly to R4.
- Modify the R1's policy routing so that if R1 loses ICMP reachability to R4 traffic from R3 to R5's Loopback0 network is rerouted directly to R5.
- Use traceroute to verify that this configuration is functional.

Copyright © 2010 Internetwork Expert

Configuration

```
R1:
ip sla monitor 1
 type echo protocol ipIcmpEcho 155.1.146.4 source-interface
FastEthernet0/0
 timeout 2000
 frequency 5
!
ip sla monitor schedule 1 life forever start-time now
!
track 1 rtr 1
!
interface Serial0/0
 ip address 155.1.0.1 255.255.255.0
 encapsulation frame-relay
 cdp enable
 frame-relay map ip 155.1.0.5 105 broadcast
 no frame-relay inverse-arp
!
interface Serial0/1
 ip policy route-map RELIABLE_POLICY_ROUTING
!
ip access-list extended FROM_R3_TO_R4_LOOPBACK
 permit ip host 155.1.13.3 host 150.1.4.4
!
ip access-list extended FROM_R3_TO_R5_LOOPBACK
 permit ip host 155.1.13.3 host 150.1.5.5
!
route-map RELIABLE_POLICY_ROUTING permit 10
 match ip address FROM_R3_TO_R4_LOOPBACK
 set ip next-hop 155.1.0.5
 set ip next-hop verify-availability
 set ip default next-hop 155.1.146.4
!
route-map RELIABLE_POLICY_ROUTING permit 20
 match ip address FROM_R3_TO_R5_LOOPBACK
 set ip next-hop verify-availability 155.1.146.4 1 track 1
 set ip default next-hop 155.1.0.5

R3:
ip route 0.0.0.0 0.0.0.0 155.1.13.1

R4:
ip route 0.0.0.0 0.0.0.0 155.1.146.1
ip route 150.1.5.0 255.255.255.0 155.1.45.5

R5:
interface Serial0/0/0
 cdp enable
!
ip route 0.0.0.0 0.0.0.0 155.1.0.1
ip route 150.1.4.0 255.255.255.0 155.1.45.4
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> In this example of policy routing R1 matches traffic as it comes in Serial0/1 from R3, and routes the traffic based on both the source and the destination. The traceroute output on R3 indicates that the traffic is policy routed correctly.

```
Rack1R3#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.13.1 16 msec 16 msec 12 msec
  2 155.1.0.5 40 msec 44 msec 40 msec
  3 155.1.45.4 32 msec *  32 msec

Rack1R3#traceroute 150.1.5.5

Type escape sequence to abort.
Tracing the route to 150.1.5.5

  1 155.1.13.1 16 msec 16 msec 16 msec
  2 155.1.146.4 16 msec 20 msec 16 msec
  3 155.1.45.5 40 msec *  36 msec
```

Since R1's policy routing configuration is only locally significant, network failures do not automatically update the routing policy of R1. As seen below if either R5's connection to the Frame Relay network goes down or R4's connection to VLAN 146 goes down, traffic will be blackholed. This is based on the fact that the next-hop values of 155.1.0.5 and 155.1.146.4 can still be recursed in the routing table, because the interfaces R1 uses to reach them do not go down when the other side of the link fails.

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#shutdown
Rack1R5(config-if)#

Rack1R3#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.13.1 16 msec 12 msec 12 msec
  2  *  *  *
```

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#no shutdown
Rack1R5(config-if)#

Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface FastEthernet0/1
Rack1R4(config-if)#shutdown
Rack1R4(config-if)#

Rack1R3#traceroute 150.1.5.5

Type escape sequence to abort.
Tracing the route to 150.1.5.5

  1 155.1.13.1 16 msec 16 msec 16 msec
  2  *  *  *
```

In order to resolve this design problem R1 needs some way to track end-to-end reachability on these links used for the outbound forwarding through policy routing. The two ways illustrated in this example are through the IP SLA and Enhanced Object Tracking features, and through CDP.

With IP SLA configured R1 tracks the end-to-end circuit status of VLAN 146 through ICMP ping. When R4's connection to VLAN 146 goes down, R1's SLA instance reports its status down, which in turn causes the tracked object to do down. The tracked object is called from the route-map syntax **`set ip next-hop verify-availability 155.1.146.4 1 track 1`**. This means that if tracked object 1 goes down, do not use the next-hop 155.1.146.4. Instead this route-map sequence fails over to the "default" next-hop of 155.1.0.5. The result of this is seen through the below traceroute of R3 and debug ip policy output of R1.

```
Rack1R1#debug track
Rack1R1#debug ip policy
Policy routing debugging is on

Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface FastEthernet0/1
Rack1R4(config-if)#shutdown
Rack1R4(config-if)#

Rack1R1#
Track: 1 Change #2 rtr 1, state Up->Down
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#traceroute 150.1.5.5

Type escape sequence to abort.
Tracing the route to 150.1.5.5

  1 155.1.13.1 16 msec 16 msec 16 msec
  2 155.1.0.5 44 msec *  40 msec

Rack1R1#
IP: s=155.1.13.3 (Serial0/1), d=150.1.5.5, len 28, FIB policy match
IP: s=155.1.13.3 (Serial0/1), d=150.1.5.5, g=155.1.0.5, len 28, FIB
policy routed
```

With CDP tracking for policy routing R1 looks into the CDP table to see if there is a neighbor installed with the IP address that matches the next-hop value being set in the route-map. In this case the syntax **set ip next-hop 155.1.0.5**, **set ip next-hop verify-availability** and **set ip default next-hop 155.1.146.4** means if there is no CDP neighbor with the IP address 155.1.0.5 traffic that matches this sequence will be routed to 155.1.146.4. This can be seen from the below output when R5's Frame Relay interface fails.

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface FastEthernet0/1
Rack1R4(config-if)#no shutdown
Rack1R4(config-if)#

Rack1R1#show cdp neighbors
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
Rack1SW1         Fas 0/0            155          S I       WS-C3560- Fas 0/1
Rack1R3          Ser 0/1            152         R S I      2611XM    Ser 1/2
Rack1R5          Ser 0/0            102         R S I      2611XM    Ser 0/0/0

Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#shutdown
Rack1R5(config-if)#

Rack1R1#show cdp neighbors
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
Rack1SW1         Fas 0/0            137          S I       WS-C3560- Fas 0/1
Rack1R3          Ser 0/1            134         R S I      2611XM    Ser 1/2
```

```
Rack1R3#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.13.1 16 msec 12 msec 12 msec
  2 155.1.146.4 20 msec *  16 msec

Rack1R1#
IP: s=155.1.13.3 (Serial0/1), d=150.1.4.4, len 28, FIB policy match
IP: s=155.1.13.3 (Serial0/1), d=150.1.4.4, g=155.1.146.4, len 28, FIB
policy routed
```

Note

Ensure that you “no shut” any interfaces shutdown during verification testing before proceeding to the next lab.

Copyright © 2010 Internetwork Expert

3.9 Local Policy Routing

- Create two access-lists named TO_R4 and TO_R5 on R1.
- Access-list TO_R4 should match all IP packets going to the Loopback0 network of R4, while access-list TO_R5 should match all packets going to the Loopback0 network of R5.
- Configure local policy-routing on R1 so that locally generated traffic matched by the list TO_R5 is routed out the Ethernet link to R4, and traffic matched by the access-list TO_R4 is routed out the Frame Relay link to R5.
- Use traceroute on R1 for R4 and R5's Loopback 0 networks to verify that this configuration is functional.

Configuration

```
R1:
ip local policy route-map LOCAL_POLICY
!
ip access-list extended TO_R4
 permit ip any host 150.1.4.4
!
ip access-list extended TO_R5
 permit ip any host 150.1.5.5
!
route-map LOCAL_POLICY permit 10
 match ip address TO_R4
 set ip next-hop 155.1.0.5
!
route-map LOCAL_POLICY permit 20
 match ip address TO_R5
 set ip next-hop 155.1.146.4
```

Verification

> **Note**
>
> Local policy routing is similar in operation to normal policy routing, except it affects locally generated traffic from the router instead of traffic received inbound on an interface. In the below output we can see that R1 does not have a normal route to either of the destinations 150.1.4.4 or 150.1.5.5, however traffic is successfully routed due to the locally configured policy.

```
Rack1R1#show ip route 150.1.4.4
% Subnet not in table
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.0.5 32 msec 28 msec 32 msec
  2 155.1.45.4 16 msec *  16 msec

Rack1R1#show ip route 150.1.5.5
% Subnet not in table

Rack1R1#traceroute 150.1.5.5

Type escape sequence to abort.
Tracing the route to 150.1.5.5

  1 155.1.146.4 4 msec 4 msec 0 msec
  2 155.1.45.5 28 msec *  24 msec
```

Pitfall

Note that when the remote devices receive traffic from R1 it is sourced from the Loopback 0 interface of R1. Normally the router uses the IP address of the outgoing interface in the routing table as the source IP address in its own packets. However since the routing table is not consulted for the lookup you may see inconsistencies in what the source address of the local traffic is. This behavior could have a negative impact on protocols such as BGP which need to agree on what the source and destination IP addresses are for a peering.

```
Rack1R5#debug ip icmp
ICMP packet debugging is on

Rack1R1#ping 150.1.5.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.5.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 44/44/45 ms

Rack1R5#
Jun 30 16:59:40.118: ICMP: echo reply sent, src 150.1.5.5, dst 150.1.1.1
Jun 30 16:59:40.166: ICMP: echo reply sent, src 150.1.5.5, dst 150.1.1.1
Jun 30 16:59:40.210: ICMP: echo reply sent, src 150.1.5.5, dst 150.1.1.1
Jun 30 16:59:40.254: ICMP: echo reply sent, src 150.1.5.5, dst 150.1.1.1
Jun 30 16:59:40.302: ICMP: echo reply sent, src 150.1.5.5, dst 150.1.1.1
```

Copyright © 2010 Internetwork Expert

3.10 GRE Tunneling

- Remove all previous static and policy routing configurations.
- Enable RIPv2 on all links in the 150.X.0.0 and 155.X.0.0 address space on all internal devices and disable auto-summary.
- Create a GRE tunnel between SW3 and SW4 using the IP addresses 10.34.0.Y/24.
- The tunnel should be sourced from and destined to these devices Loopback 0 networks.
- Create new Loopback 1 interfaces on SW3 and SW4 with the IP addresses 172.16.0.Y/32.
- Configure static routes so that traffic from SW3 to SW4's new Loopback 1 interface is routed over the tunnel, and vice-versa.
- Use traceroute on SW3 and SW4 to verify that this configuration is functional.

Configuration

```
R1 - R6:
router rip
 version 2
 no auto-summary
 network 150.1.0.0
 network 155.1.0.0

SW1 & SW2:
ip routing
!
router rip
 version 2
 no auto-summary
 network 150.1.0.0
 network 155.1.0.0
```

Copyright © 2010 Internetwork Expert

```
SW3:
ip routing
!
interface Tunnel0
 ip address 10.34.0.9 255.255.255.0
 tunnel source Loopback0
 tunnel destination 150.1.10.10
!
interface Loopback1
 ip address 172.16.0.9 255.255.255.255
!
ip route 172.16.0.10 255.255.255.255 Tunnel0
!
router rip
 version 2
 no auto-summary
 network 150.1.0.0
 network 155.1.0.0

SW4:
ip routing
!
interface Tunnel0
 ip address 10.34.0.10 255.255.255.0
 tunnel source Loopback0
 tunnel destination 150.1.9.9
!
interface Loopback1
 ip address 172.16.0.10 255.255.255.255
!
ip route 172.16.0.9 255.255.255.255 Tunnel0
!
router rip
 version 2
 no auto-summary
 network 150.1.0.0
 network 155.1.0.0
```

Copyright © 2010 Internetwork Expert

Verification

Note

Generic Routing Encapsulation (GRE) tunneling is used to take another protocol payload, such as IPv4, IPv6, IPX, etc., and tunnel it over an IPv4 transit network. In this case GRE is used to tunnel IPv4 packets between the 172.16.0.9/32 and 172.16.0.10/32 segments.

We can verify that traffic between these segments is going out the tunnel because the traceroute output shows only one hop between the networks, which is the tunnel.

```
Rack1SW4#show ip route 172.16.0.9
Routing entry for 172.16.0.9/32
  Known via "static", distance 1, metric 0 (connected)
  Routing Descriptor Blocks:
  * directly connected, via Tunnel0
      Route metric is 0, traffic share count is 1

Rack1SW4#traceroute 172.16.0.9

Type escape sequence to abort.
Tracing the route to 172.16.0.9

  1 10.34.0.9 40 msec *  40 msec
```

Copyright © 2010 Internetwork Expert

The second verification is in the transit path. From the output of **debug ip packet detail** we can see that R5 is forwarding IP protocol number 47 between SW3 and SW4, which is GRE.

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#no ip route-cache
Rack1R5(config-if)#interface FastEthernet0/0
Rack1R5(config-if)#no ip route-cache
Rack1R5(config-if)#end
Rack1R5#debug ip packet detail
IP packet debugging is on (detailed)
Rack1R5#

Rack1SW4#ping 172.16.0.9

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.0.9, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 76/76/80 ms

Rack1R5#
IP: s=150.1.10.10 (FastEthernet0/0), d=150.1.9.9 (Serial0/0),
g=155.1.0.3, len 124, forward, proto=47
IP: tableid=0, s=150.1.9.9 (Serial0/0/0), d=150.1.10.10
(FastEthernet0/0), routed via FIB
IP: s=150.1.9.9 (Serial0/0/0), d=150.1.10.10 (FastEthernet0/0),
g=155.1.58.8, len 124, forward, proto=47
IP: tableid=0, s=150.1.10.10 (FastEthernet0/0), d=150.1.9.9
(Serial0/0/0), routed via FIB
```

Pitfall

In order to debug this traffic on R5 the **no ip route-cache** command is needed on the transit interfaces in order to force the traffic to be process switched. Be very careful using this command in production networks as it means that all previously CEF switched traffic must use the CPU for forwarding on a per packet basis.

Copyright © 2010 Internetwork Expert

3.11 GRE Tunneling and Recursive Routing

- Enable RIPv2 on the tunnel interface between SW3 and SW4.
- Configure distribute-list filtering on SW3 and SW4 so that a recursive routing error does not occur for routing lookups out the tunnel interface between them.

Configuration

```
SW3:
router rip
 network 10.0.0.0
 distribute-list prefix STOP_RECURSIVE_ERROR out Tunnel0
!
ip prefix-list STOP_RECURSIVE_ERROR seq 5 deny 150.1.9.0/24
ip prefix-list STOP_RECURSIVE_ERROR seq 10 permit 0.0.0.0/0 le 32

SW4:
router rip
 network 10.0.0.0
 distribute-list prefix STOP_RECURSIVE_ERROR out Tunnel0
!
ip prefix-list STOP_RECURSIVE_ERROR seq 5 deny 150.1.10.0/24
ip prefix-list STOP_RECURSIVE_ERROR seq 10 permit 0.0.0.0/0 le 32
```

Verification

> **Note**
>
> A recursive routing error is when a lookup for prefix X points at prefix Y, and a lookup for prefix Y points at prefix X. For tunnel interfaces this occurs when the tunnel destination is dynamically learned in the tunnel interface itself. This problem can be easily identified by the IOS log message %TUN-5-RECURDOWN: Tunnel0 temporarily disabled due to recursive routing. The step-by-step process by which this error occurs is as follows.
>
> First, SW4 learns the route to the tunnel destination 150.1.9.9 via SW2.

```
Rack1SW4#debug ip routing
IP routing debugging is on

Rack1SW4#clear ip route *
Rack1SW4#
<output omitted>
RT: SET_LAST_RDB for 150.1.9.0/24
  NEW rdb: via 155.1.108.8

RT: add 150.1.9.0/24 via 155.1.108.8, rip metric [120/5]
<output omitted>
```

Copyright © 2010 Internetwork Expert

Now that there is a valid path for the tunnel control traffic, the tunnel interface comes up. With RIP enabled on the 10.0.0.0 network and no filtering, SW4 learns an alternate route for the tunnel destination.

```
Rack1SW4#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state
to up
<output omitted>
RT: del 150.1.9.0/24 via 155.1.108.8, rip metric [120/5]
RT: SET_LAST_RDB for 150.1.9.0/24
  OLD rdb: via 13.11.13.11

RT: SET_LAST_RDB for 150.1.9.0/24
  NEW rdb: via 10.34.0.9

RT: add 150.1.9.0/24 via 10.34.0.9, rip metric [120/1]
<output omitted>
```

With the reception of a new RIP route for 150.1.9.0/24 with a metric of 1, the old route with a metric of 5 is flushed out. The recursive error now occurs because 150.1.9.0/24 is reachable via 10.34.0.9, but 10.34.0.9 is the tunnel reachable via 150.1.9.9. The router detects this error, disables the tunnel, and all routes learned via the tunnel are flushed.

```
Rack1SW4#
%TUN-5-RECURDOWN: Tunnel0 temporarily disabled due to recursive routing
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state
to down
<output omitted>
RT: del 150.1.9.0/24 via 10.34.0.9, rip metric [120/1]
RT: delete subnet route to 150.1.9.0/24
<output omitted>
```

To solve this problem a distribute-list filter is put in place to never advertise the tunnel control networks 150.1.9.0/24 and 150.1.10.0/24 out the tunnel interface itself. This problem could also be solved by configuring static routes to these destinations with a lower administrative distance than the dynamically learned routes. Once complete the tunnel can be used for any destinations other than the Loopbacks of SW3 and SW4.

```
Rack1SW4#traceroute 155.1.9.9

Type escape sequence to abort.
Tracing the route to 155.1.9.9

  1 10.34.0.9 44 msec *  44 msec
```

Copyright © 2010 Internetwork Expert

3.12 Reliable Backup Interface with GRE

- Remove all previous routing configurations.
- Configure static routes on R5 for R4's Loopback 0 interface via both the Frame Relay and point-to-point connections between them.
- Configure static routes on R4 for R5's Loopback 0 interface via both the Frame Relay and point-to-point connections between them.
- The static routes on R4 and R5 via the Frame Relay circuit should have a higher administrative distance than those out the point-to-point link.
- Create a GRE tunnel between R4 and R5 using the addresses 10.0.0.Y/24 that is sourced from and destined to the Frame Relay connection between them, and enable GRE keepalives.
- Configure the backup interface feature on R5 such that if the tunnel between R4 and R5 goes down the point-to-point Serial link between them is activated.
- To verify this configuration ensure that traffic between the Loopback 0 interfaces of R4 and R5 is routed out the Frame Relay link between them, but if R4's Frame Relay interface is shut down this traffic is rerouted out the point-to-point link.

Configuration

```
R4:
ip route 150.1.5.0 255.255.255.0 155.1.0.5 20
ip route 150.1.5.0 255.255.255.0 155.1.45.5 10
!
interface Tunnel0
 ip address 10.0.0.4 255.255.255.0
 tunnel source 155.1.0.4
 tunnel destination 155.1.0.5
 keepalive 1 3

R5:
ip route 150.1.4.0 255.255.255.0 155.1.0.4 20
ip route 150.1.4.0 255.255.255.0 155.1.45.4 10
!
interface Tunnel0
 ip address 10.0.0.5 255.255.255.0
 tunnel source 155.1.0.5
 tunnel destination 155.1.0.4
 keepalive 1 3
 backup interface Serial0/1/0
```

Copyright © 2010 Internetwork Expert

Verification

Note

In the previous backup design, R4 configured the **backup interface** command on its point-to-point Frame Relay subinterface. Since the backup interface feature tracks the line protocol status of the link, and the point-to-point subinterface line protocol status is based on the Frame Relay PVC status, if there is a failure anywhere in the end-to-end LMI path R4 will be able to detect this and switch over to the backup link.

In this design case the backup interface command is configured on R5's main Serial0/0/0 interface. Since the line protocol status of this interface is based on the LMI keepalive status from the local Frame Relay switch, not the end-to-end PVC status, R5 cannot trigger the backup link when there is a failure of the circuit to R4. While the Frame Relay circuit is up we can see that R5 routes across this link to reach R4's Loopback 0 network. This is due to the fact that the route with the lower administrative distance via Serial0/1/0 cannot be installed because this link is in standby state.

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#backup interface Serial0/1/0
Rack1R5(config-if)#end
%LINK-5-CHANGED: Interface Serial0/1/0, changed state to standby mode
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0, changed
state to down

Rack1R5#show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
FastEthernet0/0        155.1.58.5      YES manual up                    up
Serial0/0/0            155.1.0.5       YES manual up                    up
FastEthernet0/1        155.1.5.5       YES manual up                    up
Serial0/1/0            155.1.45.5      YES manual standby mode          down
Loopback0              150.1.5.5       YES manual up                    up

Rack1R5#show ip route 150.1.4.4
Routing entry for 150.1.4.0/24
  Known via "static", distance 20, metric 0
  Routing Descriptor Blocks:
  * 155.1.0.4
      Route metric is 0, traffic share count is 1

Rack1R5#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.0.4 28 msec *  28 msec
```

If there is a failure in the Frame Relay network that is not on the local link between R5 and the Frame Relay switch, traffic becomes black holed.

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface Serial0/0/0
Rack1R4(config-if)#shutdown
Rack1R4(config-if)#
```

Although R5 updates DLCI 504's circuit status to INACTIVE, this does not cause the line protocol of Serial0/0 to go down.

```
Rack1R5#show frame-relay pvc | include DLCI
DLCI = 501, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0
DLCI = 502, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0
DLCI = 503, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0
DLCI = 504, DLCI USAGE = LOCAL, PVC STATUS = INACTIVE, INTERFACE = Serial0/0/0
DLCI = 513, DLCI USAGE = UNUSED, PVC STATUS = INACTIVE, INTERFACE = Serial0/0/0

Rack1R5#show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
FastEthernet0/0        155.1.58.5      YES manual up                    up
Serial0/0/0            155.1.0.5       YES manual up                    up
FastEthernet0/1        155.1.5.5       YES manual up                    up
Serial0/1/0            155.1.45.5      YES manual standby mode          down
Loopback0              150.1.5.5       YES manual up                    up
```

Since Serial0/0/0 is still installed in the routing table, the route to 150.1.4.0/24 via 155.1.0.4 is still installed in the routing table, and traffic to the final destination is dropped inside of the Frame Relay network.

```
Rack1R5#show ip route 150.1.4.4
Routing entry for 150.1.4.0/24
  Known via "static", distance 10, metric 0
  Routing Descriptor Blocks:
  * 155.1.0.4
      Route metric is 0, traffic share count is 1

Rack1R5#ping 150.1.4.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)
```

Copyright © 2010 Internetwork Expert

Similar to using the IP SLA feature to add reliability to static routing, the backup interface feature can become reliable by using a GRE tunnel and tunnel keepalives. By configuring a tunnel with keepalives between R4 and R5 sourced from the Frame Relay interface, these routers will be able to detect if end-to-end IP transport is lost between these interfaces. Furthermore if the backup interface command is configured on the tunnel interface, instead of the main Serial0/0/0 interface, R5 will be able to switch to the backup link if there is a failure anywhere in the end-to-end transit of the Frame Relay PVC.

The following output demonstrates the same failure scenario as before, but now R5 is configured with the backup interface command on the tunnel with keepalives instead of the Serial0/0/0 interface.

```
Rack1R5#debug tunnel keepalive
Tunnel keepalive debugging is on
Rack1R5#debug backup
Backup events debugging is on
Tunnel0: sending keepalive, 155.1.0.4->155.1.0.5 (len=24 ttl=255),
counter=1
Tunnel0: keepalive received, 155.1.0.4->155.1.0.5 (len=24 ttl=253),
resetting counter

Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface Serial0/0/0
Rack1R4(config-if)#shutdown
Rack1R4(config-if)#
```

With R4's Frame Relay interface down it cannot reply to the tunnel keepalives. R5 detects this, drops the line protocol status of the tunnel, and activates the backup interface.

```
Rack1R5#
Tunnel0: sending keepalive, 155.1.0.4->155.1.0.5 (len=24 ttl=255), counter=1
Tunnel0: sending keepalive, 155.1.0.4->155.1.0.5 (len=24 ttl=255), counter=2
Tunnel0: sending keepalive, 155.1.0.4->155.1.0.5 (len=24 ttl=255), counter=3
Tunnel0: sending keepalive, 155.1.0.4->155.1.0.5 (len=24 ttl=255), counter=4
Tunnel0: sending keepalive, 155.1.0.4->155.1.0.5 (len=24 ttl=255), counter=5
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state to down
BACKUP(Tunnel0): event = primary interface went down
BACKUP(Tunnel0): changed state to "waiting to backup"
BACKUP(Tunnel0): event = timer expired on primary
BACKUP(Tunnel0): secondary interface (Serial0/1/0) made active
BACKUP(Tunnel0): changed state to "backup mode"
%LINK-3-UPDOWN: Interface Serial0/1/0, changed state to up
BACKUP(Serial0/1/0): event = secondary interface came up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0, changed state to
up
BACKUP(Serial0/1/0): event = secondary interface came up
```

Copyright © 2010 Internetwork Expert

Once Serial0/1/0 is installed in the routing table, traffic from R5 to R4's Loopback is rerouted via this link.

```
Rack1R5#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.45.4 12 msec *  12 msec
```

When transport over the Frame Relay link comes back, R5's tunnel interfaces comes back up, and Serial0/1/0 is placed into standby state once again.

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface Serial0/0/0
Rack1R4(config-if)#no shutdown
Rack1R4(config-if)#

Rack1R5#
Tunnel0: sending keepalive, 155.1.0.4->155.1.0.5 (len=24 ttl=255), counter=149
Tunnel0: keepalive received, 155.1.0.4->155.1.0.5 (len=24 ttl=253), resetting
counter
Tunnel0: sending keepalive, 155.1.0.4->155.1.0.5 (len=24 ttl=255), counter=1
Tunnel0: keepalive received, 155.1.0.4->155.1.0.5 (len=24 ttl=253), resetting
counter
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state to up
BACKUP(Tunnel0): event = primary interface came up
BACKUP(Tunnel0): changed state to "waiting to revert"
BACKUP(Tunnel0): event = timer expired on primary
BACKUP(Tunnel0): secondary interface (Serial0/1/0) moved to standby
BACKUP(Tunnel0): changed state to "normal operation"
%LINK-5-CHANGED: Interface Serial0/1/0, changed state to standby mode
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0, changed state to
down
BACKUP(Serial0/1/0): event = secondary interface went down

Rack1R5#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.0.4 32 msec *  28 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

3.13 On-Demand Routing (ODR)

- Remove all previous routing configurations.
- Disable all physical interfaces on R1, R2, R3, R4, and R5 with the exception of their connections to the Frame Relay cloud.
- Ensure that CDP is enabled on the Frame Relay network between these devices.
- Enable ODR on R5.
- Ensure that all devices on this segment have IP reachability to each others' Loopback 0 networks.

Configuration

```
R5:
interface Serial0/0/0
 cdp enable
!
router odr
```

Verification

Note

On-Demand Routing (ODR) uses Cisco Discovery Protocol (CDP) to advertise connected IP routes of a stub router to the hub. The hub router then advertises a default IP route back to the spokes via CDP. Assuming that the hub and spokes are already running CDP, the configuration of this feature requires only one command on the hub, **`router odr`**.

In this design CDP is disabled on the main Serial0/0/0 interface of R5, but it is enabled on the point-to-point subinterfaces of the spokes by default. Therefore the additional command **`cdp enable`** is required on R5.

Copyright © 2010 Internetwork Expert

```
Rack1R1#show cdp neighbors
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
Rack1R5          Ser 0/0.1          169         R S I      2611XM    Ser 0/0/0

Rack1R1#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 155.1.0.5 to network 0.0.0.0

     155.1.0.0/24 is subnetted, 1 subnets
C       155.1.0.0 is directly connected, Serial0/0.1
     150.1.0.0/24 is subnetted, 1 subnets
C       150.1.1.0 is directly connected, Loopback0
o*   0.0.0.0/0 [160/1] via 155.1.0.5, 00:00:14, Serial0/0.1

Rack1R2#show cdp neighbor
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
Rack1R5          Ser 0/0.1          135         R S I      2611XM    Ser 0/0/0

Rack1R2#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 155.1.0.5 to network 0.0.0.0

     155.1.0.0/24 is subnetted, 1 subnets
C       155.1.0.0 is directly connected, Serial0/0.1
     150.1.0.0/24 is subnetted, 1 subnets
C       150.1.2.0 is directly connected, Loopback0
o*   0.0.0.0/0 [160/1] via 155.1.0.5, 00:00:44, Serial0/0.1

Rack1R3#show cdp neighbor
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
Rack1R5          Ser 1/0.1          131         R S I      2611XM    Ser 0/0/0
```

```
Rack1R3#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 155.1.0.5 to network 0.0.0.0

     155.1.0.0/24 is subnetted, 1 subnets
C       155.1.0.0 is directly connected, Serial1/0.1
     150.1.0.0/24 is subnetted, 1 subnets
C       150.1.3.0 is directly connected, Loopback0
o*   0.0.0.0/0 [160/1] via 155.1.0.5, 00:00:49, Serial1/0.1

Rack1R4#show cdp neighbor
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
Rack1R5          Ser 0/0.1          127         R S I      2611XM    Ser 0/0/0

Rack1R4#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 155.1.0.5 to network 0.0.0.0

     155.1.0.0/24 is subnetted, 1 subnets
C       155.1.0.0 is directly connected, Serial0/0/0.1
     150.1.0.0/24 is subnetted, 1 subnets
C       150.1.4.0 is directly connected, Loopback0
o*   0.0.0.0/0 [160/1] via 155.1.0.5, 00:00:52, Serial0/0/0.1

Rack1R5#show cdp neighbor
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
Rack1R1          Ser 0/0/0          161         R S I      2610XM    Ser 0/0.1
Rack1R2          Ser 0/0/0          160         R S I      2610XM    Ser 0/0.1
Rack1R3          Ser 0/0/0          163         R S I      2611XM    Ser 1/0.1
Rack1R4          Ser 0/0/0          171         R S I      2611XM    Ser 0/0/0.1
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     155.1.0.0/24 is subnetted, 1 subnets
C       155.1.0.0 is directly connected, Serial0/0/0
     150.1.0.0/24 is subnetted, 5 subnets
C       150.1.5.0 is directly connected, Loopback0
o       150.1.4.0 [160/1] via 155.1.0.4, 00:00:10, Serial0/0/0
o       150.1.3.0 [160/1] via 155.1.0.3, 00:00:17, Serial0/0/0
o       150.1.2.0 [160/1] via 155.1.0.2, 00:00:21, Serial0/0/0
o       150.1.1.0 [160/1] via 155.1.0.1, 00:00:19, Serial0/0/0


Rack1R1#ping 150.1.2.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/116/124 ms

Rack1R1#ping 150.1.3.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.3.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/116/120 ms

Rack1R1#ping 150.1.4.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 113/117/124 ms

Rack1R1#ping 150.1.5.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.5.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms
```

Copyright © 2010 Internetwork Expert

3.14 OER Components Setup

- Configure R5 as the OER MC and R2, R3, and R5 as the Border Routers.
- Authenticate all OER communications using a password value of "CISCO".
- Set up external and internal OER interfaces on all routers per the network topology presented.
- Enable OER event logging and change the keepalive interval to 5 seconds.

Configuration

Note

Performance Routing (PfR), previously called Optimized Edge Routing (OER), introduces a new concept into typical IP routing. OER enhances the classic destination-based routing concept centered on the shortest path (lowest-cost metric). It takes into account application performance characteristics across different paths. For example, if there are multiple paths between nodes A and B, classic routing will select the one with the shortest "metric". Let us assume that there is a path with high bandwidth but high delay (a satellite link), and a relatively low bandwidth link with low delay (T1). It would be beneficial to route VoIP calls between A and B over the T1 link and route file-transfers across the satellite link.

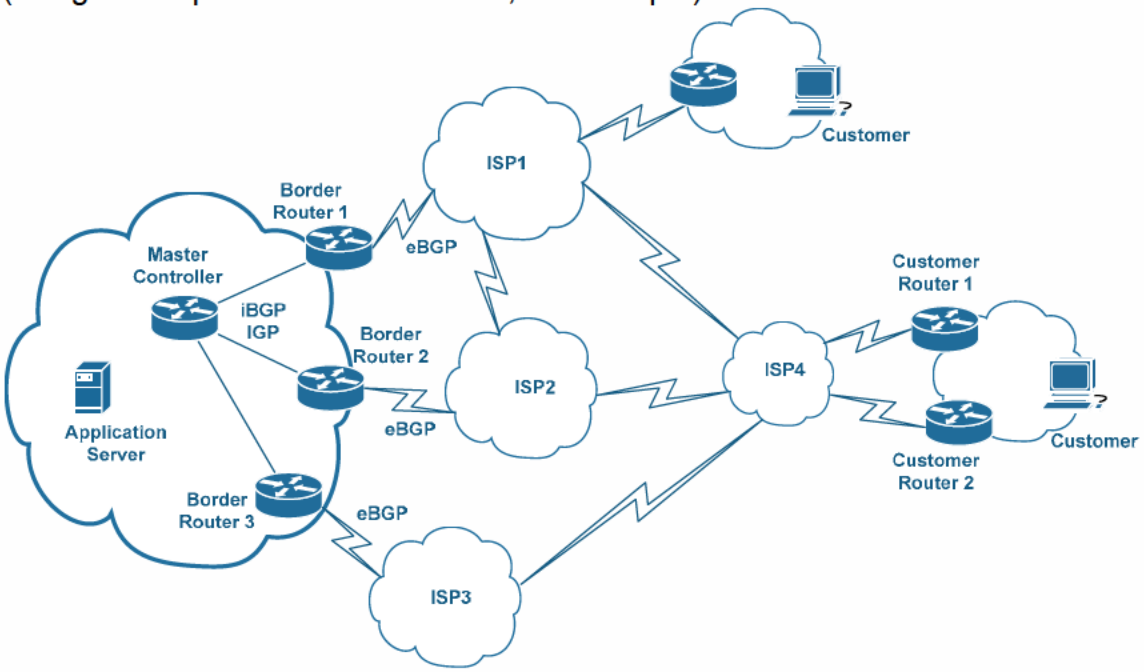
A typical routing protocol may load-balance across multiple links if configured properly. Some of them even support unequal-cost load balancing, loading the links inversely proportional to the link metric. Those methods are inflexible, however, as they do not account for application performance characteristics. The above goal could also be accomplished using the well-known policy-based routing feature, but PBR does not scale well in large networks, and cannot respond to dynamically changing environments. Another solution would be to use MPLS traffic-engineering, but this technology is designed to function within a single enterprise network.

OER allows for dynamic routing toward destinations outside the enterprise network based on the performance characteristics associated with network applications. Here is a short overview of a typical OER deployment:

An enterprise or a service provider hosts a number of applications. For example, a VoIP system, a video broadcasting server, and FTP/HTTP services and so on. The enterprise network features multiple uplinks to services providers and

Copyright © 2010 Internetwork Expert

would like to utilize these uplinks to distribute the traffic routed *outbound* in an optimal manner. Here the "optimal" condition is defined as satisfying the performance policy requirements set by the administrator. OER was initially designed to control and optimize only these *outbound* traffic flows, for example, selecting an optimal exit point for each flow. The ability to control inbound traffic (using eBGP prefix announcements, for example) was added in IOS 12.4T.



OER deployment components include a Master Controller (MC) and Border Routers (BR). The Master Controller is the core of OER functionality – it processes the information (statistics) collected from the Border Routers and uploads policy decisions (prefixes to inject into IGP) to these Border Routers. The BRs connect to the edge of the network and control traffic leaving via the external links. In effect, border routers are the policy enforcement points. Additionally, it is possible to combine the BR and MC roles into a single router.

The OER process runs the following optimization loop continually to ensure the usage of optimal exit points:

`(1) OER Profile phase`. The goal of this phase is discovering the traffic classes. In this stage, OER learns the traffic flows that experience performance issues automatically. Additionally, you may configure the traffic classes manually.

Copyright © 2010 Internetwork Expert

(2) OER Measure phase. In this phase, OER Border Routers measure traffic class performance metrics and report them to the Master Controller. Measurements can be either passive or active. Passive measurements are based on things like Netflow information and interface counters, while active measurements include Border Routers simulating traffic to discover performance characteristics using IP SLA functionality.

(3) OER Apply Policy phase. In this phase, you configure the Master Controller with the set of thresholds that define acceptable performance for traffic classes. The policy may contain such elements as acceptable loss, delay, throughput, link utilization, etc. Based on this policy, the MC determines classes or links that are out of policy (OOP).

(4) OER Control phase. After OER has determined the classes that do not conform to the policy, it controls routing protocol decisions by injecting static or BGP routes, or changing the route metrics. This, in effect, changes the exit point selection in the network.

(5) OER Verify phase. Finally, after OER has attempted the optimization procedure, the Master Controller collects the resulting statistics and verifies that the new exit point selection brings all traffic classes and links to "in policy" states.

Before we discuss all of the phases in depth, let us start with the basic setup of the OER components. When configuring OER, you need to set up both the BRs and the MC. Most of the OER configurations are applied to the MC, and setting up the Border Router is relatively simple. In fact, you only need to specify the Master's address and authentication information (if used). You use the command **oer border** to configure a Border Router, and specify the Master router using the **master** command. Do not forget to specify the local interface in the Border Router for communications with the Master.

You use the command **oer master** to enter OER configuration mode for the MC. You specify every Border Router (even the local one) using the command **border <IP> [key-chain <NAME>]**. You can use a locally configured key-chain for OER session authentication. Under the Border Router configuration sub-mode, you may specify the router-interfaces controlled by OER using the **interface <Name> {internal|external}** command. At least one external interface should be defined for every Border Router, and at least two external interfaces must be configured for a group of BRs. The external interfaces are used for traffic monitoring and forwarding out of the local network.

Copyright © 2010 Internetwork Expert

Our scenario presents AS 200 having three exit points to reach AS 100. In this task, we configure R5 as an MC and BR simultaneously. The other BRs are R3 and R2, and the external interfaces are the links connected to other ASes. All authentications use the key value of CISCO.

R5:

```
key chain OER
 key 1
  key-string CISCO
!
oer master
 keepalive 5
 logging
 !
 border 150.1.3.3 key-chain OER
  interface FastEthernet0/0 external
  interface Serial1/0.1 internal
  interface Serial1/3 internal
  interface Serial1/2 external
 !
 border 150.1.5.5 key-chain OER
  interface Serial0/0/0 internal
  interface FastEthernet0/0 internal
  interface Serial0/1/0 external
!
 border 150.1.2.2 key-chain OER
  interface Serial0/0.1 internal
  interface FastEthernet0/0 external
  interface Serial0/1 internal
!
oer border
 local Loopback0
 master 150.1.5.5 key-chain OER
```

R3 & R2:

```
key chain OER
 key 1
  key-string CISCO
!
oer border
 local Loopback0
 master 150.1.5.5 key-chain OER
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the OER Master status. Make sure all Border peers are “ACTIVE” and “UP” in the output, and there are no AuthFail messages. Note that there are four exits from this AS, defined in three Border Routers.

```
Rack1R5#show oer master
OER state: ENABLED and ACTIVE
  Conn Status: SUCCESS, PORT: 3949
  Version: 2.2
  Number of Border routers: 3
  Number of Exits: 4
  Number of monitored prefixes: 0 (max 5000)
  Max prefixes: total 5000 learn 2500
  Prefix count: total 0, learn 0, cfg 0
  PBR Requirements not met
  Nbar Status: Inactive

Border           Status   UP/DOWN              AuthFail  Version
150.1.5.5        ACTIVE   UP       00:01:12           0  2.2
150.1.2.2        ACTIVE   UP       00:11:08           0  1.0
150.1.3.3        ACTIVE   UP       00:12:06           0  1.0

Global Settings:
  max-range-utilization percent 20 recv 0
  mode route metric bgp local-pref 5000
  mode route metric static tag 5000
  trace probe delay 1000
  logging
  exit holddown time 60 secs, time remaining 0

Default Policy Settings:
  backoff 300 3000 300
  delay relative 50
  holddown 300
  periodic 0
  probe frequency 56
  number of jitter probe packets 100
  mode route observe
  mode monitor both
  mode select-exit good
  loss relative 10
  jitter threshold 20
  mos threshold 3.60 percent 30
  unreachable relative 50
  resolve delay priority 11 variance 20
  resolve range priority 12 variance 0
  resolve utilization priority 13 variance 20
```

Copyright © 2010 Internetwork Expert

```
Learn Settings:
  current state : DISABLED
  time remaining in current state : 0 seconds
  no throughput
  no delay
  no inside bgp
  no protocol
  monitor-period 5
  periodic-interval 120
  aggregation-type prefix-length 24
  prefixes 100
  expire after time 720
```

Copyright © 2010 Internetwork Expert

> **Note**
>
> Check the OER Border Router interfaces. The interfaces are configured in the Master router, and the information is uploaded to the Border Routers.

```
Rack1R2#show oer border
OER BR 150.1.2.2 ACTIVE, MC 150.1.5.5 UP/DOWN: UP 00:20:04,
  Auth Failures: 0
  Conn Status: SUCCESS, PORT: 3949
  Exits
  Fa0/0           EXTERNAL
  Se0/0.1         INTERNAL
  Se0/1           INTERNAL

Rack1R3#show oer border
OER BR 150.1.3.3 ACTIVE, MC 150.1.5.5 UP/DOWN: UP 00:22:03,
  Auth Failures: 0
  Conn Status: SUCCESS, PORT: 3949
  Exits
  Fa0/0           EXTERNAL
  Se1/0.1         INTERNAL
  Se1/2           EXTERNAL
  Se1/3           INTERNAL

Rack1R5#show oer border
OER BR 150.1.5.5 ACTIVE, MC 150.1.5.5 UP/DOWN: UP 00:16:26,
  Auth Failures: 0
  Conn Status: SUCCESS
  OER Netflow Status: ENABLED, PORT: 3949
  Version: 2.2  MC Version: 2.2
  Exits
  Fa0/0           INTERNAL
  Se0/0/0         INTERNAL
  Se0/1/0         EXTERNAL
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

3.15 OER Profile Phase

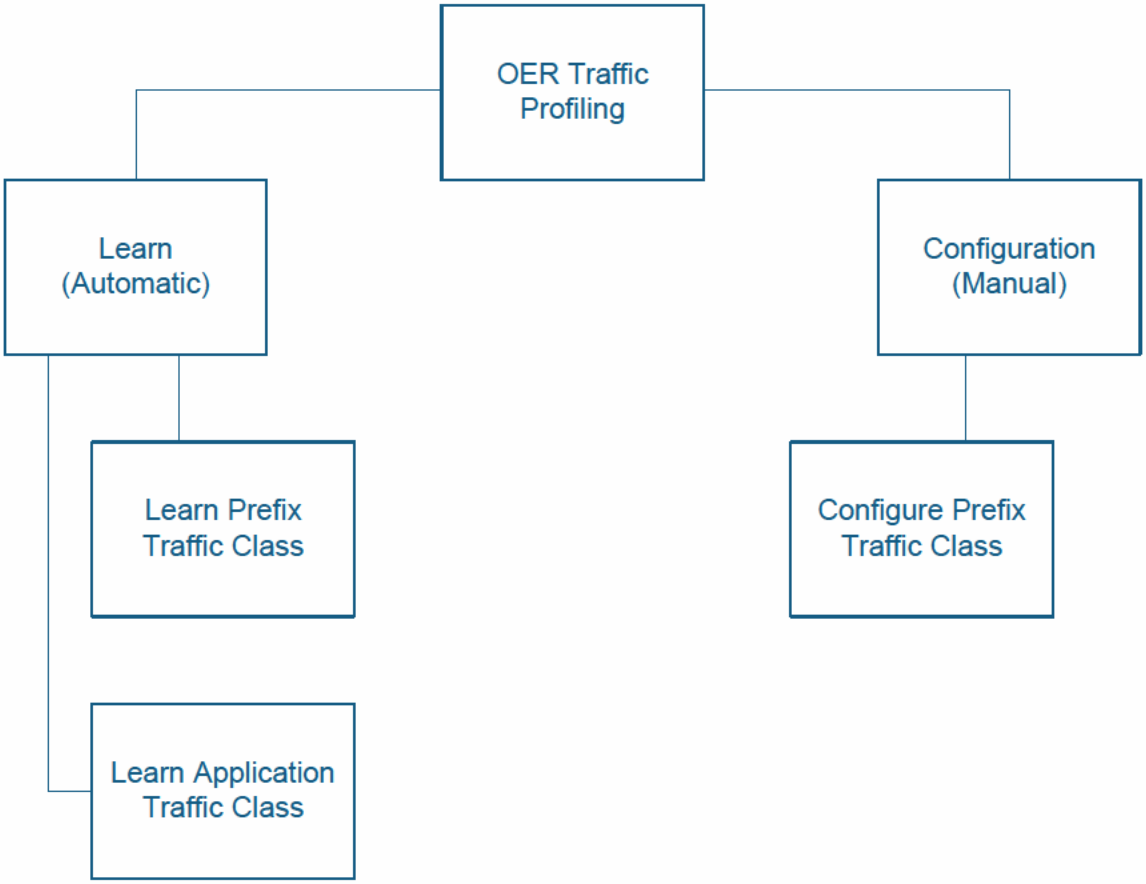
- Configure the OER deployment to learn network prefixes based on maximum throughput and delay.
- The address information should be summarized based on the BGP table contents.
- Run the periodic learning procedure every 5 minutes for a duration of 3 minutes.
- Only store the traffic flows corresponding to ICMP data exchange, WWW transactions, and VoIP bearer traffic (UDP port range 16384-32767).
- Manually configure a prefix traffic class of 112.0.0.0/24.

Copyright © 2010 Internetwork Expert

Configuration

Note

The first OER phase, known as the **`OER Profile`** phase, is about discovering or defining traffic classes. A traffic class could be as simple as a network prefix, or it could be a complex object, such as a network prefix and application port numbers. The routers configured for OER (BRs) can discover the classes automatically (learn), or you can configure the classes manually in the Master Controller. All learned or configured traffic classes are stored in the Monitored Traffic Classes (MTC) table. Let us look at the chart of OER traffic profiling options and describe all of the options in detail.



Copyright © 2010 Internetwork Expert

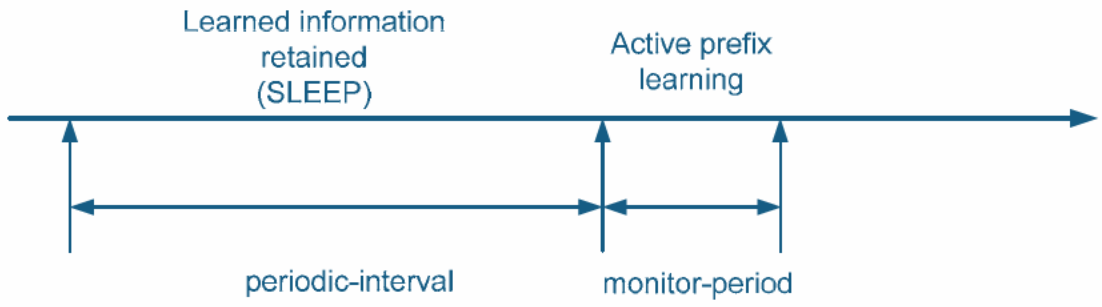
Learning Prefix Traffic Classes

Automatic prefix traffic class learning is based on the NetFlow top-talkers feature. The Master Controller collects NetFlow information from the Border Routers and selects the prefixes with the highest throughput or delay using the information collected. OER classifies prefixes as either external (learned from other ASes) or internal (originated into BGP locally). Use the following commands to enable automatic learning of the prefix traffic classes.

```
oer master
  learn
    throughput
    delay
```

This will instruct all Border Routers to collect the NetFlow information on the *external* prefixes with the highest *outbound* throughput (number of bytes transferred) and delay (RTT time). Since the traffic flows are exchanged between hosts, the number of learned flows could exceed the MTC capacity. This is why OER applies prefix aggregation automatically. By default, external traffic sources are aggregated in /24 subnets and the resulting metrics are summarized or averaged. You may change this mode using the command **aggregation-type {bgp | non-bgp | prefix-length <prefix-mask>}**. When "bgp" mode is configured, the MC will look into the BGP table to find a matching prefix for a flow. This prefix is then used as the aggregate, and the information is stored in the MTC. If the "non-bgp" keyword is specified, the static routes table is used for the same purpose. In the case of "prefix-length <prefix-mask>" option, the MC will automatically aggregate all flow addresses up to the prefix-length specified. This is the default mode, with the prefix-length value of 24.

Prefix learning starts every time interval specified by the command **periodic-interval <minutes>**, with the default value of 120 minutes. The router will start monitoring traffic flows for the duration of time specified by the **monitor-period <minutes>** command, which defaults to 5 minutes.



Copyright © 2010 Internetwork Expert

There is a limit to the number of prefixes learned by the MC and stored in the MTC. By default, 100 top flows are learned, and this value could be changed using the command **prefixes <number>**. The prefixes stored in the MTC are expired if the MC learns no further information about these prefixes. An example would be if there are no traffic flows for these prefixes. You may specify the expiration interval using the command **expire after {session <number> | time <minutes>}.** When you set the expiration timeout using the **session <number>** command, the prefixes are expired after <number> of monitoring sessions have occurred (i.e. after **<number>x(<periodic-interval>+<monitor-period>)).** You may set the expiration period explicitly using the option **time <minutes>.** This defines the amount of time an inactive prefix is stored in the MTC.

Learning Application Traffic Classes

Applications are identified by the protocols and/or the port numbers they use. For example, if your network hosts an HTTP server, then traffic flows to external prefixes sourced off the local TCP port 80 could be associated with the WWW application. In order to configure the Master Controller to learn application traffic classes, add the command **protocol {number | tcp | udp} [port <port-number> | gt <port-number> | lt <port-number> | range <lower-number upper-number>] [dst | src]**. This will filter the prefix information based on the port numbers and aggregate only flows matching the specified port criteria. Use the keywords **dst** or **src** to specify whether the configured ports are source or destination. For example, to specify the internal WWW applications, use the command **protocol tcp port 80 src**. You may configure multiple protocol commands and flows matching any of the commands are stored in the MTC list.

Configuring a Prefix Traffic Class

It is possible to specify a set of prefixes for OER monitoring and control manually, by using the **oer-map** command. The command syntax is similar to the route-map command. **oer-map** is a numbered sequence of entries, where every entry has **match** and **set** clauses. Unlike a route-map, there could be only one match clause per entry and there are no “deny” entries. If you need to include a subnet under OER’s control, but you need to exclude some subnets, use a prefix-list with “deny” entries. For example, if you want to monitor statistics for the prefix 114.0.0.0/8, but exclude the subnet 114.0.1.0/24, use the following construct:

Copyright © 2010 Internetwork Expert

```
ip prefix-list MONITOR deny 114.0.1.0/24
ip prefix-list MONITOR permit 114.0.0.0/8
!
oer-map OER 10
 match ip address prefix-list MONITOR

oer master
 policy-rules OER
```

Notice the use of the command **policy-rules <NAME>** to associate the OER map with the MC configuration. The original purpose of the oer-map is to specify an OER policy – we will discuss the policy configuration in a separate task.

The scenario asks to configure automatic application traffic class learning for WWW, ICMP, and VoIP RTP traffic. We filter the ICMP flows based on the protocol number "1", and the WWW traffic based on the source port of 80. VoIP bearer traffic is identified as sourced from the UDP ports in the range 16384 - 32767. Based on this configuration, the Border Routers will identify traffic leaving over the external interfaces and send the statistics to the Master Controller. Notice that until there are active traffic flows in the network, no information will be sent to the Master Controller.

R5:

```
ip prefix-list NET112 permit 112.0.0.0/24
!
oer-map OER 10
 match ip address prefix-list NET112
!
oer master
 policy-rules OER
!
 learn
  throughput
  delay
  protocol tcp port 80 src
  protocol 1
  protocol udp port range 16384 32767 src
  periodic-interval 5
  monitor-period 3
  aggregation-type bgp
```

Copyright © 2010 Internetwork Expert

Verification

Note

To verify the configuration, we start the following SLA probes to generate synthetic traffic:

1) ICMP pings between R1 and SW2
2) A jitter operation emulating the G.729 codec between R6 and SW2
3) A file transfer using HTTP from SW2 to R4

We are going to see if the Border Routers can actually discover all traffic flows.

R1:

```
no ip sla monitor 1
ip sla monitor 1
 type echo protocol ipIcmpEcho 150.1.8.8 source-interface Loopback0
 timeout 100
 frequency 1
!
ip sla monitor schedule 1 start-time now life forever
```

R4:

```
ip http client source-interface Loopback0

Rack1R4#copy http://cisco:cisco@150.1.8.8/c3560-advipservicesk9-mz.122-
46.SE.bin null:
```

R6:

```
ip sla 2
udp-jitter 150.1.8.8 16384 source-ip 150.1.6.6 codec g729a codec-
numpackets 10 codec-interval 10
exit
!
ip sla schedule 2 life forever start-time now
```

SW2:

```
ip http server
ip http path flash:
ip sla responder
```

Copyright © 2010 Internetwork Expert

Note

After you have set everything up, you can check the prefixes learned by the Border Routers (it may take some time for the MC to process and learn all of the information). Notice that all three external prefixes are listed in the MTC, plus the manually configured prefix.

```
Rack1R5#show oer master prefix learned
OER Prefix Statistics:
 Pas - Passive, Act - Active, S - Short term, L - Long term, Dly - Delay (ms),
 P - Percentage below threshold, Jit - Jitter (ms),
 MOS - Mean Opinion Score
 Los - Packet Loss (packets-per-million), Un - Unreachable (flows-per-million),
 E - Egress, I - Ingress, Bw - Bandwidth (kbps), N - Not applicable
 U - unknown, * - uncontrolled, + - control more specific, @ - active probe all
 # - Prefix monitor mode is Special, & - Blackholed Prefix
 % - Force Next-Hop, ^ - Prefix is denied

Prefix                  State     Time Curr BR         CurrI/F         Protocol
                     PasSDly  PasLDly   PasSUn   PasLUn  PasSLos  PasLLos
                     ActSDly  ActLDly   ActSUn   ActLUn      EBw      IBw
                     ActSJit  ActPMOS  ActSLos  ActLLos
------------------------------------------------------------------------------
-
150.1.1.0/24          DEFAULT*      @56 150.1.5.5      Se0/1/0         U
                             U        U        0        0        0        0
                            87       56        0        0        1        1
                             N        N
150.1.4.0/24          DEFAULT*      @15 150.1.5.5      Se0/1/0         U
                             U        8        0        0        0        0
                            42       63        0        0       67        9
                             N        N
150.1.6.0/24          DEFAULT*      @12 150.1.5.5      Se0/1/0         U
                             U        U        0        0        0        0
                             U        U        0        0        1        0
                             N        N
```

Note

We will discuss the numerous fields in the output above in the task dedicated to the OER measurement phase. So far, make sure the prefixes are just learned by the controller. Now, for automatically learned prefixes you may want to check the criterion used for prefix learning (based on Netflow measurements). Notice that some prefixes may experience both excessive delay and generate maximum throughput.

```
Rack1R5#show oer master prefix learned throughput
OER Prefix Statistics:
 Pas - Passive, Act - Active, S - Short term, L - Long term, Dly - Delay (ms),
 P - Percentage below threshold, Jit - Jitter (ms),
 MOS - Mean Opinion Score
 Los - Packet Loss (packets-per-million), Un - Unreachable (flows-per-million),
 E - Egress, I - Ingress, Bw - Bandwidth (kbps), N - Not applicable
 U - unknown, * - uncontrolled, + - control more specific, @ - active probe all
 # - Prefix monitor mode is Special, & - Blackholed Prefix
 % - Force Next-Hop, ^ - Prefix is denied

Prefix                  State     Time Curr BR         CurrI/F         Protocol
                      PasSDly  PasLDly   PasSUn   PasLUn  PasSLos  PasLLos
                      ActSDly  ActLDly   ActSUn   ActLUn      EBw      IBw
                      ActSJit  ActPMOS  ActSLos  ActLLos
------------------------------------------------------------------------------
-
150.1.4.0/24            DEFAULT*      @41 150.1.5.5       Se0/1/0         U
                               U        U        0        0        0        0
                              82       82        0        0      103        9
                               N        N
150.1.1.0/24            DEFAULT*      @41 150.1.5.5       Se0/1/0         U
                               U        U        0        0        0        0
                              83       80        0        0        1        1
                               N        N
150.1.6.0/24            DEFAULT*      @37 150.1.5.5       Se0/1/0         U
                               U        U        0        0        0        0
                              71       71        0        0        1        1
                               N        N
Rack12R5#
```

Copyright © 2010 Internetwork Expert

Note

You may use the following command to display the Netflow cache entries used to learn the traffic classes.

```
Rack1R3#show oer border passive cache prefix
OER Passive Prefix Cache, State: enabled, 278544 bytes
  2 active, 4094 inactive, 36 added
  764 ager polls, 0 flow alloc failures
  Active flows timeout in 1 minutes
  Inactive flows timeout in 15 seconds
IP Sub Flow Cache, 21640 bytes
  4 active, 1020 inactive, 72 added, 36 added to flow
  0 alloc failures, 0 force free
  1 chunk, 1 chunk added

Prefix              NextHop         Src If          Dst If          Flows
                      Pkts   B/Pk  Active   sDly   #Dly  PktLos  #UnRch
-------------------------------------------------------------------------
150.1.6.0/24        0.0.0.0         Se1/2           Se1/0.1             2
                         11      61     1.0      0      0       0       0
150.1.1.0/24        0.0.0.0         Se1/2           Se1/0.1             1
                         30      64     0.3      0      0       0       0
Rack12R3#
```

3.16 OER Measure Phase

- Use the combined monitoring mode and the configured manual TCP connect probes for the Loopback0 IP addresses of R1, R4, and R6.
- Configure the Master Controller to use active monitoring only for the prefix 150.X.4.0/24.
- Configure the Border Routers so that any interface utilization changes are detected as quickly as possible.

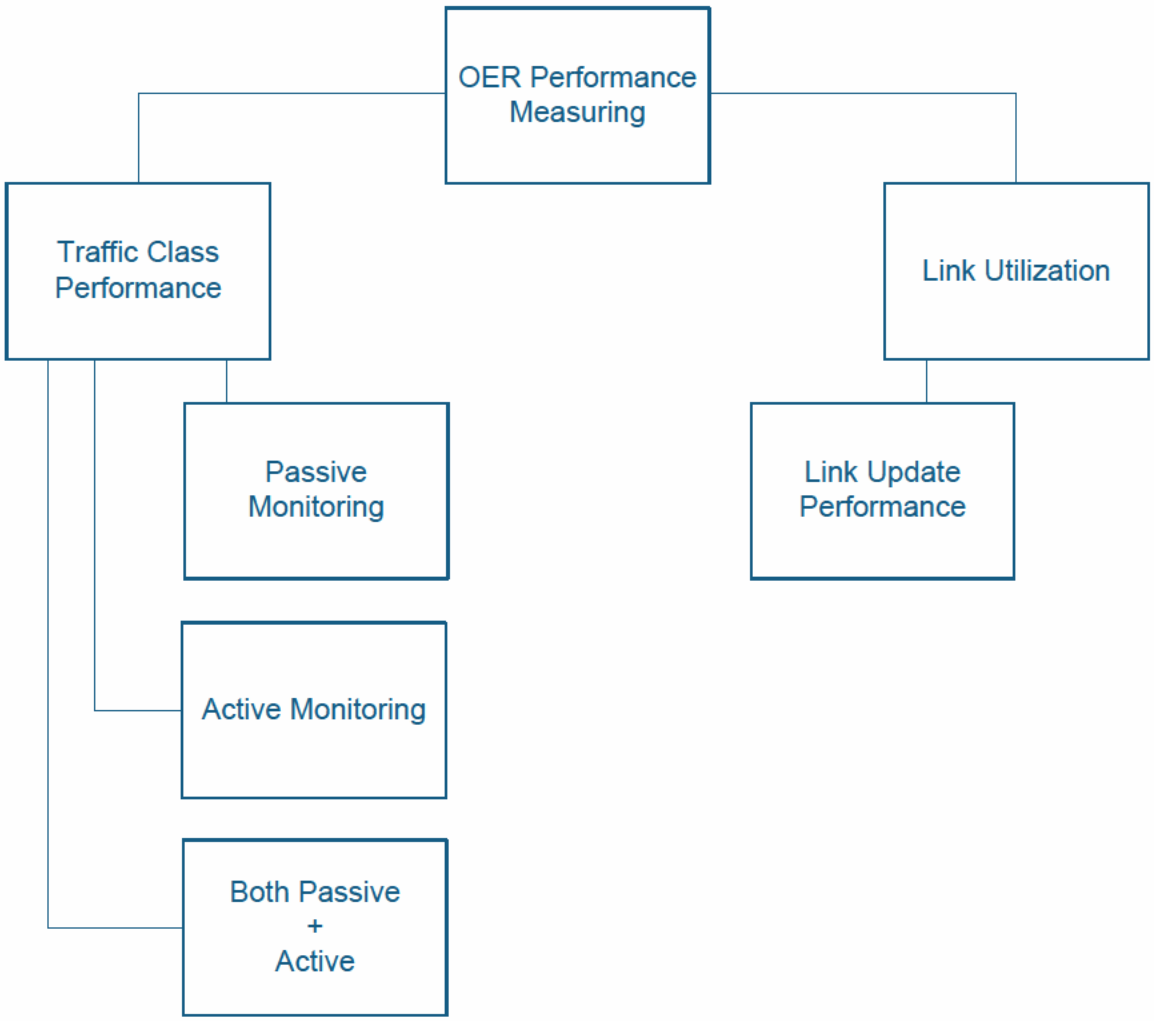
Configuration

Note

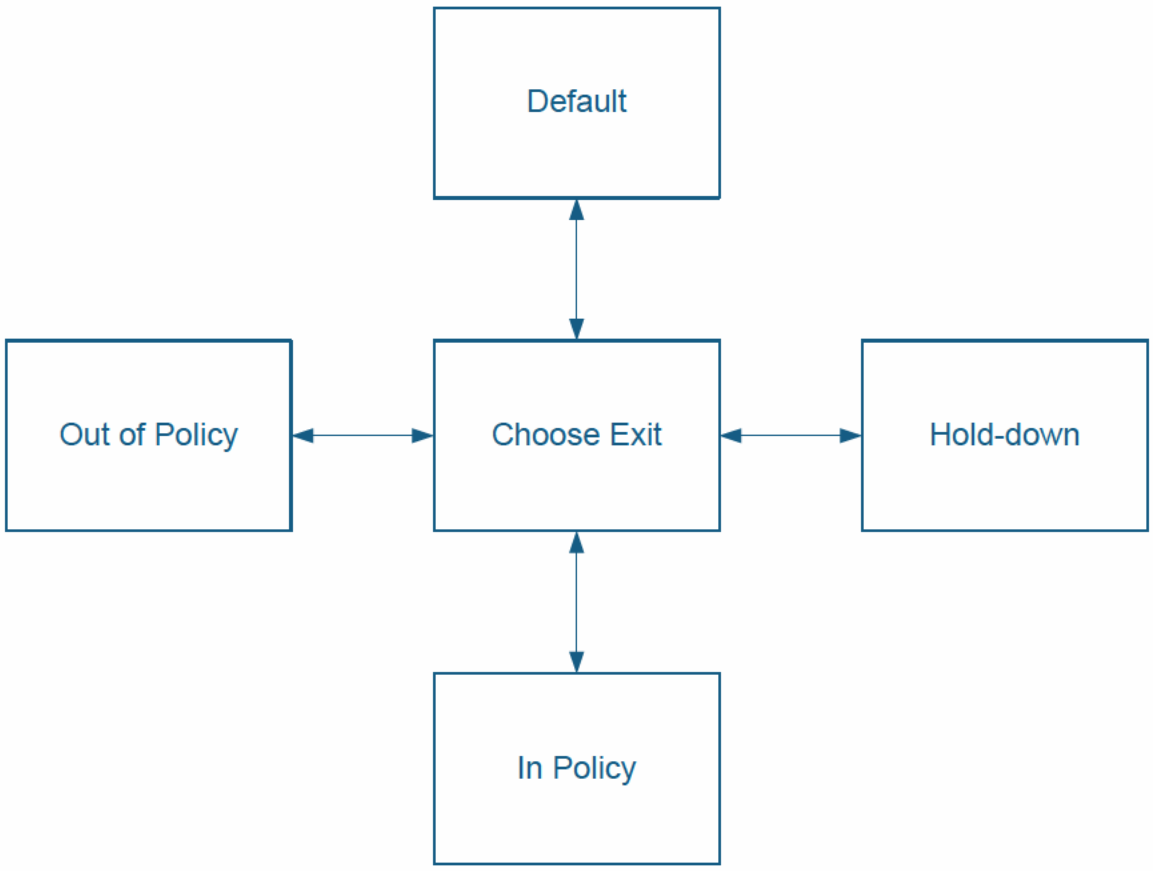
`OER Measure Phase`

This phase is the second step in the OER performance optimization loop. This step starts after the traffic classes have been learned automatically or configured by the administrator. During this phase, Border Routers passively monitor traffic flows or actively probe selected targets. In addition to monitoring traffic classes, Border Routers also monitor external link performance, based on link load and error rates. The collected information is reported back to the MC, where the information is associated with the traffic classes stored in the MTC. Here is an outline of the measurement process:

Copyright © 2010 Internetwork Expert



OER measures the performance of both traffic classes and external links, but before monitoring a traffic class or a link, OER checks its current state. OER transitions traffic classes or links through a series of states per the following diagram:



The current state selected for a traffic class or link is called the policy decision point (PDP). In some states, OER does not initiate monitoring. Here is the list of states:

`Default` – this state means the traffic class is not under OER control. Traffic classes are placed in the default state when they are initially added to the MTC. A traffic class will transition into and out of the default state depending on the measurement results and policy configuration.

`Choose Exit` – This is a transient state, where OER compares performance characteristics with the traffic policy and selects an exit point (BR/external interface) for this class. OER will try to keep a traffic class flowing through its current exit, but performance measurements, timers, and policy configurations can cause the MC to start looking for another exit point. The traffic class remains in this state until it is moved to the new exit.

Copyright © 2010 Internetwork Expert

Holddown - A traffic class is placed in this state when the MC requests a Border Router to monitor the traffic class using active probes. Measurements are collected for the selected traffic class until the holddown timer expires.

In-Policy - After performance metrics are compared against the policy settings and an exit point selection is made, the traffic class enters an in-policy state. When a traffic class is in this state, the traffic is forwarded through an exit that satisfies the default or user-defined policy settings. The MC keeps monitoring the traffic class, but no action is taken until the periodic timer expires, or an out-of-policy event is generated. In this case, the traffic class transitions back to the "Choose Exit" state.

Out-of-Policy (OOP) - A traffic class enters this state when there are no exit points conforming to the performance policy settings for this class. While the traffic class is in this state, the backoff timer controls exiting from this state. Each time the traffic class enters this state, the amount of time the traffic class spends in this state increases exponentially. The backoff timer is reset for a traffic class when the traffic class enters the "in-policy" state. If all exit links are out-of-policy, the Master Controller may select the best available exit.

As mentioned, OER uses three methods of performance measurement:

Passive monitoring

BRs measure the performance metrics of a traffic class while the corresponding traffic is flowing through the device. Passive monitoring uses NetFlow functionality and is enabled by default. Passive monitoring can also be enabled explicitly using the **mode monitor passive** command for a traffic class. NetFlow support is enabled by default on the Border Routers when passive monitoring is enabled. Passive monitoring uses only existing traffic - additional traffic is not generated. Border Routers collect and report passive monitoring statistics to the Master Controller approximately once per minute. OER uses passive monitoring to measure the following metrics for all traffic classes:

Delay - OER measures the average delay of TCP flows for a given prefix. Delay is measured as the round-trip response time (RTT) between the transmission of a TCP segment and receipt of the TCP ACK packet. This metric is measured only for the TCP flows.

Copyright © 2010 Internetwork Expert

Packet loss - OER measures packet loss by tracking TCP sequence numbers for each TCP flow. OER estimates packet loss by storing the highest TCP sequence number and comparing it to the subsequent packets. If a subsequent packet is received with a lower sequence number, meaning retransmission, OER increments the packet loss counter. Packet loss is measured in packets per million and is only measured for TCP flows.

Reachability - OER measures reachability by tracking TCP SYN messages that have been sent repeatedly without receiving a TCP ACK packet. Again, this metric applies only to TCP flows.

Throughput - OER measures throughput by measuring the total number of bytes and packets for each traffic class for a given interval of time. This metric is monitored for all types of flows, not just TCP-based.

Passive monitoring is enabled by default for all automatically learned prefixes. You may enforce the default passive mode monitoring using the **oer master** configuration mode command **mode passive** or the **oer-map** command **set mode passive**. The last command allows you to selectively enforce a particular monitoring mode for a manually configured prefix.

Active monitoring

In this mode, the Master Controller instructs the Border Routers to create a stream of synthetic traffic replicating a traffic class behavior as closely as possible and measuring the performance metrics of the resulting flows. The results of the performance metrics of the synthetic traffic are applied to the corresponding traffic class in the MTC list. Active monitoring uses the IP Service Level Agreements (IP SLAs) functionality to implement active traffic probes. Active monitoring could measure metrics such as delay or reachability for any types of flows, not just TCP flows, as with passive monitoring.

In active monitoring mode, the OER activates the probes on all the Border Routers to find the best performance path for the specific traffic class. The active probes for that traffic class are not activated again unless the traffic class goes OOP (e.g. due to a link utilization OOP event). In Cisco IOS Release 12.4(4)T, and earlier releases, the frequency of an active probe used by OER is fixed to 60 seconds.

OER uses active monitoring to measure the following metrics for all the traffic classes (not just TCP-based traffic classes!):

Copyright © 2010 Internetwork Expert

Delay - OER measures the average delay of TCP, UDP, and ICMP flows for a given prefix. Delay is the measurement of the round-trip response time (RTT) between the transmission of a TCP synchronization message and receipt of the TCP acknowledgement.

Reachability - OER measures reachability by tracking TCP synchronization messages that have been sent repeatedly without receiving a TCP acknowledgement.

Jitter - Jitter means inter-packet delay variance. OER measures jitter by sending multiple packets to a target address and a specified target port number, and measuring the delay interval between packets arriving at the destination.

MOS - Mean Opinion Score (MOS) is a standards-based method of measuring voice quality. MOS scores range between 1, representing the worst voice quality, and 5, representing the best voice quality.

Use the oer master mode command **mode active** to switch the monitoring mode to active by default. You can change the mode for a manually configured prefix using an oer-map and the clause **set mode monitoring active.** This allows for per traffic class granularity when setting the monitoring modes.

Types of Active Probes

As mentioned, active probes are implemented using IP SLA functionality in IOS routers. The following types of active probes can be configured for use by OER.

ICMP Echo - A ping is sent to the target address. OER uses ICMP Echo probes, by default, when an active probe is automatically created (described later).

Jitter - A jitter probe is sent to the target address. A target port number must be specified. A remote responder must be enabled on the target device, regardless of the configured port number. Jitter probe support was introduced in Cisco IOS Release 12.4(6)T.

TCP Connection - A TCP connection probe is sent to the target address and port number. A remote responder must be listening on the port configured in the target settings, or IP SLA responder must be enabled in the remote IOS router.

UDP Echo - A UDP echo probe is sent to the target address and port pair. A remote responder must be enabled on the target IOS device.

Copyright © 2010 Internetwork Expert

Creation of Active Probes

To create an active probe for a traffic class, a probe type has to be *discovered*, and a probe target *assigned* to the traffic class. To discover a probe type, OER uses one of the following methods:

Learned probe - Active probes are automatically generated when a traffic class is learned using the NetFlow learn mechanism (highest delay or throughput flows). Five targets are learned for each traffic class and, by default, the active probe type is set as an ICMP echo probe.

Configured probe - Active probes can be configured on the Master Controller by specifying the probe type, target address, and port if needed. These probes are assigned to traffic classes based on the longest match (see below). Additionally, manually configured traffic classes can use any of the IP SLA active probes bound to them using an oer-map. Use the **oer master** mode subcommand **active-probe <Parameters>** to configure a manual probe (see below for information regarding the probe to traffic class association process).

When creating an active probe through an external interface for a specified target, the target should be *reachable* through the external interface. To test the reachability of the specified target, OER performs a route lookup in the BGP and static routing tables for the specified target and external interface. If the target is reachable, OER uses one of the following methods to assign a probe target for a traffic class:

Longest match - By default, OER assigns a probe target to the traffic class with the longest matching prefix in the MTC list. This is the default probe assignment. This procedure works for both learned and configured probes, i.e. the configured probe will not be activated until any of the BRs report a matching prefix.

Forced assignment - An IP SLA probe can be configured using an OER map (**set active-probe** oer-map clause) and the results of the measurements are assigned to the specific traffic class associated with this OER map entry. This specific assignment of an active probe is called a forced target probe assignment. This feature is not available in IOS 12.4 code.

Copyright © 2010 Internetwork Expert

Here is a sample configuration of active probes in the Master Controller:

```
oer master
 active-probe echo 10.1.1.1
 active-probe tcp-conn 10.1.1.2 target-port 80
 active-probe udp-echo 150.1.6.6 target-port 123
```

Keep in mind that in order for the TCP/UDP probes to work, you need to make sure the remote host is configured to respond on this port. For example, you may need to configure an IOS router using the command **ip sla monitor responder type tcpConnect ipaddress <IP> port <number>** to start the SLA responder on this port.

Since active probes are assigned to the Border Routers, the probe traffic is transmitted over an external interface using the interface IP address to source the packets. Notice that all Border Routers will run the probe, even through external interfaces that might not be an optimal exit point for the traffic class. The reason for this is the collection of performance metrics on all alternate exit paths for the particular traffic class.

Combined Monitoring

This includes both active and passive monitoring; both are used in order to generate a more complete picture of traffic flows within the network. This is the monitoring mode implemented by default. All traffic classes are passively monitored using integrated NetFlow functionality, and out-of-policy traffic classes are actively monitored using IP SLA functionality. This allows for selection of the new best exit point without collecting any information passively, as a given traffic class uses just one exit point.

An example scenario is when you want to learn traffic classes and then monitor them passively, but you also want to determine the alternate path performance metrics in order to control the traffic classes. The alternate metrics in the absence of the actual traffic can be measured using the active probes. OER automates this process by learning traffic classes at five targets (learned probes) and probing through all the alternate paths using active probes when detecting an OOP event.

Copyright © 2010 Internetwork Expert

OER Link Utilization Measurement

Link Utilization Threshold

After an external interface is configured for a Border Router, OER automatically monitors the utilization of the external link, which typically links to an ISP. Every 20 seconds, by default, the Border Router reports the link utilization to the Master Controller. If the exit link utilization is above the default threshold of 75 percent, the exit link is in an OOP state and OER starts the monitoring process to find an alternative link for the traffic classes using this link. The OER polls the interface load counters that are calculated by the Border Router's IOS. In order to increase the reaction time to loading changes, you may want to set the **load-interval** value to the minimum of 30 seconds on all external interfaces.

Although configuring link thresholds is a part of the Apply Policy phase, we are going to discuss it here. The link utilization threshold can be manually configured either as an absolute value in kilobits per second (kbps), or as a percentage using the external interface configuration command **max-xmit-utilization {absolute kbps | percentage value}**. Look at the configuration sample that sets the transmit bandwidth usage for an interface:

```
oer master
  border 1.1.1.1
    interface Serial 0/0 external
       max-xmit-utilization percentage value 50
```

Link Utilization Range

You may also configure OER to calculate the range of utilization over all the links, for example, to detect if the exit links are not being used equally. By default, OER automatically monitors the utilization of external links on a Border Router every 20 seconds, and the Border Router reports the utilization to the Master Controller. If the utilization range between all the exit links exceeds 20%, the Master Controller tries to equalize the traffic load by moving some traffic classes to another exit link. The maximum utilization range is configured as a percentage using the **oer master** mode command **max-range-utilization percent <maximum>**.

The scenario changes the external link's load-interval and adds three active probes manually, using the TCP connection operation. Since the default port of 23 is used for the TCP operation, there is no need to configure IP SLA responder in the target systems. Here, for the prefix 150.X.4.0/24, the monitoring mode is changed to active.

Copyright © 2010 Internetwork Expert

```
R5:
ip prefix-list R4_LO0 permit 150.1.4.0/24
!
oer-map OER 20
 match ip address prefix-list R4_LO0
 set mode monitor active
!
oer master
 active-probe tcp-conn 150.1.1.1 target-port 23
 active-probe tcp-conn 150.1.4.4 target-port 23
 active-probe tcp-conn 150.1.6.6 target-port 23
!
interface Serial 0/1/0
 load-interval 30

R3:
interface FastEthernet 0/0
 load-interval 30
!
interface Serial 1/2
 load-interval 30

R2:
interface FastEthernet 0/0
 load-interval 30
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Let us configure R1, R4, and R6 for the downloading of IOS images via HTTP from R2 and SW2 again. We are going to use the Loopback0 interfaces on R1, R4, and R6 to source the connections.

R1:

```
ip http client source-interface Loopback0

copy http://cisco:cisco@150.1.2.2/c2600-adventerprisek9-mz.124-10.bin
nulll:
```

R4:

```
ip http client source-interface Loopback0

Rack1R4#copy http://cisco:cisco@150.1.8.8/c3560-advipservicesk9-mz.122-
46.SE.bin null:
```

R6:

```
ip http client source-interface Loopback0

copy http://cisco:cisco@150.1.8.8/c3560-advipservicesk9-mz.122-
46.SE.bin null:
```

R2 & SW2:

```
ip http server
ip http path flash:
```

Copyright © 2010 Internetwork Expert

Note

Check once again the prefixes learned and configured in the Master Controller. Now for the description of the fields in the output:

PasSDly – passively measured delay, short-range (5 minutes interval)
ActSDly – actively measured delay, short-range
PasLDly – passively measured delay, long-range (60 minute interval)
PasSUn/PasLUn – passively measured short and long range unreachable metric
ActSUn/ActLUn – the same unreachable, just measured actively
PasSLos/PasSLos – short and long range loss, measured passively
EBw/IBw – egress and ingress bandwidth usage for this class in kbps
State – any of the states from the traffic state diagram
Time – the amount of time spent in the state
Curr BR – current Border Router selected for this class
Curr I/F – current exit interface selected for this class
Protocol – protocol used to influence routing for this traffic class

Please note that the show output below is a "snap shot" of the status of the performance routing process. This outcome is dynamic due to the different delays between interfaces, the monitor-period and periodic-interval, and the passage of time.

```
Rack1R5#show oer master prefix
OER Prefix Statistics:
 Pas - Passive, Act - Active, S - Short term, L - Long term, Dly - Delay (ms),
 Los - Packet Loss (packets-per-million), Un - Unreachable (flows-per-million),
 E - Egress, I - Ingress, Bw - Bandwidth (kbps), N - Not applicable
 U - unknown, * - uncontrolled, + - control more specific, @ - active probe all

Prefix                  State      Time Curr BR         CurrI/F         Protocol
                     PasSDly   PasLDly   PasSUn   PasLUn  PasSLos  PasLLos
                     ActSDly   ActLDly   ActSUn   ActLUn      EBw      IBw
------------------------------------------------------------------------------
112.0.0.0/24         DEFAULT*       57 U                U
150.1.1.0/24         INPOLICY      @45 150.1.3.3        Fa0/0           BGP
                           16       16         0        0     1669     2277
                           22       16         0        0       24        3
150.1.4.0/24         HOLDDOWN      310 150.1.5.5        Se0/1/0         BGP
                            N        N         N        N        N        N
                            U        U         0        0        N        N
150.1.6.0/24         INPOLICY      @33 150.1.3.3        Fa0/0           BGP
                          316      316         0        0     1978     2534
                           10       10         0        0       15        5
```

Note

Now check the probes existing in the OER domain. Notice that the "echo" probes were auto-generated for the traffic classes learned by the MC. The TCP connect probes that we configured manually have been assigned to the respective classes based on the longest-match criteria.

Finally, the last part of the output shows the currently active probes and the Border Routers running them.

```
Rack1R5#show oer master active-probes
        OER Master Controller active-probes
Border   = Border Router running this Probe
State    = Un/Assigned to a Prefix
Prefix   = Probe is assigned to this Prefix
Type     = Probe Type
Target   = Target Address
TPort    = Target Port
How      = Was the probe Learned or Configured
N - Not applicable

The following Probes exist:

State      Prefix            Type     Target          TPort   How     Codec
Assigned   150.1.6.0/24      tcp-conn 150.1.6.6          23  Cfgd        N
Assigned   150.1.4.0/24      tcp-conn 150.1.4.4          23  Cfgd        N
Assigned   150.1.1.0/24      tcp-conn 150.1.1.1          23  Cfgd        N
Assigned   150.1.6.0/24      echo     150.1.6.6           N  Lrnd        N
Assigned   150.1.1.0/24      echo     150.1.1.1           N  Lrnd        N
Assigned   150.1.4.0/24      echo     150.1.4.4           N  Lrnd        N

The following Probes are running:

Border          State    Prefix            Type     Target          TPort
150.1.3.3       ACTIVE   150.1.1.0/24      echo     150.1.1.1           N
150.1.3.3       ACTIVE   150.1.1.0/24      echo     150.1.1.1           N
150.1.5.5       ACTIVE   150.1.1.0/24      echo     150.1.1.1           N
150.1.3.3       ACTIVE   150.1.6.0/24      echo     150.1.6.6           N
150.1.3.3       ACTIVE   150.1.6.0/24      echo     150.1.6.6           N
150.1.3.3       ACTIVE   150.1.4.0/24      echo     150.1.4.4           N
150.1.3.3       ACTIVE   150.1.4.0/24      echo     150.1.4.4           N
150.1.5.5       ACTIVE   150.1.6.0/24      echo     150.1.6.6           N
150.1.5.5       ACTIVE   150.1.4.0/24      echo     150.1.4.4           N
```

Copyright © 2010 Internetwork Expert

3.17 OER Apply Policy Phase

- Configure the network's performance policy to look for the best exit for the traffic classes.
- OER should generate an OOP event once any link utilization exceeds 80% or when the range between link utilization is above 10%.
- Any acceptable exit point should have a relative loss of no higher than 3% and delay no higher than 200ms.
- Best exit should be selected based on the minimum external utilization, and then based on the lowest delay for the prefix class.
- Reverse the above policy logic for the prefix 150.X.6.0/24 and change the acceptable delay to 100ms.
- All exit points with performance metrics within 15% of the best exit should be considered equivalent.
- Ensure periodic re-optimization occurs every minimal time interval.
- After an OOP event, the traffic class should retain the new exit for no less than 10 minutes.

Configuration

Note

The OER apply policy phase is the third step in the OER performance optimization loop. After the profile phase has identified traffic classes and the measure phase has determined their performance characteristics, OER compares the metrics collected against the default or configured thresholds to see if the traffic classes meet the performance requirements. If the performance metrics do not conform to the policy, a decision is made by OER to move the traffic class into another state. This can cause OER to look for a better exit point or start additional probing.

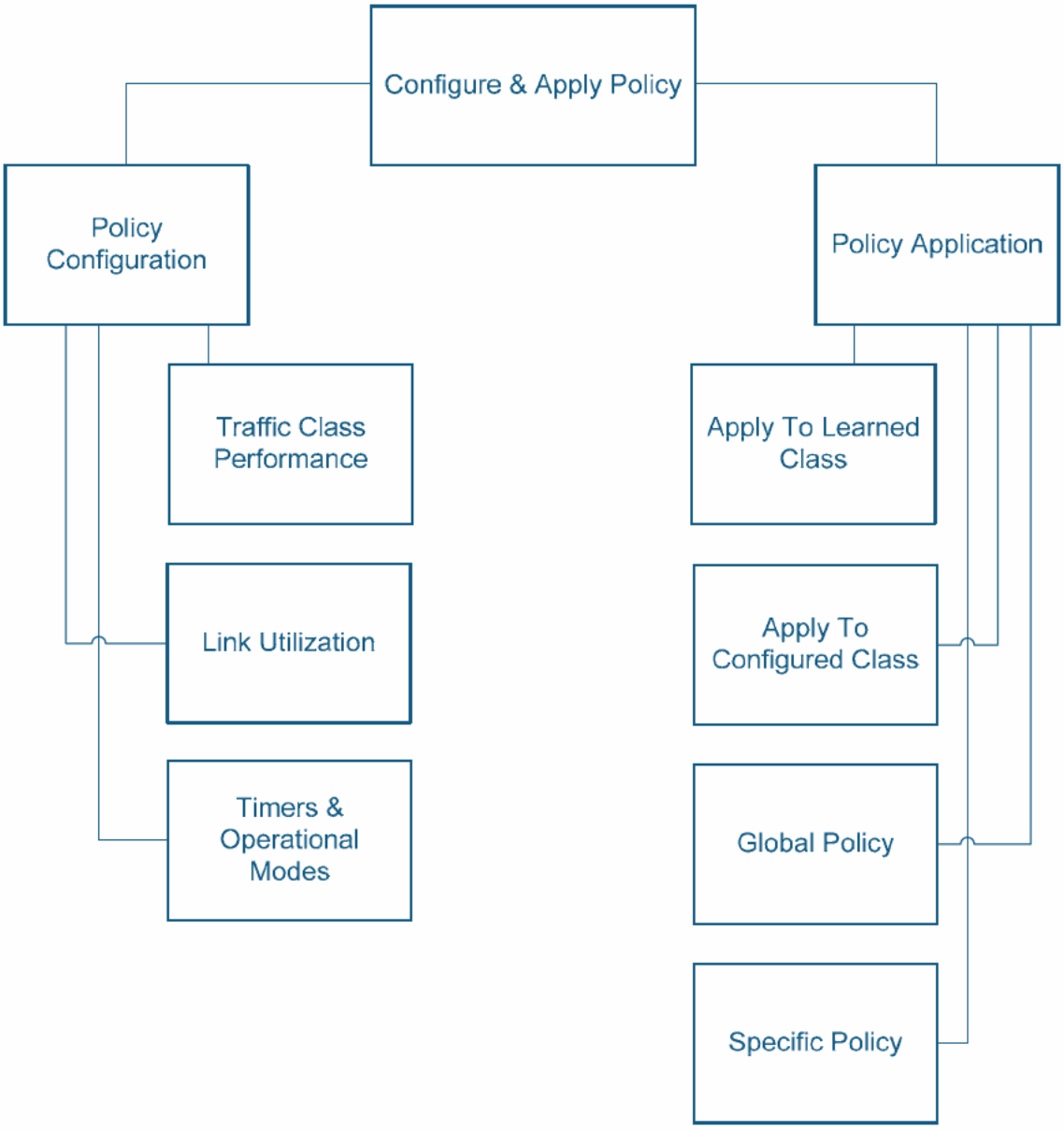
The OER performance policy is a set of rules defined via the oer-map command. Every rule (oer-map entry) has the following attributes:

a) a scope, i.e. the traffic class
b) An action, e.g. insert a static route on the Border Router to change the exit point
c) A condition that triggers the rule: e.g. the acceptable delay thresholds or MOS score range

Copyright © 2010 Internetwork Expert

For example, a policy may require that the MOS for a voice traffic class (the scope) should not fall below the value of 4 (the triggering event). If this happens, OER should look for another exit point for the particular traffic class (the action). The actual policy action may not execute until OER is configured to control the traffic in the OER control phase (next step). By default, OER runs in an observe mode during the profile, measure, and apply policy phases and no changes are being made to the network.

During this phase you can configure and apply policies. Different types of OER policies can be configured, as outlined in the diagram below:



Copyright © 2010 Internetwork Expert

After an OER policy is configured, the policy can be applied to learned traffic classes or configured traffic classes. OER policies can be applied globally - to all the traffic classes - or to just a specific set of traffic classes.

Traffic Class Performance

An OER traffic class performance policy is a set of rules that specify acceptable performance characteristics for traffic classes. The classes can be network addresses (prefixes) or applications defined via protocols, port numbers, or DSCP values. The list of the performance characteristics controlled by OER include the following:

Reachability - OER extends the concept of reachability by automatically verifying that the destination can be reached through the particular path using the active probe or passive packet exchange monitoring. Reachability is specified as the relative percentage or the absolute number of unreachable hosts (based on flows per million (fpm)) that OER will permit for a single traffic class. If the measured value exceeds the configured threshold, the traffic class is declared out of policy (OOP) and OER starts looking for an alternate exit. Use the command **[set] unreachable {relative <average%> | threshold <absolute-maximum-fpm>}** under the oer master configuration mode (global) or oer-map configuration mode (class-specific).

The relative unreachable host percentage is based on a comparison of *short-term* and *long-term* measurements. The short-term measurement reflects the percentage of hosts that are unreachable within the last measured 5-minute period. The long-term measurement reflects the percentage of unreachable hosts within the last 60-minute period. The following formula is used to calculate this value:

```
Relative-Average% = ((short-term percentage - long-term
percentage) / long-term percentage) * 100%
```

This formula allows to adaptively track the number of unreachable hosts for the traffic class – the “quick” value is normalized based on the “slow” base value. This allows for comparing the short-term traffic class behavior with its own behavior on a longer scale and avoiding “whiplash” false detections. For example, if there were 10 unreachable hosts in the last 60 minutes and 20 in the last 5 minutes, the result will be a 100% jump in the unreachable hosts.

Copyright © 2010 Internetwork Expert

Delay - defined as the time elapsed between when the packet was sent from the source, and when it arrived at the destination. Delay is measured in passive mode for the TCP flows and in active mode for any traffic class. You may set the relative and absolute policy thresholds using the oer master mode or the oer-map command **[set] delay {relative <average>%|threshold <absolute-msecs>}**. These settings reflect the global and the class-specific policy settings, respectively. The formula to calculate the relative deviation in percent is the same as the above formula.

Packet Loss - measured in packets per million and monitored for TCP traffic classes in passive mode. When using the active monitored mode, this metric could be measure for any traffic class. The commands for the oer global mode and the oer-map is **[set] delay {relative <average>%|threshold <absolute-packets-per-million>}**.

Jitter and **Mean Opinion Score (MOS)** are two advanced metrics introduced in the recent IOS releases. For the sake of simplicity, we will not cover those here, and rather provide a separate lab on their usage. Keep in mind, that in Cisco IOS Release 12.4(4)T and prior releases, only reachability, delay, and loss performance characteristics can be used.

OER Link Policies

OER link policies apply to the external links of the OER Border Router and define the desired performance characteristics of the links, considering the performance of the link as a whole. The following performance characteristics are controlled by link policies (collected every 20 seconds during the measurement phase).

Link Utilization (or Traffic Load) - OER Border Routers report the current outbound load of their external interfaces taking the rate value from the interface counters. The interface load is calculated by the Border Router's IOS using the averaging interval specified by the interface command **load-interval <seconds>**, with a minimum value of 30 seconds and a default of 300 seconds. It is recommended to set the load-interval to 30 seconds to allow the Master Controller to respond to interface load changes rapidly.

If the link utilization exceeds the threshold, the exit link is declared to be in an OOP state and OER starts the monitoring process to find an alternative link for the traffic classes affected by this event.

Copyright © 2010 Internetwork Expert

The link utilization threshold can be manually configured either as an absolute value in kilobits per second (kbps), or as a percentage using the Border Router external interface command **max-xmit-utilization percentage <percents%>**. For absolute values use the command **max-xmit-utilization absolute <Kbps>**. In the former case, the percentage is taken off the total interface bandwidth configured via the interface **bandwidth** command and the default threshold is 75%. You may see the sample relative utilization configuration in the previous scenario, as part of the measurement configuration.

Range - the range policy maintains all links within a certain utilization range, relative to each other, in order to ensure that the traffic load is distributed evenly between the exit points. The OER range functionality allows you to configure OER to maintain the traffic utilization on a set of links within a certain percentage range of each other. If the difference between the links becomes too great, OER will attempt to bring the link back to an in-policy state by distributing traffic classes among the available links. The default difference between the link load is 20%, but this value can be changed globally in oer master mode using the command **max-range-utilization percent <percent>**.

Cost - cost-based optimization allows you to configure policies based on the monetary cost of each exit link in your network. To implement OER cost-based optimization, the OER Master Controller sends traffic over exit links that provide the most cost-effective bandwidth utilization, while still maintaining the desired performance characteristics.

Cost-based optimization supports two billing models: fixed-rate billing or tier-based billing.

Fixed-rate - reflects the flat payment rate for network access regardless of bandwidth usage. If only fixed-rate billing is configured on the exit links, all exits are considered equal with regard to cost-optimization and other parameters such as delay, loss, and link utilization are used to determine if the traffic class or exit link is in-policy. If the exit links are configured with tiered and fixed policies, then exit links with fixed policies have the highest priority with regard to cost optimization. Only if all the fixed exit links are at maximum utilization, the tiered exit links will be used.

Tier-based - reflects the ISP billing at a tiered rate based on the percentage of exit link utilization. Each cost tier is configured separately as a percentage of bandwidth utilization with an associated monetary cost. We are not going to discuss this cost-optimization model here, due to its complexity.

Copyright © 2010 Internetwork Expert

Based on the facts above, we will not be concerned too much with cost optimization policies for now, as they only make sense when using the tier-based billing mode.

OER Policy Operational Modes and Timers

So far, we have discussed the policies that apply to traffic classes or links and that relate to performance metrics. Now we will examine the timers used by OER and different OER operational modes.

OER Timers

Three types of timers can be configured as OER policy operational parameters:

Backoff Timer - after a traffic class enters the out-of-policy state, the Master Controller will wait some time before attempting to find another exit for this class. The special backoff timer defines the time-interval that the MC waits, and this interval gets incremented every time the Master Controller cannot find another optimal exit. The purpose of such behavior is allowing the Master Controller to have more time to make a decision after every unsuccessful attempt. The command to define the backoff timer is entered in the oer master mode: **backoff <min-inteval-seconds> <max-interval-seconds> <step-seconds>**. When a class enters the OOP state, the Master Controller waits for the initial minimum interval and attempts to find another exit. If the exit cannot be found, the minimum wait timer is incremented by the <step> value and the counting down starts again. The minimum interval could be incremented until it reaches the maximum interval. The default values for all three parameters are 300, 3000, and 300 respectively.

Holddown Timer - sets the minimum period of time that a new exit point selected after an OOP *must* be used before an alternate exit can be selected. This procedure prevents the traffic class entry from flapping because of rapid state changes. A traffic class entry will remain in a holddown state for the time period defined via the Master Controller configuration mode command **holddown <300-65535 sec>**. This defaults to 300 seconds. When the holddown timer expires, OER will select the best exit based on performance and policy configuration. The only event that allows for early breaking of the holddown timer is when the prefix becomes unreachable via the currently selected exit point.

Copyright © 2010 Internetwork Expert

Together, the backoff and the holddown timer determine the important event dampening procedure used by OER. Dampening is about finding the compromise between quick reactions to network events (e.g. OOP even) versus the unwanted network churn that can occur when reacting to multiple events happening within a short time period (e.g. changing routes often in response to link flapping). Dampening allows the software to react quickly to initial events, while giving the network time to "relax" and adjust to the changes before starting any further actions.

`Periodic Timer` - by default, the Master Controller will look for an alternate path when an OOP event occurs. However, you may configure a complementary periodic optimization process, launched every time-interval specified using the Master Controller configuration command **`periodic <180-5000 sec>`**.The periodic process finds a better path for a traffic class entry, even if the traffic class entry is in-policy on the current exit. When the periodic timer expires, the Master Controller evaluates current exit links for the traffic class entry and if a better exit exists based on the current measurements, the traffic class entry is moved to a new in-policy exit link.

For all three timers, newly configured setting will immediately replace the existing setting if the value of the new setting is less than the time remaining. If the value is greater than the time remaining, the new setting will be applied when the existing timer expires or is reset.

OER Operational Mode Options

Three types of OER policy operational options could be set globally under the Master Controller configuration mode or per specific traffic class using the oer-map command **`set mode`**.

`Mode Monitor` - set the parameters for OER measurements. We encountered this mode before, when discussing the OER Measurement Phase. This can be changed using the command **`mode monitor {active|passive|both}`** under the Master Controller configuration mode or an oer-map entry using the **`set mode`** command.

`Mode Route` - this option specifies one of three OER route control policy settings. The **`mode route control`** command enables OER to control routes associated with the traffic classes automatically. **`mode route metric`** specifies OER route protocol-related settings, and **`mode route observe`** offers route control advice, but does not take any action. We will discuss this mode in the scenario dedicated to the route control.

Copyright © 2010 Internetwork Expert

`Mode Select-Exit` - this option defines the exit selection policy. While the traffic class is in policy using the currently assigned exit, OER does not search for an alternate exit link. This is the default mode configured using the command **`[set] mode select-exit good`** under the Master Controller configuration or oer-map entry. There are other scenarios where OER selects the best performance path, as opposed to just any path that satisfies the policy requirement for the traffic class. This type of configuration can be activated by using the **`[set] mode select-exit best`** command. In this type of situation, OER measures alternate path performance metrics while the traffic class entry is in-policy on the current path, and changes the current exit point if a better performance path is found. After the first selection of the best path, however, OER does not initiate another search unless the periodic timer is configured (see the timers above) or an OOP event occurs. When the periodic timer expires, the Master Controller evaluates the current exit links for the traffic class entry and, if a better exit exists based on the current measurements and priorities, the traffic class entry is moved to a new in-policy exit link. Notice that the periodic timer is disabled by default, so you may want to enable it when selecting the best exit path at any time.

There is another difference between the select-exit mode. If OER does not find an in-policy exit when in select-exit good mode, OER transitions the traffic class entry to an *uncontrolled* state. If the same event occurs when the select-exit best mode is active, OER selects the best of the OOP exit links for the traffic class entry.

Copyright © 2010 Internetwork Expert

OER Policy Application

OER policies can be applied to learned or configured traffic classes. OER policies can be applied on a global basis when the policy settings mentioned above are configured directly under OER Master Controller configuration mode. All traffic classes inherit global policies. If you want to apply a policy to a subset of the traffic classes, then a specific policy can be configured. Specific policies inherit global policy settings unless the same settings get overwritten by the specific policy.

To apply specific policies to learned or configured traffic classes, the OER map configuration is used. Use the match clause of the oer-map to define the traffic class of interest. It is possible to match any statically configured prefix, or match OER learned prefixes by means of the command **match oer learn {delay|throughput}**. The OER map applies the configuration of the **set** clause after a successful match occurs. An OER set clause can be used to set policy parameters such as the backoff timer, packet delay, holddown timer, packet loss, operational mode settings, periodic timer, resolve settings, unreachable hosts, etc. The OER map can be applied using the command **policy-list** in the Master Controller configuration mode.

Resolving Multiple OER Policies Conflicts

When configuring multiple policy criteria for a single traffic class entry, or a set of traffic classes, it is possible to have multiple conflicting policies select different exit points. For example, one exit point may provide best delay while the other has lowest link utilization. To resolve the potential conflict of the policy selection, OER uses its resolve function based on the priority set for a performance metric.

Each metric, such as delay, utilization range, loss, etc. is assigned a unique value, and the metric with the *lowest* numerical value is used to select the optimal exit point. By default, OER assigns the highest priority to delay settings, followed by utilization settings. To configure the policy conflict resolution, use the **resolve** command in OER master controller configuration mode, or the **set resolve** command in OER map configuration mode. However, prior to a configuration example, we need to discuss the metric variance settings for the resolution engine.

Copyright © 2010 Internetwork Expert

Using Variance for Policy Conflict Resolution

When configuring the OER resolve settings, you can also set an allowable variance for the defined policy values. Variance configures the acceptable "range" between the metrics measured for different exits that allows treating the different exits as equivalent with respect to a particular policy (e.g. a delay policy). Variance is specified in percent and defines the acceptable deviation from the *best* metric among all network exits. For example, if the delay on the best exit link for a traffic class is 80 ms and a 10% variance is configured, then any other exit links with a delay between 80 and 88 ms for the same traffic class entry are considered *equivalent* to the best exit link. This allows for OER to select the set of the exit points that are equal with respect to the highest-priority characteristic and then select the best based on the second-priority policy.

To illustrate this concept, look at the following example. The network has three exit links with the following characteristics (ppm = packet per million).

Exit A - Delay is 100 ms, jitter is 3ms, loss threshold absolute 3000 ppm
Exit B - Delay is 220 ms, jitter is 5ms, loss threshold absolute 1200 ppm
Exit C - Delay is 240 ms, jitter is 1ms, loss threshold absolute 1000 ppm

The following OER policy conflict resolution is configured and applied to the traffic class entry:

```
oer-map TEST 10
 match ip address prefix-list NETWORK
 set resolve loss priority 1 variance 30
 set resolve delay priority 2 variance 20
 set resolve jitter priority 3 variance 10
```

Assume all exit links are within the policy (all parameters satisfy the thresholds) and OER needs to find the best exit (select-exit best mode) for the configured prefix. First, OER looks for the highest-priority policy which loss. The best Exit is C and the acceptable variance is 30%, so any link with the absolute loss of up to 1300 is equivalent to link C. This appears to be Exit B. Now OER attempts to select the best exit between B and C based on the next priority policy, which is delay. The best exit link now is Exit B (delay 220 ms) and the acceptable variance is 20%, which puts Exit C on par with B, as 220*20%=44ms and 240 ms < 220+44 ms. Now, the final resolution should be done using the lowest priority policy which is jitter. It appears that Exit C has the lowest jitter value and based on this, the conflict resolution procedure selects Exit C as optimal.

Copyright © 2010 Internetwork Expert

Now let us briefly discuss the scenario itself. The task has a lot of policy requirements which should be analyzed and the following information extracted:

1) Global policy: metric thresholds and resolution priorities. From the task wording, we may conclude that the delay and loss metric should be bounded. We set the relative value for the loss to 25%, and the absolute threshold for the delay to 200ms. The conflict resolution is configured so that utilization is preferred over delay and the variance is set to 15%.

Additionally, the link utilization and range thresholds are configured globally in the oer Master Controller configuration and the Border Router settings. We reflect the maximum outbound utilization of 60% and the range of 10%.

2) Class-specific policy: there is one prefix that requires special handling, 150.X.6.0/24. We create an OER map entry matching this prefix and tune the absolute delay threshold to 100ms. Additionally we reverse the exit selection priority so that delay is preferred over utilization with the variance for both metrics being the same 15%.

3) Operational parameters: timers and modes. We set the periodic timer to 180 seconds, as this is the minimal value. At the same time, the exit selection mode is changed to “best”, so that the best path is selected on every period. Based on the requirement to hold traffic classes to their exits for 6 minutes after an OOP, we configure the holddown timer for 360 seconds.

Copyright © 2010 Internetwork Expert

R5:

```
ip prefix-list R6_LO0 permit 150.1.6.0/24
!
oer-map OER 30
 match ip address prefix-list R6_LO0
 set delay threshold 100
 set resolve delay priority 1 variance 15
 set resolve utilization priority 2 variance 15
!
oer master
 policy-rules OER
 max-range-utilization percent 10
!
 border 150.1.3.3 key-chain OER
  interface Serial1/2 external
   max-xmit-utilization percentage 80
  !
  interface FastEthernet0/0 external
   max-xmit-utilization percentage 80
 !
 border 150.1.5.5 key-chain OER
  interface Serial0/1 external
   max-xmit-utilization percentage 80
 !
 border 150.1.2.2 key-chain OER
  interface FastEthernet0/0 external
   max-xmit-utilization percentage 80
 !
 delay threshold 200
 loss relative 30
 holddown 600
 mode select-exit best
 periodic 90
 resolve utilization priority 1 variance 15
 resolve delay priority 2 variance 15
 max-range-utilization percent 10
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> At this point, policy has been applied but the route mode is set to observe by default. Thus, OER takes no action upon policy violation. You may check the current global (default) policy settings and the specific configuration using the following show command.

```
Rack1R5#show oer master policy
Default Policy Settings:
  backoff 300 3000 300
  delay threshold 200
  holddown 360
  periodic 180
  probe frequency 56
  number of jitter probe packets 100
  mode route observe
  mode monitor both
  mode select-exit best
  loss relative 25
  jitter threshold 20
  mos threshold 3.60 percent 30
  unreachable relative 50
  resolve utilization priority 1 variance 15
  resolve delay priority 2 variance 15
  resolve range priority 12 variance 0
oer-map OER 10
  sequence no. 8444249301975040, provider id 1, provider priority 30
    host priority 0, policy priority 10, Session id 0
  match ip prefix-lists: NET112
  backoff 300 3000 300
  delay threshold 200
  holddown 360
  periodic 180
  probe frequency 56
  number of jitter probe packets 100
  mode route observe
  mode monitor both
  mode select-exit best
  loss relative 25
  jitter threshold 20
  mos threshold 3.60 percent 30
  unreachable relative 50
  next-hop not set
  forwarding interface not set
  resolve utilization priority 1 variance 15
  resolve delay priority 2 variance 15
  resolve range priority 12 variance 0
oer-map OER 20
  sequence no. 8444249302630400, provider id 1, provider priority 30
    host priority 0, policy priority 20, Session id 0
  match ip prefix-lists: R4_LO0
  backoff 300 3000 300
```

Copyright © 2010 Internetwork Expert

```
  delay threshold 200
  holddown 360
  periodic 180
  probe frequency 56
  number of jitter probe packets 100
  mode route observe
 *mode monitor active
  mode select-exit best
  loss relative 25
  jitter threshold 20
  mos threshold 3.60 percent 30
  unreachable relative 50
  next-hop not set
  forwarding interface not set
  resolve utilization priority 1 variance 15
  resolve delay priority 2 variance 15
  resolve range priority 12 variance 0
oer-map OER 30
  sequence no. 8444249303285760, provider id 1, provider priority 30
    host priority 0, policy priority 30, Session id 0
  match ip prefix-lists: R6_LO0
  backoff 300 3000 300
 *delay threshold 100
  holddown 360
  periodic 180
  probe frequency 56
  number of jitter probe packets 100
  mode route observe
  mode monitor both
  mode select-exit best
  loss relative 25
  jitter threshold 20
  mos threshold 3.60 percent 30
  unreachable relative 50
  next-hop not set
  forwarding interface not set
 *resolve delay priority 1 variance 15
 *resolve utilization priority 2 variance 15
 *resolve range priority 12 variance 0

* Overrides Default Policy Setting
```

Note

Now check the border interface utilization thresholds. Notice that the Max BW is set based on 80% of the interface bandwidth.

```
Rack1R5#show oer master border detail
Border           Status   UP/DOWN             AuthFail  Version
150.1.2.2        ACTIVE   UP       01:45:52          0  1.0
 Se0/1           INTERNAL UP
 Fa0/0           EXTERNAL UP
 Se0/0.1         INTERNAL UP
```

Copyright © 2010 Internetwork Expert

```
 External            Capacity      Max BW   BW Used    Load Status
Exit Id
 Interface            (kbps)       (kbps)    (kbps)    (%)
 ---------           --------      ------   ------- ------- ------           --
----
 Fa0/0           Tx    100000       80000         0       0 UP
4
                 Rx                100000         0       0
-------------------------------------------------------------------------------
-
Border           Status   UP/DOWN             AuthFail  Version
150.1.5.5        ACTIVE   UP       01:46:32          0  2.2
 Serial0/1       EXTERNAL Invalid
 Se0/1/0         EXTERNAL UP
 Fa0/0           INTERNAL UP
 Se0/0/0         INTERNAL UP

 External            Capacity      Max BW   BW Used    Load Status
Exit Id
 Interface            (kbps)       (kbps)    (kbps)    (%)
 ---------           --------      ------   ------- ------- ------           --
----
 Se0/1/0         Tx       128          96         0       0 UP
3
                 Rx                   128         0       0
-------------------------------------------------------------------------------
-
Border           Status   UP/DOWN             AuthFail  Version
150.1.3.3        ACTIVE   UP       01:46:14          0  1.0
 Se1/2           EXTERNAL UP
 Se1/3           INTERNAL UP
 Se1/0.1         INTERNAL UP
 Fa0/0           EXTERNAL UP

 External            Capacity      Max BW   BW Used    Load Status
Exit Id
 Interface            (kbps)       (kbps)    (kbps)    (%)
 ---------           --------      ------   ------- ------- ------           --
----
 Se1/2           Tx       128         102       125      97 UP
2
                 Rx                   128        19      14
 Fa0/0           Tx       256         204         0       0 UP
1
                 Rx                   256         0       0
```

Copyright © 2010 Internetwork Expert

3.18 OER Control & Verify Phase

- Enable OER to control routes in the network domain.
- Use the tag value of 1000 for injected static routes and a BGP Local Preference value of 6000.
- Ensure that any injected static route will be redistributed into the IGP used by AS 200.
- To accelerate the convergence process, change the periodic-interval and monitor-period values to 1 on the Master Controller.

Configuration

Note

The next step in the OER performance optimization loop is the Control Phase, where OER actively attempts to influence traffic routing in the network. After the traffic classes have been profiled, their performance metrics determined, and then compared to the policy thresholds, OER needs to perform control actions to optimize the network exit point usage.

OER, by default, operates in an observation mode - the Master Controller monitors traffic classes and exit links based on the OER policies, reports the status of the network, but does not implement any changes. To activate the control mode, use the OER Master Controller **`mode route control`** command. In this mode, the Master Controller coordinates information from the Borders Routers in the same way as observe mode, but commands are sent back to the Border Routers to alter routing. You may selectively control a specific traffic class by using an OER map command **`set mode route control`**. This allows for the combined observe and control modes to be applied to different traffic classes.

OER initiates route changes when one of the following occurs:

`A traffic class goes OOP`, e.g. exceeds any metric threshold and OER starts looking for an alternate exit.

`An exit link goes OOP`, e.g. by exceeding the link utilization or losing link connectivity.

`The periodic timer expires` and the select exit mode is configured as select best mode. This instructs OER to looks for a better path if any exists, and possible re-route the traffic class.

Copyright © 2010 Internetwork Expert

During the OER control phase, the Master Controller continues to monitor all in-policy traffic classes to ensure that they remain in-policy. Changes are only implemented for OOP traffic classes in order to bring them in-policy. The backoff and holddown timers described in the previous task can be used to provide dampening in an OER-managed network. If aggressive delay or loss policies are defined, and the exit links are seriously over-subscribed (i.e. peak usage might exceed the maximum link bandwidth), it is possible that OER will find it impossible to bring a traffic class in-policy. In this case, the Master Controller will either choose the link that most closely conforms to the performance policy, even though the traffic class still remains OOP, or it will remove the prefix from OER control. OER is designed to allow you to make the best use of available bandwidth, but it does not solve the problem of over-subscribed bandwidth.

OER Traffic Class Control Techniques

We are going to review just the basic route control techniques here, available in IOS 12.4. As for the entrance link selecting features available in recent IOS revisions and other advanced features; these will be covered in separate tasks.

There are a number of techniques that can be used to influence the routing decision; by changing IGP prefix metrics, altering BGP attributes, or introducing policy-based routing using a route map. If the traffic associated with the traffic class is defined only by a prefix, then a traditional routing control mechanism such as injecting a BGP route, or a static route can be used. This type of control is network-wide, as new prefixes are redistributed either into IGP or BGP and propagate to all network devices, influencing path selection at all routers. Before injecting a route, OER will verify that a route exists in the BGP or static table that includes the prefix and points to the exit link. This route may be a default route or a supernet, matching the prefix being injected. This condition ensures that the Border Router actually has the path to reach the injected prefix. Notice that OER will automatically select the type of injected route (BGP or static) based on the matching route present in the Border Router's routing table.

If the traffic associated with the traffic class is defined by a prefix and other matching criteria for the packet such as port number, then traditional routing cannot be employed to control the application traffic. In this situation, the control becomes device specific and not network specific. This device specific control is implemented by OER using policy-based routing (PBR) functionality and available since IOS 12.4(2)T. Notice that using PBR to change the routing decisions requires the Border Routers to be one-hop away, either physically or via a tunnel mesh. We are not going to discuss PBR-based route control in this task, but mention it here for the sake of completeness.

Copyright © 2010 Internetwork Expert

Static Route Injection

An OER Master Controller can enforce the use of a particular Border Router as the preferred exit link for a traffic class by injecting temporary static routes. These routes are temporary and intentionally not saved to the permanent configuration. There are different methods that the Master Controller can use to inject static routes in the Border Routers.

(a) Existing static routes can be overwritten with new static routes having a better routing metric.
(b) The Master Controller may inject a more specific route, if the Border Routers use default routing. The longest-match routing will attract the traffic to the proper exit point.
(c) The Master Controller can also use the technique known a split prefixes. This method adds a more specific prefix in the existence of a less-specific one. For example, if the border route has a static route to 10.0.0.0/8, then adding the prefix 10.0.1.0/16 to another Border Router will "split" traffic for the specific subnet. OER can use split prefixes to redirect subsets of an existing prefix to a more optimal exit link, and can use split prefixes for both internal BGP (iBGP) and static routes.

When redistributing the static injected routes, it is helpful to know what routes were injected by OER. To make this easy, you may instruct the OER process to associate a tag with the static routes. The injected static route will be tagged with the value specified using the command **mode route metric static tag <value>**. You may reference this tag when performing route redistribution using the route-map command **match tag**.

BGP Control Techniques

OER uses two BGP techniques to enforce the best exit path; injecting a BGP route, or modifying the BGP local preference attribute. BGP route injection is similar to split-prefix static route injection, but it happens in the BGP table. All OER injected routes remain local to an autonomous system, and these injected routes are never shared with external BGP peers. To ensure this behavior, when OER injects a BGP route, it will set the no-export community. This requires enabling the send-community configuration for all iBGP peers to make sure that the prefix will not leak out of the AS. In some cases, it may also be necessary to redistribute this information into the IGP, to make sure all routers (even non-BGP speakers) learn the new information.

Copyright © 2010 Internetwork Expert

OER may also use BGP local preference to control traffic classes. The Master Controller instructs one of the Border Routers to apply the Local Preference attribute to a prefix or set of prefixes associated with a traffic class. This information is then propagated to all iBGP peers affecting best-path selection within the AS. Once the iBGP convergence is complete, the Border Router with the highest Local Preference for the prefix will become the exit link from the network. For example, assume that all border peers learn the prefix 10.0.0.0/24 which is under OER control. In order to select a particular exit point, OER will adjust the Local Preference on the respective border peer and this will attract the traffic to the proper exit.

In order to specify the Local Preference value used by OER injected/adjusted prefixes, use the OER Master Controller configuration command **`mode route metric bgp local-pref <value>`**. You may change the Local Preference value for a particular traffic class using the OER map command **`set mode route metric bgp local-pref <value>`**. Notice that changing the static metric values requires that you reset the active OER Master Controller connections using the command **`clear oer master border *`**.

OER Verify Phase

The last phase of the OER performance loop is to verify that the actions taken during the OER control phase actually change the flow of traffic, and that the performance of the traffic class or link has moved to an in-policy state. OER uses NetFlow for automatic route control verification. The Master Controller expects a Netflow update for the traffic class from the new exit link interface and ignores Netflow updates from the previous exit. If a Netflow update does not appear after two minutes, the Master Controller moves the traffic class into the default state, releasing it from OER control.

In this scenario, we activate route control by default, but disable it for the prefix 150.X.6.0/24. Additionally, we configured the OER BGP Local Preference value of 6000 and static route tag of 1000. We do not configure BGP community sending on iBGP peering sessions, as no new prefixes are expected to be injected into BGP. OER is only supposed to modify the Local Preference to affect the exit points.

Copyright © 2010 Internetwork Expert

R2:

```
router ospf 1
 redistribute static route-map STATIC_TO_OSPF
!
route-map STATIC_TO_OSPF
 match tag 1000
```

R3:

```
router ospf 1
 redistribute static route-map STATIC_TO_OSPF
!
route-map STATIC_TO_OSPF
 match tag 1000
```

R5:

```
oer-map 10
 set mode route observe
!
oer master
 mode route control
 mode route metric static tag 1000
 mode route metric bgp local-pref 6000
!
router ospf 1
 redistribute static route-map STATIC_TO_OSPF
!
route-map STATIC_TO_OSPF
 match tag 1000
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> To verify route control, we configure R1, R4, and R6 for downloading of IOS images via HTTP from R2 and SW2 again. We are going to use the Loopback0 interfaces on R1, R4, and R6 to source the connections. This will simulate three traffic flows from AS 200 to AS 100. Additionally, we configure R5 so that traffic from SW2 is limited to approximately 112Kbps and thus may "fit" the "slow" link between R4 and R5. The transfers from R2 are limited to 128Kbps automatically by the speed of the link connecting R2 and R3.

R1:

```
ip http client source-interface Loopback0

copy http://cisco:cisco@150.1.2.2/c2600-adventerprisek9-mz.124-17.bin
null:
```

R4:

```
ip http client source-interface Loopback0

Rack1R4#copy http://cisco:cisco@150.1.8.8/c3560-advipservicesk9-mz.122-
46.SE.bin null:
```

R5:

```
ip access-list extended HTTP
 permit tcp any eq www any
!
class-map match-all HTTP
 match access-group name HTTP
!
policy-map RATE_LIMIT
 class HTTP
    police 112000
!
interface FastEthernet 0/0
 service-policy input RATE_LIMIT
```

R6:

```
ip http client source-interface Loopback0

copy http://cisco:cisco@150.1.8.8/c3560-advipservicesk9-mz.122-
46.SE.bin null:
```

R2 & SW2:

```
ip http server
ip http path flash:
```

Copyright © 2010 Internetwork Expert

Note

Check the exit link selection on the Master Controller router. Notice that two prefixes are in policy state, while prefix for R4 is in holddown.

Again it is important to note that the products of the following show commands are "snap shots" of the particular router's status at a given time. The performance routing process is a cycle that constantly monitors the status of the specified interfaces and will continually make changes in an attempt to optimize traffic flow based on established parameters.

```
Rack1R5#show oer master prefix
OER Prefix Statistics:
 Pas - Passive, Act - Active, S - Short term, L - Long term, Dly - Delay (ms),
 Los - Packet Loss (packets-per-million), Un - Unreachable (flows-per-million),
 E - Egress, I - Ingress, Bw - Bandwidth (kbps), N - Not applicable
 U - unknown, * - uncontrolled, + - control more specific, @ - active probe all

Prefix                  State      Time Curr BR         CurrI/F         Protocol
                      PasSDly   PasLDly   PasSUn    PasLUn   PasSLos   PasLLos
                      ActSDly   ActLDly   ActSUn    ActLUn       EBw       IBw
-----------------------------------------------------------------------------
112.0.0.0/24          DEFAULT*         3 U                 U
150.1.1.0/24          HOLDDOWN       238 150.1.5.5         Se0/1/0           BGP
                            U          U         0         0       628       628
                            U          U         0         0        69        10
150.1.4.0/24          INPOLICY       @48 150.1.3.3         Se1/2             BGP
                            N          N         N         N         N         N
                           16         16         0         0         N         N
150.1.6.0/24          INPOLICY       @62 150.1.3.3         Fa0/0             BGP
                            U          U         0         0         0      1509
                           16         11         0         0         3         4
```

Note

Now check the OER controlled routes in the BGP tables of all Border Routers. Notice that R3 and R5 show the prefixes as "Controlled" meaning these prefixes have been modified by OER. The Local Preference value is set to 6000 per our configuration. The next-hop values in the BGP table show the exit currently used for the particular traffic class.

The current exit distribution is different from the one that would be used by default. You can see that all three exits from the AS are used concurrently; even the path across AS 300 (via SW1) that would not be preferred under normal conditions.

Copyright © 2010 Internetwork Expert

Note

This lab does not inject bgp routes to affect traffic flow, rather it manipulates the local preference of specific routes in order to affect exit points the traffic uses to leave the AS. Please observe that the Master Controller has done exactly that, on each Border Router having exit interfaces "of interest" to the policies defined on the MC.

```
Rack1R5#show oer border routes bgp
OER BR 150.1.2.2 ACTIVE, MC 150.1.2.2 UP/DOWN: UP 00:36:49,
  Auth Failures: 0
  Conn Status: SUCCESS, PORT: 3949
BGP table version is 64, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
OER Flags: C - Controlled, X - Excluded, E - Exact, N - Non-exact, I -
Injected

   Network          Next Hop        OER    LocPrf Weight Path
*>i150.1.1.0/24     155.1.45.4      XN       6000      0 100 i
*>i150.1.4.0/24     155.1.13.1      XN       6000      0 100 i
*>i150.1.6.0/24     155.1.37.7      XN       6000      0 300 100 i

Rack1R3#show oer border routes bg
OER BR 150.1.3.3 ACTIVE, MC 150.1.2.2 UP/DOWN: UP 00:36:56,
  Auth Failures: 0
  Conn Status: SUCCESS, PORT: 3949
BGP table version is 65, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
OER Flags: C - Controlled, X - Excluded, E - Exact, N - Non-exact, I -
Injected

   Network          Next Hop        OER    LocPrf Weight Path
*>i150.1.1.0/24     155.1.45.4      XN       6000      0 100 i
*> 150.1.4.0/24     155.1.13.1      CE                 0 100 i
*> 150.1.6.0/24     155.1.37.7      CE                 0 300 100 i
Rack1R3#
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show oer border routes bgp
OER BR 150.1.5.5 ACTIVE, MC 150.1.2.2 UP/DOWN: UP 00:37:02,
  Auth Failures: 0
  Conn Status: SUCCESS, PORT: 3949
BGP table version is 60, local router ID is 150.1.5.5
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
OER Flags: C - Controlled, X - Excluded, E - Exact, N - Non-exact, I -
Injected

   Network          Next Hop        OER    LocPrf Weight Path
*> 150.1.1.0/24     155.1.45.4      CE                0 100 i
*>i150.1.4.0/24     155.1.13.1      XN      6000      0 100 i
*>i150.1.6.0/24     155.1.37.7      XN      6000      0 300 100 i
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

RIP Solutions

Note

Load the *RIP* initial configurations prior to starting. Note that R4's connection to VLAN 146 and the Serial link between R2 and R3 are disabled.

4.1 Basic RIP Configuration

- Configure RIPv2 on all interfaces of all devices in the internal network.
- Disable auto-summary.
- R4 and R6 should be learning RIP routes from BB3 and BB1 respectively.
- Test reachability to all networks and note any problems within the topology.

Configuration

```
R1:
router rip
 version 2
 network 150.1.0.0
 network 155.1.0.0
 no auto-summary

R2:
router rip
 version 2
 network 150.1.0.0
 network 155.1.0.0
 network 192.10.1.0
 no auto-summary

R3:
router rip
 version 2
 network 150.1.0.0
 network 155.1.0.0
 no auto-summary

R4:
router rip
 version 2
 network 150.1.0.0
 network 155.1.0.0
 network 204.12.1.0
 no auto-summary
```

Copyright © 2010 Internetwork Expert

```
R5:
router rip
 version 2
 network 150.1.0.0
 network 155.1.0.0
 no auto-summary

R6:
router rip
 version 2
 network 54.0.0.0
 network 150.1.0.0
 network 155.1.0.0
 no auto-summary

SW1:
ip routing
!
router rip
 version 2
 network 150.1.0.0
 network 155.1.0.0
 no auto-summary

SW2:
ip routing
!
router rip
 version 2
 network 150.1.0.0
 network 155.1.0.0
 no auto-summary

SW3:
ip routing
!
router rip
 version 2
 network 150.1.0.0
 network 155.1.0.0
 no auto-summary

SW4:
ip routing
!
router rip
 version 2
 network 150.1.0.0
 network 155.1.0.0
 no auto-summary
```

Copyright © 2010 Internetwork Expert

Verification

```
Rack1R5#debug ip rip
RIP protocol debugging is on
```

> **Note**
>
> R5 receives an update from R1 which includes the routes to BB1

```
RIP: received v2 update from 155.1.0.1 on Serial0/0.1
     54.1.1.0/24 via 0.0.0.0 in 2 hops
     150.1.1.0/24 via 0.0.0.0 in 1 hops
     150.1.3.0/24 via 0.0.0.0 in 2 hops
     150.1.6.0/24 via 0.0.0.0 in 2 hops
     150.1.7.0/24 via 0.0.0.0 in 3 hops
     150.1.9.0/24 via 0.0.0.0 in 4 hops
     155.1.7.0/24 via 0.0.0.0 in 3 hops
     155.1.9.0/24 via 0.0.0.0 in 4 hops
     155.1.13.0/24 via 0.0.0.0 in 1 hops
     155.1.37.0/24 via 0.0.0.0 in 2 hops
     155.1.67.0/24 via 0.0.0.0 in 2 hops
     155.1.79.0/24 via 0.0.0.0 in 3 hops
     155.1.146.0/24 via 0.0.0.0 in 1 hops
     212.18.0.0/24 via 0.0.0.0 in 3 hops
     212.18.1.0/24 via 0.0.0.0 in 3 hops
     212.18.2.0/24 via 0.0.0.0 in 3 hops
     212.18.3.0/24 via 0.0.0.0 in 3 hops
```

R5 receives an update from R2 which includes the route to the link to BB2.

```
RIP: received v2 update from 155.1.0.2 on Serial0/0.1
     150.1.2.0/24 via 0.0.0.0 in 1 hops
     192.10.1.0/24 via 0.0.0.0 in 1 hops
```

Copyright © 2010 Internetwork Expert

R5 receives updates from R4 from both the Frame Relay and point-to-point links which include the routes to BB3.

```
RIP: received v2 update from 155.1.0.4 on Serial0/0.1
     30.0.0.0/16 via 0.0.0.0 in 2 hops
     30.1.0.0/16 via 0.0.0.0 in 2 hops
     30.2.0.0/16 via 0.0.0.0 in 2 hops
     30.3.0.0/16 via 0.0.0.0 in 2 hops
     31.0.0.0/16 via 0.0.0.0 in 2 hops
     31.1.0.0/16 via 0.0.0.0 in 2 hops
     31.2.0.0/16 via 0.0.0.0 in 2 hops
     31.3.0.0/16 via 0.0.0.0 in 2 hops
     54.1.1.0/24 via 0.0.0.0 in 4 hops
     150.1.1.0/24 via 0.0.0.0 in 3 hops
     150.1.2.0/24 via 0.0.0.0 in 3 hops
     150.1.3.0/24 via 0.0.0.0 in 3 hops
     150.1.4.0/24 via 0.0.0.0 in 1 hops
     150.1.6.0/24 via 0.0.0.0 in 4 hops
     150.1.7.0/24 via 0.0.0.0 in 4 hops
     150.1.9.0/24 via 0.0.0.0 in 5 hops
     155.1.7.0/24 via 0.0.0.0 in 4 hops
     155.1.9.0/24 via 0.0.0.0 in 5 hops
     155.1.13.0/24 via 0.0.0.0 in 3 hops
     155.1.37.0/24 via 0.0.0.0 in 3 hops
     155.1.45.0/24 via 0.0.0.0 in 1 hops
     155.1.67.0/24 via 0.0.0.0 in 4 hops
     155.1.79.0/24 via 0.0.0.0 in 4 hops
     155.1.146.0/24 via 0.0.0.0 in 3 hops
     192.10.1.0/24 via 0.0.0.0 in 3 hops
RIP: received v2 update from 155.1.0.4 on Serial0/0.1
     204.12.1.0/24 via 0.0.0.0 in 1 hops
     212.18.0.0/24 via 0.0.0.0 in 5 hops
     212.18.1.0/24 via 0.0.0.0 in 5 hops
     212.18.2.0/24 via 0.0.0.0 in 5 hops
     212.18.3.0/24 via 0.0.0.0 in 5 hops

RIP: received v2 update from 155.1.45.4 on Serial0/1
     30.0.0.0/16 via 0.0.0.0 in 2 hops
     30.1.0.0/16 via 0.0.0.0 in 2 hops
     30.2.0.0/16 via 0.0.0.0 in 2 hops
     30.3.0.0/16 via 0.0.0.0 in 2 hops
     31.0.0.0/16 via 0.0.0.0 in 2 hops
     31.1.0.0/16 via 0.0.0.0 in 2 hops
     31.2.0.0/16 via 0.0.0.0 in 2 hops
     31.3.0.0/16 via 0.0.0.0 in 2 hops
     150.1.4.0/24 via 0.0.0.0 in 1 hops
     155.1.0.0/24 via 0.0.0.0 in 1 hops
     204.12.1.0/24 via 0.0.0.0 in 1 hops
```

Copyright © 2010 Internetwork Expert

Due to split horizon being enabled on the multipoint Frame Relay subinterface of R5, the routes coming from R1, R2, R3, and R4 on this link cannot be advertised back out the Frame Relay. However, updates from R1, R2, and R3 on the Frame Relay network can be advertised to R4 out the point-to-point link.

This means that R4, SW2, and SW4 will have full routing information to everyone, per the below output.

```
Rack1R4#show ip route rip
     155.1.0.0/24 is subnetted, 14 subnets
R       155.1.146.0 [120/2] via 155.1.45.5, 00:00:25, Serial0/1/0
R       155.1.10.0 [120/3] via 155.1.45.5, 00:00:25, Serial0/1/0
                   [120/3] via 155.1.0.5, 00:00:13, Serial0/0/0.1
R       155.1.8.0 [120/2] via 155.1.45.5, 00:00:25, Serial0/1/0
                  [120/2] via 155.1.0.5, 00:00:13, Serial0/0/0.1
R       155.1.9.0 [120/4] via 155.1.45.5, 00:00:25, Serial0/1/0
R       155.1.13.0 [120/2] via 155.1.45.5, 00:00:25, Serial0/1/0
R       155.1.7.0 [120/3] via 155.1.45.5, 00:00:25, Serial0/1/0
R       155.1.5.0 [120/1] via 155.1.45.5, 00:00:25, Serial0/1/0
                  [120/1] via 155.1.0.5, 00:00:13, Serial0/0/0.1
R       155.1.58.0 [120/1] via 155.1.45.5, 00:00:25, Serial0/1/0
                   [120/1] via 155.1.0.5, 00:00:13, Serial0/0/0.1
R       155.1.37.0 [120/2] via 155.1.45.5, 00:00:25, Serial0/1/0
R       155.1.79.0 [120/3] via 155.1.45.5, 00:00:25, Serial0/1/0
R       155.1.67.0 [120/3] via 155.1.45.5, 00:00:25, Serial0/1/0
R       155.1.108.0 [120/2] via 155.1.45.5, 00:00:26, Serial0/1/0
                    [120/2] via 155.1.0.5, 00:00:14, Serial0/0/0.1
     54.0.0.0/24 is subnetted, 1 subnets
R       54.1.1.0 [120/3] via 155.1.45.5, 00:00:27, Serial0/1/0
R    212.18.1.0/24 [120/4] via 155.1.45.5, 00:00:26, Serial0/1/0
R    212.18.0.0/24 [120/4] via 155.1.45.5, 00:00:27, Serial0/1/0
R    212.18.3.0/24 [120/4] via 155.1.45.5, 00:00:27, Serial0/1/0
R    212.18.2.0/24 [120/4] via 155.1.45.5, 00:00:27, Serial0/1/0
R    192.10.1.0/24 [120/2] via 155.1.45.5, 00:00:27, Serial0/1/0
     31.0.0.0/16 is subnetted, 4 subnets
R       31.3.0.0 [120/1] via 204.12.1.254, 00:00:08, FastEthernet0/0
R       31.2.0.0 [120/1] via 204.12.1.254, 00:00:08, FastEthernet0/0
R       31.1.0.0 [120/1] via 204.12.1.254, 00:00:08, FastEthernet0/0
R       31.0.0.0 [120/1] via 204.12.1.254, 00:00:08, FastEthernet0/0
     150.1.0.0/24 is subnetted, 10 subnets
R       150.1.7.0 [120/3] via 155.1.45.5, 00:00:27, Serial0/1/0
R       150.1.6.0 [120/3] via 155.1.45.5, 00:00:28, Serial0/1/0
R       150.1.5.0 [120/1] via 155.1.45.5, 00:00:28, Serial0/1/0
                  [120/1] via 155.1.0.5, 00:00:15, Serial0/0/0.1
R       150.1.3.0 [120/2] via 155.1.45.5, 00:00:28, Serial0/1/0
R       150.1.2.0 [120/2] via 155.1.45.5, 00:00:28, Serial0/1/0
R       150.1.1.0 [120/2] via 155.1.45.5, 00:00:28, Serial0/1/0
R       150.1.10.0 [120/3] via 155.1.45.5, 00:00:28, Serial0/1/0
                   [120/3] via 155.1.0.5, 00:00:15, Serial0/0/0.1
R       150.1.9.0 [120/4] via 155.1.45.5, 00:00:28, Serial0/1/0
R       150.1.8.0 [120/2] via 155.1.45.5, 00:00:28, Serial0/1/0
                  [120/2] via 155.1.0.5, 00:00:15, Serial0/0/0.1
     30.0.0.0/16 is subnetted, 4 subnets
R       30.2.0.0 [120/1] via 204.12.1.254, 00:00:09, FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

```
R       30.3.0.0 [120/1] via 204.12.1.254, 00:00:09, FastEthernet0/0
R       30.0.0.0 [120/1] via 204.12.1.254, 00:00:09, FastEthernet0/0
R       30.1.0.0 [120/1] via 204.12.1.254, 00:00:09, FastEthernet0/0

Rack1SW2#show ip route rip
R    204.12.1.0/24 [120/2] via 155.1.58.5, 00:00:09, Vlan58
     155.1.0.0/24 is subnetted, 14 subnets
R       155.1.146.0 [120/2] via 155.1.58.5, 00:00:09, Vlan58
R       155.1.10.0 [120/1] via 155.1.108.10, 00:00:09, Port-channel1
R       155.1.9.0 [120/4] via 155.1.58.5, 00:00:09, Vlan58
R       155.1.13.0 [120/2] via 155.1.58.5, 00:00:09, Vlan58
R       155.1.0.0 [120/1] via 155.1.58.5, 00:00:09, Vlan58
R       155.1.7.0 [120/3] via 155.1.58.5, 00:00:09, Vlan58
R       155.1.5.0 [120/1] via 155.1.58.5, 00:00:09, Vlan58
R       155.1.45.0 [120/1] via 155.1.58.5, 00:00:10, Vlan58
R       155.1.37.0 [120/2] via 155.1.58.5, 00:00:10, Vlan58
R       155.1.79.0 [120/3] via 155.1.58.5, 00:00:10, Vlan58
R       155.1.67.0 [120/3] via 155.1.58.5, 00:00:10, Vlan58
     54.0.0.0/24 is subnetted, 1 subnets
R       54.1.1.0 [120/3] via 155.1.58.5, 00:00:10, Vlan58
R    212.18.1.0/24 [120/4] via 155.1.58.5, 00:00:10, Vlan58
R    212.18.0.0/24 [120/4] via 155.1.58.5, 00:00:10, Vlan58
R    212.18.3.0/24 [120/4] via 155.1.58.5, 00:00:10, Vlan58
R    212.18.2.0/24 [120/4] via 155.1.58.5, 00:00:10, Vlan58
R    192.10.1.0/24 [120/2] via 155.1.58.5, 00:00:10, Vlan58
     31.0.0.0/16 is subnetted, 4 subnets
R       31.3.0.0 [120/3] via 155.1.58.5, 00:00:10, Vlan58
R       31.2.0.0 [120/3] via 155.1.58.5, 00:00:10, Vlan58
R       31.1.0.0 [120/3] via 155.1.58.5, 00:00:10, Vlan58
R       31.0.0.0 [120/3] via 155.1.58.5, 00:00:10, Vlan58
     150.1.0.0/24 is subnetted, 10 subnets
R       150.1.7.0 [120/3] via 155.1.58.5, 00:00:10, Vlan58
R       150.1.6.0 [120/3] via 155.1.58.5, 00:00:11, Vlan58
R       150.1.5.0 [120/1] via 155.1.58.5, 00:00:11, Vlan58
R       150.1.4.0 [120/2] via 155.1.58.5, 00:00:11, Vlan58
R       150.1.3.0 [120/2] via 155.1.58.5, 00:00:11, Vlan58
R       150.1.2.0 [120/2] via 155.1.58.5, 00:00:11, Vlan58
R       150.1.1.0 [120/2] via 155.1.58.5, 00:00:11, Vlan58
R       150.1.10.0 [120/1] via 155.1.108.10, 00:00:11, Port-channel1
R       150.1.9.0 [120/4] via 155.1.58.5, 00:00:11, Vlan58
     30.0.0.0/16 is subnetted, 4 subnets
R       30.2.0.0 [120/3] via 155.1.58.5, 00:00:11, Vlan58
R       30.3.0.0 [120/3] via 155.1.58.5, 00:00:11, Vlan58
R       30.0.0.0 [120/3] via 155.1.58.5, 00:00:11, Vlan58
R       30.1.0.0 [120/3] via 155.1.58.5, 00:00:11, Vlan58
```

Copyright © 2010 Internetwork Expert

R2 will only have reachability information about connected networks of R5 and the routes learned from R5's link to SW2, since routes from R1, R3, and R4 cannot come in the Frame Relay of R5 and go back out the Frame Relay to reach R2.

```
Rack1R2#show ip route rip
     155.1.0.0/24 is subnetted, 7 subnets
R       155.1.10.0 [120/3] via 155.1.0.5, 00:00:07, Serial0/0.1
R       155.1.8.0 [120/2] via 155.1.0.5, 00:00:07, Serial0/0.1
R       155.1.5.0 [120/1] via 155.1.0.5, 00:00:07, Serial0/0.1
R       155.1.58.0 [120/1] via 155.1.0.5, 00:00:07, Serial0/0.1
R       155.1.45.0 [120/1] via 155.1.0.5, 00:00:07, Serial0/0.1
R       155.1.108.0 [120/2] via 155.1.0.5, 00:00:07, Serial0/0.1
     150.1.0.0/24 is subnetted, 4 subnets
R       150.1.5.0 [120/1] via 155.1.0.5, 00:00:07, Serial0/0.1
R       150.1.10.0 [120/3] via 155.1.0.5, 00:00:07, Serial0/0.1
R       150.1.8.0 [120/2] via 155.1.0.5, 00:00:07, Serial0/0.1
```

R1, R3, R6, SW1, and SW3 will not have reachability information about the link to BB2 or the routes learned from BB3, since R1 and R3 cannot learn the routes R2 and R4 send to R5 over the Frame Relay network.

```
Rack1SW3#show ip route rip
     155.1.0.0/24 is subnetted, 14 subnets
R       155.1.146.0 [120/2] via 155.1.79.7, 00:00:04, Vlan79
R       155.1.10.0 [120/5] via 155.1.79.7, 00:00:04, Vlan79
R       155.1.8.0 [120/4] via 155.1.79.7, 00:00:04, Vlan79
R       155.1.13.0 [120/2] via 155.1.79.7, 00:00:04, Vlan79
R       155.1.0.0 [120/2] via 155.1.79.7, 00:00:04, Vlan79
R       155.1.7.0 [120/1] via 155.1.79.7, 00:00:04, Vlan79
R       155.1.5.0 [120/3] via 155.1.79.7, 00:00:04, Vlan79
R       155.1.58.0 [120/3] via 155.1.79.7, 00:00:04, Vlan79
R       155.1.45.0 [120/3] via 155.1.79.7, 00:00:04, Vlan79
R       155.1.37.0 [120/1] via 155.1.79.7, 00:00:04, Vlan79
R       155.1.67.0 [120/1] via 155.1.79.7, 00:00:04, Vlan79
R       155.1.108.0 [120/4] via 155.1.79.7, 00:00:04, Vlan79
     54.0.0.0/24 is subnetted, 1 subnets
R       54.1.1.0 [120/2] via 155.1.79.7, 00:00:04, Vlan79
R    212.18.1.0/24 [120/3] via 155.1.79.7, 00:00:04, Vlan79
R    212.18.0.0/24 [120/3] via 155.1.79.7, 00:00:04, Vlan79
R    212.18.3.0/24 [120/3] via 155.1.79.7, 00:00:04, Vlan79
R    212.18.2.0/24 [120/3] via 155.1.79.7, 00:00:05, Vlan79
     150.1.0.0/24 is subnetted, 8 subnets
R       150.1.7.0 [120/1] via 155.1.79.7, 00:00:05, Vlan79
R       150.1.6.0 [120/2] via 155.1.79.7, 00:00:05, Vlan79
R       150.1.5.0 [120/3] via 155.1.79.7, 00:00:05, Vlan79
R       150.1.3.0 [120/2] via 155.1.79.7, 00:00:05, Vlan79
R       150.1.1.0 [120/3] via 155.1.79.7, 00:00:05, Vlan79
R       150.1.10.0 [120/5] via 155.1.79.7, 00:00:05, Vlan79
R       150.1.8.0 [120/4] via 155.1.79.7, 00:00:05, Vlan79
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

4.2 RIPv2 Authentication

- Configure RIPv2 authentication on the Ethernet link between R2 and BB2.
- Use the MD5 key number 1 with a password of CISCO.
- R2 should be learning RIP routes from BB2.
- Configure clear-text RIP authentication on the segment between R1 and R6 using the password CCIE.

Configuration

```
R1:
key chain RIP
 key 1
  key-string CCIE
!
interface FastEthernet0/0
 ip rip authentication mode text
 ip rip authentication key-chain RIP

R2:
key chain RIP
 key 1
  key-string CISCO
!
interface FastEthernet0/0
 ip address 192.10.1.2 255.255.255.0
 ip rip authentication mode md5
 ip rip authentication key-chain RIP

R6:
key chain RIP
 key 1
  key-string CCIE
!
interface FastEthernet0/0.146
 ip rip authentication mode text
 ip rip authentication key-chain RIP
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Without authentication configured R2 receives updates from BB2 but cannot install them in the RIP database.

```
Rack1R2#debug ip rip
RIP protocol debugging is on
RIP: received packet with MD5 authentication
RIP: ignored v2 packet from 192.10.1.254 (invalid authentication)
```

Once authentication is enabled updates from BB2 are installed.

```
Rack1R2#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R2(config)#key chain RIP
Rack1R2(config-keychain)#key 1
Rack1R2(config-keychain-key)#key-string CISCO
Rack1R2(config)#interface Fa0/0
Rack1R2(config-if)#ip rip authentication key-chain RIP
Rack1R2(config-if)#ip rip authentication mode md5
Rack1R2(config-if)#end
Rack1R2#
Rack1R2#show ip route rip
R    222.22.2.0/24 [120/7] via 192.10.1.254, 00:00:07, FastEthernet0/0
     155.1.0.0/24 is subnetted, 7 subnets
R       155.1.10.0 [120/3] via 155.1.0.5, 00:00:28, Serial0/0.1
R       155.1.8.0 [120/2] via 155.1.0.5, 00:00:28, Serial0/0.1
R       155.1.5.0 [120/1] via 155.1.0.5, 00:00:28, Serial0/0.1
R       155.1.58.0 [120/1] via 155.1.0.5, 00:00:28, Serial0/0.1
R       155.1.45.0 [120/1] via 155.1.0.5, 00:00:28, Serial0/0.1
R       155.1.108.0 [120/2] via 155.1.0.5, 00:00:28, Serial0/0.1
R    220.20.3.0/24 [120/7] via 192.10.1.254, 00:00:07, FastEthernet0/0
     150.1.0.0/24 is subnetted, 4 subnets
R       150.1.5.0 [120/1] via 155.1.0.5, 00:00:28, Serial0/0.1
R       150.1.10.0 [120/3] via 155.1.0.5, 00:00:28, Serial0/0.1
R       150.1.8.0 [120/2] via 155.1.0.5, 00:00:28, Serial0/0.1
R    205.90.31.0/24 [120/7] via 192.10.1.254, 00:00:07, FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

Pitfall

Whitespace counts as a valid character for key chain authentication. Use the **show key-chain** command to ensure that a space is not at the end of the authentication key for either RIP or EIGRP.

```
Rack1R2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R2(config)#key chain RIP
Rack1R2(config-keychain)#key 1
Rack1R2(config-keychain-key)#key-string CISCO ?
LINE    <cr>

Rack1R2(config-keychain-key)#key-string CISCO
Rack1R2(config-keychain-key)#end
Rack1R2#

Rack1R2#show key chain
Key-chain RIP:
    key 1 -- text "CISCO "
        accept lifetime (always valid) - (always valid) [valid now]
        send lifetime (always valid) - (always valid) [valid now]

Rack1R2#debug ip rip
RIP protocol debugging is on
Rack1R2#
RIP: received packet with MD5 authentication
RIP: ignored v2 packet from 192.10.1.254 (invalid authentication)
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

4.3 RIPv2 Split Horizon

- Disable split-horizon on R5's connection to the Frame Relay cloud.
- Test reachability to all networks and note any changes within the topology.

Configuration

```
R5:
interface Serial0/0/0.1 multipoint
 no ip split-horizon
```

Verification

> **Note**
>
> Now that R5 has split horizon disabled on the Frame Relay subinterface, R1, R2, and R3 can be advertised all updates, and all devices in the topology have full reachability information.

```
Rack1R2#show ip route rip
R    222.22.2.0/24 [120/7] via 192.10.1.254, 00:00:21, FastEthernet0/0
R    204.12.1.0/24 [120/2] via 155.1.0.4, 00:00:17, Serial0/0.1
     155.1.0.0/24 is subnetted, 14 subnets
R       155.1.146.0 [120/2] via 155.1.0.1, 00:00:17, Serial0/0.1
R       155.1.10.0 [120/3] via 155.1.0.5, 00:00:18, Serial0/0.1
R       155.1.8.0 [120/2] via 155.1.0.5, 00:00:18, Serial0/0.1
R       155.1.9.0 [120/4] via 155.1.0.3, 00:00:18, Serial0/0.1
R       155.1.13.0 [120/2] via 155.1.0.1, 00:00:17, Serial0/0.1
R       155.1.7.0 [120/3] via 155.1.0.3, 00:00:18, Serial0/0.1
R       155.1.5.0 [120/1] via 155.1.0.5, 00:00:18, Serial0/0.1
R       155.1.58.0 [120/1] via 155.1.0.5, 00:00:17, Serial0/0.1
R       155.1.45.0 [120/1] via 155.1.0.5, 00:00:17, Serial0/0.1
R       155.1.37.0 [120/2] via 155.1.0.3, 00:00:17, Serial0/0.1
R       155.1.79.0 [120/3] via 155.1.0.3, 00:00:17, Serial0/0.1
R       155.1.67.0 [120/3] via 155.1.0.1, 00:00:17, Serial0/0.1
R       155.1.108.0 [120/2] via 155.1.0.5, 00:00:17, Serial0/0.1
R    220.20.3.0/24 [120/7] via 192.10.1.254, 00:00:22, FastEthernet0/0
     54.0.0.0/24 is subnetted, 1 subnets
R       54.1.1.0 [120/3] via 155.1.0.1, 00:00:19, Serial0/0.1
R    212.18.1.0/24 [120/4] via 155.1.0.1, 00:00:19, Serial0/0.1
R    212.18.0.0/24 [120/4] via 155.1.0.1, 00:00:19, Serial0/0.1
R    212.18.3.0/24 [120/4] via 155.1.0.1, 00:00:19, Serial0/0.1
R    212.18.2.0/24 [120/4] via 155.1.0.1, 00:00:19, Serial0/0.1
     31.0.0.0/16 is subnetted, 4 subnets
R       31.3.0.0 [120/3] via 155.1.0.4, 00:00:19, Serial0/0.1
R       31.2.0.0 [120/3] via 155.1.0.4, 00:00:19, Serial0/0.1
R       31.1.0.0 [120/3] via 155.1.0.4, 00:00:19, Serial0/0.1
R       31.0.0.0 [120/3] via 155.1.0.4, 00:00:19, Serial0/0.1
     150.1.0.0/24 is subnetted, 10 subnets
R       150.1.7.0 [120/3] via 155.1.0.3, 00:00:19, Serial0/0.1
R       150.1.6.0 [120/3] via 155.1.0.1, 00:00:19, Serial0/0.1
R       150.1.5.0 [120/1] via 155.1.0.5, 00:00:19, Serial0/0.1
```

Copyright © 2010 Internetwork Expert

```
R       150.1.4.0 [120/2] via 155.1.0.4, 00:00:20, Serial0/0.1
R       150.1.3.0 [120/2] via 155.1.0.3, 00:00:20, Serial0/0.1
R       150.1.1.0 [120/2] via 155.1.0.1, 00:00:20, Serial0/0.1
R       150.1.10.0 [120/3] via 155.1.0.5, 00:00:20, Serial0/0.1
R       150.1.9.0 [120/4] via 155.1.0.3, 00:00:20, Serial0/0.1
R       150.1.8.0 [120/2] via 155.1.0.5, 00:00:20, Serial0/0.1
R    205.90.31.0/24 [120/7] via 192.10.1.254, 00:00:23, FastEthernet0/0
     30.0.0.0/16 is subnetted, 4 subnets
R       30.2.0.0 [120/3] via 155.1.0.4, 00:00:20, Serial0/0.1
R       30.3.0.0 [120/3] via 155.1.0.4, 00:00:20, Serial0/0.1
R       30.0.0.0 [120/3] via 155.1.0.4, 00:00:20, Serial0/0.1
R       30.1.0.0 [120/3] via 155.1.0.4, 00:00:20, Serial0/0.1
```

Copyright © 2010 Internetwork Expert

4.4 RIPv2 Auto-Summary

- Enable auto-summary under the RIP process of R4.
- Note any changes in the network advertisements that R4 is sending.

Configuration

```
R4:
router rip
 auto-summary
```

Verification

> **Note**
>
> Before auto-summary is enabled:

```
Rack1R4#debug ip rip
RIP protocol debugging is on
RIP: sending v2 update to 224.0.0.9 via Serial0/1/0 (155.1.45.4)
RIP: build update entries
        30.0.0.0/16 via 0.0.0.0, metric 2, tag 0
        30.1.0.0/16 via 0.0.0.0, metric 2, tag 0
        30.2.0.0/16 via 0.0.0.0, metric 2, tag 0
        30.3.0.0/16 via 0.0.0.0, metric 2, tag 0
        31.0.0.0/16 via 0.0.0.0, metric 2, tag 0
        31.1.0.0/16 via 0.0.0.0, metric 2, tag 0
        31.2.0.0/16 via 0.0.0.0, metric 2, tag 0
        31.3.0.0/16 via 0.0.0.0, metric 2, tag 0
        150.1.4.0/24 via 0.0.0.0, metric 1, tag 0
        155.1.0.0/24 via 0.0.0.0, metric 1, tag 0
        204.12.1.0/24 via 0.0.0.0, metric 1, tag 0
RIP: sending v2 update to 224.0.0.9 via Serial0/0/0.1 (155.1.0.4)
RIP: build update entries
        30.0.0.0/16 via 0.0.0.0, metric 2, tag 0
        30.1.0.0/16 via 0.0.0.0, metric 2, tag 0
        30.2.0.0/16 via 0.0.0.0, metric 2, tag 0
        30.3.0.0/16 via 0.0.0.0, metric 2, tag 0
        31.0.0.0/16 via 0.0.0.0, metric 2, tag 0
        31.1.0.0/16 via 0.0.0.0, metric 2, tag 0
        31.2.0.0/16 via 0.0.0.0, metric 2, tag 0
        31.3.0.0/16 via 0.0.0.0, metric 2, tag 0
        150.1.4.0/24 via 0.0.0.0, metric 1, tag 0
        155.1.45.0/24 via 0.0.0.0, metric 1, tag 0
        204.12.1.0/24 via 0.0.0.0, metric 1, tag 0
```

Copyright © 2010 Internetwork Expert

After auto-summary is enabled R4 summarizes the routes learned from BB3 and its own Loopback 0 network out to R5, and summarizes the 155.1.0.0/16 transit network, its own Loopback 0 network, and the link to BB1 out to BB3.

```
RIP: sending v2 update to 224.0.0.9 via Serial0/1/0 (155.1.45.4)
RIP: build update entries
        30.0.0.0/8 via 0.0.0.0, metric 2, tag 0
        31.0.0.0/8 via 0.0.0.0, metric 2, tag 0
        150.1.0.0/16 via 0.0.0.0, metric 1, tag 0
        155.1.0.0/24 via 0.0.0.0, metric 1, tag 0
        204.12.1.0/24 via 0.0.0.0, metric 1, tag 0
RIP: sending v2 update to 224.0.0.9 via Serial0/0/0.1 (155.1.0.4)
RIP: build update entries
        30.0.0.0/8 via 0.0.0.0, metric 2, tag 0
        31.0.0.0/8 via 0.0.0.0, metric 2, tag 0
        150.1.0.0/16 via 0.0.0.0, metric 1, tag 0
        155.1.45.0/24 via 0.0.0.0, metric 1, tag 0
        204.12.1.0/24 via 0.0.0.0, metric 1, tag 0
RIP: sending v2 update to 224.0.0.9 via FastEthernet0/0 (204.12.1.4)
RIP: build update entries
        54.0.0.0/8 via 0.0.0.0, metric 4, tag 0
        150.1.0.0/16 via 0.0.0.0, metric 1, tag 0
        155.1.0.0/16 via 0.0.0.0, metric 1, tag 0
        192.10.1.0/24 via 0.0.0.0, metric 3, tag 0
        205.90.31.0/24 via 0.0.0.0, metric 10, tag 0
        212.18.0.0/24 via 0.0.0.0, metric 5, tag 0
        212.18.1.0/24 via 0.0.0.0, metric 5, tag 0
        212.18.2.0/24 via 0.0.0.0, metric 5, tag 0
        212.18.3.0/24 via 0.0.0.0, metric 5, tag 0
        220.20.3.0/24 via 0.0.0.0, metric 10, tag 0
        222.22.2.0/24 via 0.0.0.0, metric 10, tag 0
```

Copyright © 2010 Internetwork Expert

4.5 RIP Send and Receive Versions

- Remove the **`version 2`** commands under the RIP processes of SW2 and SW4.
- Configure SW2 to send and receive only RIPv2 updates on VLAN 58.
- Configure SW2 to send and receive only RIPv1 updates on the link to SW4.
- Note any changes in reachability or routing information in the network.

Configuration

```
SW2:
interface Vlan58
 ip rip send version 2
 ip rip receive version 2
!
interface Port-channel1
 ip rip send version 1
 ip rip receive version 1
!
router rip
 network 150.1.0.0
 network 155.1.0.0
 no auto-summary

SW4:
router rip
 network 150.1.0.0
 network 155.1.0.0
 no auto-summary
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> SW2 and SW4's routing tables before the change to RIPv1.

```
Rack1SW2#show ip route rip
R    222.22.2.0/24 [120/9] via 155.1.58.5, 00:00:15, Vlan58
R    204.12.1.0/24 [120/2] via 155.1.58.5, 00:00:15, Vlan58
     155.1.0.0/24 is subnetted, 14 subnets
R       155.1.146.0 [120/2] via 155.1.58.5, 00:00:15, Vlan58
R       155.1.10.0 [120/1] via 155.1.108.10, 00:00:24, Port-channel1
R       155.1.9.0 [120/4] via 155.1.58.5, 00:00:15, Vlan58
R       155.1.13.0 [120/2] via 155.1.58.5, 00:00:15, Vlan58
R       155.1.0.0 [120/1] via 155.1.58.5, 00:00:15, Vlan58
R       155.1.7.0 [120/3] via 155.1.58.5, 00:00:15, Vlan58
R       155.1.5.0 [120/1] via 155.1.58.5, 00:00:15, Vlan58
R       155.1.45.0 [120/1] via 155.1.58.5, 00:00:15, Vlan58
R       155.1.37.0 [120/2] via 155.1.58.5, 00:00:15, Vlan58
R       155.1.79.0 [120/3] via 155.1.58.5, 00:00:15, Vlan58
R       155.1.67.0 [120/3] via 155.1.58.5, 00:00:15, Vlan58
R    220.20.3.0/24 [120/9] via 155.1.58.5, 00:00:15, Vlan58
     54.0.0.0/24 is subnetted, 1 subnets
R       54.1.1.0 [120/3] via 155.1.58.5, 00:00:15, Vlan58
R    212.18.1.0/24 [120/4] via 155.1.58.5, 00:00:15, Vlan58
R    212.18.0.0/24 [120/4] via 155.1.58.5, 00:00:16, Vlan58
R    212.18.3.0/24 [120/4] via 155.1.58.5, 00:00:16, Vlan58
R    212.18.2.0/24 [120/4] via 155.1.58.5, 00:00:16, Vlan58
R    192.10.1.0/24 [120/2] via 155.1.58.5, 00:00:16, Vlan58
     31.0.0.0/8 is subnetted, 1 subnets
R       31.0.0.0 [120/3] via 155.1.58.5, 00:00:16, Vlan58
     150.1.0.0/16 is variably subnetted, 10 subnets, 2 masks
R       150.1.7.0/24 [120/3] via 155.1.58.5, 00:00:16, Vlan58
R       150.1.6.0/24 [120/3] via 155.1.58.5, 00:00:16, Vlan58
R       150.1.5.0/24 [120/1] via 155.1.58.5, 00:00:16, Vlan58
R       150.1.3.0/24 [120/2] via 155.1.58.5, 00:00:16, Vlan58
R       150.1.2.0/24 [120/2] via 155.1.58.5, 00:00:16, Vlan58
R       150.1.1.0/24 [120/2] via 155.1.58.5, 00:00:16, Vlan58
R       150.1.0.0/16 [120/2] via 155.1.58.5, 00:00:16, Vlan58
R       150.1.10.0/24 [120/1] via 155.1.108.10, 00:00:26, Port-channel1
R       150.1.9.0/24 [120/4] via 155.1.58.5, 00:00:16, Vlan58
R    205.90.31.0/24 [120/9] via 155.1.58.5, 00:00:17, Vlan58
     30.0.0.0/8 is subnetted, 1 subnets
R       30.0.0.0 [120/3] via 155.1.58.5, 00:00:17, Vlan58
```

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show ip route rip
R    222.22.2.0/24 [120/10] via 155.1.108.8, 00:00:07, Port-channel1
R    204.12.1.0/24 [120/3] via 155.1.108.8, 00:00:07, Port-channel1
     155.1.0.0/24 is subnetted, 14 subnets
R       155.1.146.0 [120/3] via 155.1.108.8, 00:00:07, Port-channel1
R       155.1.8.0 [120/1] via 155.1.108.8, 00:00:07, Port-channel1
R       155.1.9.0 [120/5] via 155.1.108.8, 00:00:07, Port-channel1
R       155.1.13.0 [120/3] via 155.1.108.8, 00:00:07, Port-channel1
R       155.1.0.0 [120/2] via 155.1.108.8, 00:00:08, Port-channel1
R       155.1.7.0 [120/4] via 155.1.108.8, 00:00:08, Port-channel1
R       155.1.5.0 [120/2] via 155.1.108.8, 00:00:08, Port-channel1
R       155.1.58.0 [120/1] via 155.1.108.8, 00:00:08, Port-channel1
R       155.1.45.0 [120/2] via 155.1.108.8, 00:00:08, Port-channel1
R       155.1.37.0 [120/3] via 155.1.108.8, 00:00:08, Port-channel1
R       155.1.79.0 [120/4] via 155.1.108.8, 00:00:08, Port-channel1
R       155.1.67.0 [120/4] via 155.1.108.8, 00:00:08, Port-channel1
R    220.20.3.0/24 [120/10] via 155.1.108.8, 00:00:08, Port-channel1
     54.0.0.0/24 is subnetted, 1 subnets
R       54.1.1.0 [120/4] via 155.1.108.8, 00:00:08, Port-channel1
R    212.18.1.0/24 [120/5] via 155.1.108.8, 00:00:08, Port-channel1
R    212.18.0.0/24 [120/5] via 155.1.108.8, 00:00:08, Port-channel1
R    212.18.3.0/24 [120/5] via 155.1.108.8, 00:00:08, Port-channel1
R    212.18.2.0/24 [120/5] via 155.1.108.8, 00:00:08, Port-channel1
R    192.10.1.0/24 [120/3] via 155.1.108.8, 00:00:09, Port-channel1
     31.0.0.0/8 is subnetted, 1 subnets
R       31.0.0.0 [120/4] via 155.1.108.8, 00:00:09, Port-channel1
     150.1.0.0/16 is variably subnetted, 10 subnets, 2 masks
R       150.1.7.0/24 [120/4] via 155.1.108.8, 00:00:09, Port-channel1
R       150.1.6.0/24 [120/4] via 155.1.108.8, 00:00:09, Port-channel1
R       150.1.5.0/24 [120/2] via 155.1.108.8, 00:00:09, Port-channel1
R       150.1.3.0/24 [120/3] via 155.1.108.8, 00:00:09, Port-channel1
R       150.1.2.0/24 [120/3] via 155.1.108.8, 00:00:09, Port-channel1
R       150.1.1.0/24 [120/3] via 155.1.108.8, 00:00:09, Port-channel1
R       150.1.0.0/16 [120/3] via 155.1.108.8, 00:00:09, Port-channel1
R       150.1.9.0/24 [120/5] via 155.1.108.8, 00:00:09, Port-channel1
R       150.1.8.0/24 [120/1] via 155.1.108.8, 00:00:09, Port-channel1
R    205.90.31.0/24 [120/10] via 155.1.108.8, 00:00:09, Port-channel1
     30.0.0.0/8 is subnetted, 1 subnets
R       30.0.0.0 [120/4] via 155.1.108.8, 00:00:09, Port-channel1
```

Copyright © 2010 Internetwork Expert

After RIPv1 between SW2 and SW4.

```
Rack1SW2#show ip route rip
R    222.22.2.0/24 [120/9] via 155.1.58.5, 00:00:18, Vlan58
R    204.12.1.0/24 [120/2] via 155.1.58.5, 00:00:18, Vlan58
     155.1.0.0/24 is subnetted, 14 subnets
R       155.1.146.0 [120/2] via 155.1.58.5, 00:00:18, Vlan58
R       155.1.10.0 [120/1] via 155.1.108.10, 00:00:03, Port-channel1
R       155.1.9.0 [120/4] via 155.1.58.5, 00:00:18, Vlan58
R       155.1.13.0 [120/2] via 155.1.58.5, 00:00:18, Vlan58
R       155.1.0.0 [120/1] via 155.1.58.5, 00:00:18, Vlan58
R       155.1.7.0 [120/3] via 155.1.58.5, 00:00:18, Vlan58
R       155.1.5.0 [120/1] via 155.1.58.5, 00:00:19, Vlan58
R       155.1.45.0 [120/1] via 155.1.58.5, 00:00:19, Vlan58
R       155.1.37.0 [120/2] via 155.1.58.5, 00:00:19, Vlan58
R       155.1.79.0 [120/3] via 155.1.58.5, 00:00:19, Vlan58
R       155.1.67.0 [120/3] via 155.1.58.5, 00:00:19, Vlan58
R    220.20.3.0/24 [120/9] via 155.1.58.5, 00:00:19, Vlan58
     54.0.0.0/24 is subnetted, 1 subnets
R       54.1.1.0 [120/3] via 155.1.58.5, 00:00:19, Vlan58
R    212.18.1.0/24 [120/4] via 155.1.58.5, 00:00:19, Vlan58
R    212.18.0.0/24 [120/4] via 155.1.58.5, 00:00:19, Vlan58
R    212.18.3.0/24 [120/4] via 155.1.58.5, 00:00:19, Vlan58
R    212.18.2.0/24 [120/4] via 155.1.58.5, 00:00:19, Vlan58
R    192.10.1.0/24 [120/2] via 155.1.58.5, 00:00:19, Vlan58
R    31.0.0.0/8 [120/3] via 155.1.58.5, 00:00:19, Vlan58
     150.1.0.0/16 is variably subnetted, 9 subnets, 2 masks
R       150.1.7.0/24 [120/3] via 155.1.58.5, 00:00:19, Vlan58
R       150.1.6.0/24 [120/3] via 155.1.58.5, 00:00:19, Vlan58
R       150.1.5.0/24 [120/1] via 155.1.58.5, 00:00:20, Vlan58
R       150.1.3.0/24 [120/2] via 155.1.58.5, 00:00:20, Vlan58
R       150.1.2.0/24 [120/2] via 155.1.58.5, 00:00:20, Vlan58
R       150.1.1.0/24 [120/2] via 155.1.58.5, 00:00:20, Vlan58
R       150.1.0.0/16 [120/2] via 155.1.58.5, 00:00:20, Vlan58
R       150.1.9.0/24 [120/4] via 155.1.58.5, 00:00:20, Vlan58
R    205.90.31.0/24 [120/9] via 155.1.58.5, 00:00:20, Vlan58
R    30.0.0.0/8 [120/3] via 155.1.58.5, 00:00:20, Vlan58
```

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show ip route rip
R    222.22.2.0/24 [120/10] via 155.1.108.8, 00:00:17, Port-channel1
R    204.12.1.0/24 [120/3] via 155.1.108.8, 00:00:17, Port-channel1
     155.1.0.0/24 is subnetted, 14 subnets
R       155.1.146.0 [120/3] via 155.1.108.8, 00:00:17, Port-channel1
R       155.1.8.0 [120/1] via 155.1.108.8, 00:00:17, Port-channel1
R       155.1.9.0 [120/5] via 155.1.108.8, 00:00:17, Port-channel1
R       155.1.13.0 [120/3] via 155.1.108.8, 00:00:17, Port-channel1
R       155.1.0.0 [120/2] via 155.1.108.8, 00:00:17, Port-channel1
R       155.1.7.0 [120/4] via 155.1.108.8, 00:00:17, Port-channel1
R       155.1.5.0 [120/2] via 155.1.108.8, 00:00:17, Port-channel1
R       155.1.58.0 [120/1] via 155.1.108.8, 00:00:17, Port-channel1
R       155.1.45.0 [120/2] via 155.1.108.8, 00:00:17, Port-channel1
R       155.1.37.0 [120/3] via 155.1.108.8, 00:00:17, Port-channel1
R       155.1.79.0 [120/4] via 155.1.108.8, 00:00:17, Port-channel1
R       155.1.67.0 [120/4] via 155.1.108.8, 00:00:17, Port-channel1
R    220.20.3.0/24 [120/10] via 155.1.108.8, 00:00:18, Port-channel1
R    54.0.0.0/8 [120/4] via 155.1.108.8, 00:00:18, Port-channel1
R    212.18.1.0/24 [120/5] via 155.1.108.8, 00:00:18, Port-channel1
R    212.18.0.0/24 [120/5] via 155.1.108.8, 00:00:18, Port-channel1
R    212.18.3.0/24 [120/5] via 155.1.108.8, 00:00:18, Port-channel1
R    212.18.2.0/24 [120/5] via 155.1.108.8, 00:00:18, Port-channel1
R    192.10.1.0/24 [120/3] via 155.1.108.8, 00:00:18, Port-channel1
R    31.0.0.0/8 [120/4] via 155.1.108.8, 00:00:18, Port-channel1
R    205.90.31.0/24 [120/10] via 155.1.108.8, 00:00:18, Port-channel1
R    30.0.0.0/8 [120/4] via 155.1.108.8, 00:00:18, Port-channel1
```

Since SW2 and SW4 are running only RIPv1 between them, which does not include subnet mask information in the update, only classful networks with their classful masks can be advertised, along with contiguous subnets that share the same major network and subnet mask as the transit link between the neighbors. This is the reason why SW4 still learns about all the 155.1.X.0/24 networks, but cannot learn any of the 150.1.X.0/24 Loopback networks. Likewise SW2 cannot learn the 150.1.10.0/24 network from SW4. When traffic is sent to 155.1.10.10 it is routed based on the RIPv2 summary generated by R4. Since R4 does not have a longer match to the 150.1.10.10 destination, traffic is dropped and an ICMP unreachable message is sent back to the originating host. This problem is inherent to the design of RIPv1 and cannot be fixed in this topology without renumbering subnets or running RIPv2.

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show ip route 150.1.0.0
Routing entry for 150.1.0.0/24, 1 known subnets
  Attached (1 connections)
  Redistributing via rip

C       150.1.10.0 is directly connected, Loopback0
```

```
Rack1SW2#debug ip icmp
ICMP packet debugging is on
Rack1SW2#ping 150.1.10.10

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.10.10, timeout is 2 seconds:
U.U.U
Success rate is 0 percent (0/5)
ICMP: dst (155.1.58.8) host unreachable rcv from 155.1.0.4
ICMP: dst (155.1.58.8) host unreachable rcv from 155.1.0.4
ICMP: dst (155.1.58.8) host unreachable rcv from 155.1.0.4
```

```
Rack1SW4#debug ip packet
IP packet debugging is on
Rack1SW4#ping 150.1.8.8

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.8.8, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)
IP: s=150.1.10.10 (local), d=150.1.8.8, len 100, unroutable.
IP: s=150.1.10.10 (local), d=150.1.8.8, len 100, unroutable.
IP: s=150.1.10.10 (local), d=150.1.8.8, len 100, unroutable
IP: s=150.1.10.10 (local), d=150.1.8.8, len 100, unroutable
IP: s=150.1.10.10 (local), d=150.1.8.8, len 100, unroutable
```

While RIPv1 will not be tested on in the CCIE Lab Exam, understanding the problems with legacy protocol design can assist you in understanding the “why” behind routing logic for IPv4.

Copyright © 2010 Internetwork Expert

4.6 RIPv2 Manual Summarization

- Configure R4 to send two summary routes for the RIP networks learned from BB3 to R5.
- Ensure that these summaries do not overlap any address space that R4 does not have a longer match route to.

Configuration

```
R4:
interface Serial0/0/0.1 point-to-point
 ip summary-address rip 30.0.0.0 255.252.0.0
 ip summary-address rip 31.0.0.0 255.252.0.0
!
interface Serial0/1/0
 ip summary-address rip 30.0.0.0 255.252.0.0
 ip summary-address rip 31.0.0.0 255.252.0.0
!
router rip
 no auto-summary
```

Verification

```
Rack1R5#show ip route 30.0.0.0
Routing entry for 30.0.0.0/14, 1 known subnets
  Redistributing via rip

R       30.0.0.0 [120/2] via 155.1.45.4, 00:00:04, Serial0/1/0
                 [120/2] via 155.1.0.4, 00:00:00, Serial0/0/0.1

Rack1R5#show ip route 31.0.0.0
Routing entry for 31.0.0.0/14, 1 known subnets
  Redistributing via rip

R       31.0.0.0 [120/2] via 155.1.45.4, 00:00:11, Serial0/1/0
                 [120/2] via 155.1.0.4, 00:00:06, Serial0/0/0.1
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

4.7 RIPv2 Convergence Timers

- Change the RIP timers throughout the topology to make convergence three times faster than the default.
- Ensure that this configuration does not affect the links to the BB routers.

Configuration

```
R1:
router rip
 version 2
 timers basic 10 60 60 80

R2:
interface FastEthernet0/0
 ip rip advertise 30
!
router rip
 version 2
 timers basic 10 60 60 80

R3:
router rip
 version 2
 timers basic 10 60 60 80

R4:
interface FastEthernet0/0
 ip rip advertise 30
!
router rip
 version 2
 timers basic 10 60 60 80

R5:
router rip
 version 2
 timers basic 10 60 60 80

R6:
interface Serial0/0/0
 ip rip advertise 30
!
router rip
 version 2
 timers basic 10 60 60 80

SW1:
router rip
 version 2
 timers basic 10 60 60 80
```

Copyright © 2010 Internetwork Expert

```
SW2:
router rip
 version 2
 timers basic 10 60 60 80

SW3:
router rip
 version 2
 timers basic 10 60 60 80

SW4:
router rip
 version 2
 timers basic 10 60 60 80
```

Verification

> **Note**
>
> Prior to changing RIP timers.

```
Rack1R1#show ip protocols | include seconds
  Sending updates every 30 seconds, next due in 22 seconds
  Invalid after 180 seconds, hold down 180, flushed after 240
```

After changing RIP timers:

```
Rack1R1#show ip protocols | include seconds
  Sending updates every 10 seconds, next due in 9 seconds
  Invalid after 60 seconds, hold down 60, flushed after 80
```

Copyright © 2010 Internetwork Expert

4.8 RIPv2 Offset List

- Configure an offset-list on R6 so that all traffic going to BB1 uses the Ethernet link to SW1.
- If this link is down traffic should be rerouted over the link to R1.

Configuration

```
R6:
router rip
 offset-list 0 out 5 FastEthernet0/0.146
```

Verification

> **Note**
>
> Before metric offset.

```
Rack1R2#traceroute 212.18.0.1

Type escape sequence to abort.
Tracing the route to 212.18.0.1

  1 155.1.0.5 28 msec 28 msec 28 msec
  2 155.1.0.1 64 msec 56 msec 56 msec
  3 155.1.146.6 56 msec 57 msec 56 msec
  4 54.1.1.254 76 msec *  76 msec
```

After metric offset,

```
Rack1R2#traceroute 212.18.0.1

Type escape sequence to abort.
Tracing the route to 212.18.0.1

  1 155.1.0.5 28 msec 32 msec 28 msec
  2 155.1.0.3 64 msec 56 msec 56 msec
  3 155.1.37.7 56 msec 56 msec 60 msec
  4 155.1.67.6 56 msec 57 msec 56 msec
  5 54.1.1.254 72 msec *  72 msec
```

Copyright © 2010 Internetwork Expert

Traffic should reroute to R1 if SW1 fails.

```
Rack1SW1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface Vlan67
Rack1SW1(config-if)#shut
Rack1SW1(config-if)#end
Rack1SW1#

Rack1R2#traceroute 212.18.0.1

Type escape sequence to abort.
Tracing the route to 212.18.0.1

  1 155.1.0.5 28 msec 28 msec 28 msec
  2 155.1.0.1 68 msec 56 msec 56 msec
  3 155.1.146.6 56 msec 56 msec 56 msec
  4 54.1.1.254 76 msec *  72 msec
```

Copyright © 2010 Internetwork Expert

4.9 RIPv2 Filtering with Passive Interface

- Configure the passive interface feature on SW2 so that it learns RIP updates from SW4 but does not advertise any information back to SW4.

Configuration

```
SW2:
router rip
 passive-interface Port-channel1
```

Verification

```
Rack1SW2#show ip route | include Port-channel1
R       155.1.10.0 [120/1] via 155.1.108.10, 00:00:07, Port-channel1
C       155.1.108.0 is directly connected, Port-channel1

Rack1SW4#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     155.1.0.0/24 is subnetted, 2 subnets
C       155.1.10.0 is directly connected, Vlan10
C       155.1.108.0 is directly connected, Port-channel1
     150.1.0.0/24 is subnetted, 1 subnets
C       150.1.10.0 is directly connected, Loopback0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

4.10 RIPv2 Filtering with Prefix-Lists

- Configure a prefix-list on R5 so that it does not advertise the two RIP summaries of the BB3 networks to SW2, and so that all other networks are allowed to be advertised.
- Configure a prefix-list on R5 so that it does not install any updates received from R4 on the Frame Relay network. This configuration should not affect the updates received from other neighbors on this segment.
- Ensure that route feedback does not occur for the summaries that R4 is generating to R5.

Configuration

```
R4:
ip route 30.0.0.0 255.252.0.0 Null0
ip route 31.0.0.0 255.252.0.0 Null0

R5:
router rip
 distribute-list prefix RIP_FILTER_TO_SW2 out FastEthernet0/0
 distribute-list prefix PERMIT_ALL gateway NOT_FROM_R4 in
!
ip prefix-list NOT_FROM_R4 seq 5 deny 155.1.0.4/32
ip prefix-list NOT_FROM_R4 seq 10 permit 0.0.0.0/0 le 32
!
ip prefix-list PERMIT_ALL seq 5 permit 0.0.0.0/0 le 32
!
ip prefix-list RIP_FILTER_TO_SW2 seq 5 deny 30.0.0.0/14
ip prefix-list RIP_FILTER_TO_SW2 seq 10 deny 31.0.0.0/14
ip prefix-list RIP_FILTER_TO_SW2 seq 15 permit 0.0.0.0/0 le 32
```

Verification

```
Rack1R5#show ip route rip
R    222.22.2.0/24 [120/8] via 155.1.0.2, 00:00:05, Serial0/0/0.1
R    204.12.1.0/24 [120/1] via 155.1.45.4, 00:00:01, Serial0/1/0
     155.1.0.0/24 is subnetted, 14 subnets
R       155.1.146.0 [120/1] via 155.1.0.1, 00:00:02, Serial0/0/0.1
R       155.1.10.0 [120/2] via 155.1.58.8, 00:00:02, FastEthernet0/0
R       155.1.8.0 [120/1] via 155.1.58.8, 00:00:02, FastEthernet0/0
R       155.1.9.0 [120/3] via 155.1.0.3, 00:00:08, Serial0/0/0.1
R       155.1.13.0 [120/1] via 155.1.0.3, 00:00:08, Serial0/0/0.1
                   [120/1] via 155.1.0.1, 00:00:02, Serial0/0/0.1
R       155.1.7.0 [120/2] via 155.1.0.3, 00:00:08, Serial0/0/0.1
R       155.1.37.0 [120/1] via 155.1.0.3, 00:00:08, Serial0/0/0.1
R       155.1.79.0 [120/2] via 155.1.0.3, 00:00:08, Serial0/0/0.1
R       155.1.67.0 [120/2] via 155.1.0.3, 00:00:08, Serial0/0/0.1
R       155.1.108.0 [120/1] via 155.1.58.8, 00:00:02, FastEthernet0/0
R    220.20.3.0/24 [120/8] via 155.1.0.2, 00:00:05, Serial0/0/0.1
     54.0.0.0/24 is subnetted, 1 subnets
R       54.1.1.0 [120/3] via 155.1.0.3, 00:00:08, Serial0/0/0.1
R    212.18.1.0/24 [120/4] via 155.1.0.3, 00:00:08, Serial0/0/0.1
R    212.18.0.0/24 [120/4] via 155.1.0.3, 00:00:08, Serial0/0/0.1
```

Copyright © 2010 Internetwork Expert

```
R    212.18.3.0/24 [120/4] via 155.1.0.3, 00:00:08, Serial0/0/0.1
R    212.18.2.0/24 [120/4] via 155.1.0.3, 00:00:08, Serial0/0/0.1
R    192.10.1.0/24 [120/1] via 155.1.0.2, 00:00:05, Serial0/0/0.1
     31.0.0.0/14 is subnetted, 1 subnets
R       31.0.0.0 [120/2] via 155.1.45.4, 00:00:01, Serial0/1/0
     150.1.0.0/24 is subnetted, 9 subnets
R       150.1.7.0 [120/2] via 155.1.0.3, 00:00:08, Serial0/0/0.1
R       150.1.6.0 [120/3] via 155.1.0.3, 00:00:08, Serial0/0/0.1
R       150.1.4.0 [120/1] via 155.1.45.4, 00:00:01, Serial0/1/0
R       150.1.3.0 [120/1] via 155.1.0.3, 00:00:08, Serial0/0/0.1
R       150.1.2.0 [120/1] via 155.1.0.2, 00:00:05, Serial0/0/0.1
R       150.1.1.0 [120/1] via 155.1.0.1, 00:00:02, Serial0/0/0.1
R       150.1.9.0 [120/3] via 155.1.0.3, 00:00:08, Serial0/0/0.1
R       150.1.8.0 [120/1] via 155.1.58.8, 00:00:03, FastEthernet0/0
R    205.90.31.0/24 [120/8] via 155.1.0.2, 00:00:05, Serial0/0/0.1
     30.0.0.0/14 is subnetted, 1 subnets
R       30.0.0.0 [120/2] via 155.1.45.4, 00:00:01, Serial0/1/0

Rack1SW2#show ip route 30.0.0.0
% Network not in table

Rack1SW2#show ip route 31.0.0.0
% Network not in table

Rack1R2#show ip route 30.0.0.0
Routing entry for 30.0.0.0/14, 1 known subnets
  Redistributing via rip

R       30.0.0.0 [120/3] via 155.1.0.5, 00:00:04, Serial0/0.1

Rack1R2#show ip route 31.0.0.0
Routing entry for 31.0.0.0/14, 1 known subnets
  Redistributing via rip

R       31.0.0.0 [120/3] via 155.1.0.5, 00:00:08, Serial0/0.1
```

Note

The first filter on R5, out to SW2, denies the exact two prefixes that are the summaries coming from R4, and permits all other routes. The syntax **`0.0.0.0/0 le 32`** in a prefix-list means match all routes. The next filter is done based on both the routes learned and who they are learned from. This filter says match any route coming in any interface, per the PERMIT_ALL list, and allow them to come in as long as they were not learned from R4, per the **`deny 155.1.0.4/32`** syntax.

Copyright © 2010 Internetwork Expert

Pitfall

Routing filters introduce the possibility of routing loops due to route feedback, or traffic black holes due to incomplete routing information. In this particular scenario route feedback is introduced on the segments between R4 and R5 with the two summaries R5 learns from R4.

Without our filtering configuration R5 learns the 30.0.0.0/14 and 31.0.0.0/14 summaries from both links to R4. Since split-horizon is enabled on the point-to-point link to R4, this information is not sent back out this link. However since split-horizon was disabled on the Frame Relay link these two routes are sent back out this link. When this update is generated, the next-hop value of the originating router on the subnet is used. In this case the value of the next-hop field is 155.1.0.4 for R4. This can be seen from the routing output on R1, R2, or R3:

```
Rack1R1#show ip route 30.0.0.1
Routing entry for 30.0.0.0/14
  Known via "rip", distance 120, metric 3
  Redistributing via rip
  Last update from 155.1.0.4 on Serial0/0.1, 00:00:00 ago
  Routing Descriptor Blocks:
  * 155.1.0.4, from 155.1.0.5, 00:00:00 ago, via Serial0/0.1
      Route metric is 3, traffic share count is 1
```

When R4 receives this update back in is it discarded, because its own local IP address is the next-hop value in the update. The problem in this scenario comes in when the update R5 receives from R4 in the Frame Relay link is filtered out. In this case, R5 receives two updates, one from 155.1.0.4 and one from 155.1.45.4. The update from 155.1.0.4 is filtered out, per the distribute-list gateway filter, and the route via 155.1.45.4 is installed. This route is then sent out the Frame Relay link to all neighbors on the segment, with R5 set as the next-hop.

When R4 receives this update back in the Frame Relay link it is installed as a valid route in the routing table. This is due to the fact that **RIP does not generate a route to Null0 when generating summaries.** Shortly after being installed R4 realizes that the route is not valid, since it is locally generating the summary, and the route is poisoned. This process results in an infinite loop of the route being advertised and withdrawn. The result of this can be seen on R2 as a periodic traffic drop.

Copyright © 2010 Internetwork Expert

```
Rack1R2#ping 30.0.0.1 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:
..!!!!!U.....!!!!!U.U....!U.
Success rate is 39 percent (11/28), round-trip min/avg/max = 100/102/105 ms
```

The step by step troubleshooting and verification of this problem is as follows. Note that other interfaces in the topology connecting to R4 and R5 have been disabled to simplify debug output, NTP is configured, and timestamping to the millisecond has been enabled to correlate events on both devices simultaneously.

To start, R4 sends the initial update containing 30.0.0.0/14 and 31.0.0.0/14 to R5.

```
Rack1R4#
Jun 26 13:44:01.349: RIP: sending v2 update to 224.0.0.9 via Serial0/1/0
(155.1.45.4)
Jun 26 13:44:01.349: RIP: build update entries
Jun 26 13:44:01.349:    30.0.0.0/14 via 0.0.0.0, metric 2, tag 0
Jun 26 13:44:01.349:    31.0.0.0/14 via 0.0.0.0, metric 2, tag 0
Jun 26 13:44:01.349:    150.1.4.0/24 via 0.0.0.0, metric 1, tag 0
Jun 26 13:44:01.349:    155.1.0.0/24 via 0.0.0.0, metric 1, tag 0
Jun 26 13:44:01.353:    204.12.1.0/24 via 0.0.0.0, metric 1, tag 0
```

R5 receives this update and installs it in the RIP database and the routing table.

```
Rack1R5#
Jun 26 13:44:01.373: RIP: received v2 update from 155.1.45.4 on Serial0/1/0
Jun 26 13:44:01.373:      30.0.0.0/14 via 0.0.0.0 in 2 hops
Jun 26 13:44:01.373: RT: SET_LAST_RDB for 30.0.0.0/14
  NEW rdb: via 155.1.45.4

Jun 26 13:44:01.373: RT: add 30.0.0.0/14 via 155.1.45.4, rip metric [120/2]
Jun 26 13:44:01.373: RT: NET-RED 30.0.0.0/14
Jun 26 13:44:01.373: RIP-DB: network_update with 30.0.0.0/14 succeeds
Jun 26 13:44:01.377: RIP-DB: adding 30.0.0.0/14 (metric 2) via 155.1.45.4 on
Serial0/1 to RIP database
Jun 26 13:44:01.377: RIP-DB: add 30.0.0.0/14 (metric 2) via 155.1.45.4 on
Serial0/1/0
Jun 26 13:44:01.377:      31.0.0.0/14 via 0.0.0.0 in 2 hops
Jun 26 13:44:01.377: RT: SET_LAST_RDB for 31.0.0.0/14
  NEW rdb: via 155.1.45.4

Jun 26 13:44:01.381: RT: add 31.0.0.0/14 via 155.1.45.4, rip metric [120/2]
Jun 26 13:44:01.381: RT: NET-RED 31.0.0.0/14
Jun 26 13:44:01.381: RIP-DB: network_update with 31.0.0.0/14 succeeds
Jun 26 13:44:01.381: RIP-DB: adding 31.0.0.0/14 (metric 2) via 155.1.45.4 on
Serial0/1/0 to RIP database
Jun 26 13:44:01.381: RIP-DB: add 31.0.0.0/14 (metric 2) via 155.1.45.4 on
Serial0/1/0
Jun 26 13:44:01.381:      150.1.4.0/24 via 0.0.0.0 in 1 hops
```

Copyright © 2010 Internetwork Expert

R5 takes the update from Serial0/1/0 and sends it out Serial0/0/0.1

```
Rack1R5#
Jun 26 13:44:02.026: RIP: sending v2 update to 224.0.0.9 via Serial0/0/0.1
(155.1.0.5)
Jun 26 13:44:02.026: RIP: build update entries
Jun 26 13:44:02.026:    30.0.0.0/14 via 0.0.0.0, metric 3, tag 0
Jun 26 13:44:02.026:    31.0.0.0/14 via 0.0.0.0, metric 3, tag 0
Jun 26 13:44:02.026:    150.1.4.0/24 via 0.0.0.0, metric 2, tag 0
Jun 26 13:44:02.026:    155.1.0.0/24 via 0.0.0.0, metric 1, tag 0
Jun 26 13:44:02.030:    155.1.45.0/24 via 0.0.0.0, metric 1, tag 0
Jun 26 13:44:02.030:    204.12.1.0/24 via 0.0.0.0, metric 2, tag 0
```

R4 receives the update on Serial0/0/0.1 from R5. Since there is no Null0 route for the summary R4 mistakenly installs its own origination.

```
Rack1R4#
Jun 26 13:44:02.118: RIP: received v2 update from 155.1.0.5 on Serial0/0/0.1
Jun 26 13:44:02.118:      30.0.0.0/14 via 0.0.0.0 in 3 hops
Jun 26 13:44:02.118: RT: network 30.0.0.0 is now variably masked
Jun 26 13:44:02.118: RT: SET_LAST_RDB for 30.0.0.0/14
  NEW rdb: via 155.1.0.5

Jun 26 13:44:02.118: RT: add 30.0.0.0/14 via 155.1.0.5, rip metric [120/3]
Jun 26 13:44:02.122: RT: NET-RED 30.0.0.0/14
Jun 26 13:44:02.122: RIP-DB: network_update with 30.0.0.0/14 succeeds
Jun 26 13:44:02.122: RIP-DB: adding 30.0.0.0/14 (metric 3) via 155.1.0.5 on
Serial0/0/0.1 to RIP database
Jun 26 13:44:02.122: RIP-DB: add 30.0.0.0/14 (metric 3) via 155.1.0.5 on
Serial0/0/0.1
Jun 26 13:44:02.122:      31.0.0.0/14 via 0.0.0.0 in 3 hops
Jun 26 13:44:02.126: RT: network 31.0.0.0 is now variably masked
Jun 26 13:44:02.126: RT: SET_LAST_RDB for 31.0.0.0/14
  NEW rdb: via 155.1.0.5

Jun 26 13:44:02.126: RT: add 31.0.0.0/14 via 155.1.0.5, rip metric [120/3]
Jun 26 13:44:02.126: RT: NET-RED 31.0.0.0/14
Jun 26 13:44:02.126: RIP-DB: network_update with 31.0.0.0/14 succeeds
Jun 26 13:44:02.130: RIP-DB: adding 31.0.0.0/14 (metric 3) via 155.1.0.5 on
Serial0/0/0.1 to RIP database
Jun 26 13:44:02.130: RIP-DB: add 31.0.0.0/14 (metric 3) via 155.1.0.5 on
Serial0/0/0.1
```

Two seconds later R4 realizes that the route is not valid, and poisons it back to R5 with a metric of 16.

```
Jun 26 13:44:04.125: RIP: sending v2 flash update to 224.0.0.9 via
Serial0/0/0.1 (155.1.0.4)
Jun 26 13:44:04.130: RIP: build flash update entries
Jun 26 13:44:04.130:    30.0.0.0/14 via 155.1.0.5, metric 16, tag 0
Jun 26 13:44:04.130:    31.0.0.0/14 via 155.1.0.5, metric 16, tag 0
Jun 26 13:44:04.130: RIP: sending v2 flash update to 224.0.0.9 via Serial0/1/0
(155.1.45.4)
Jun 26 13:44:04.130: RIP: build flash update entries
Jun 26 13:44:04.130:    30.0.0.0/14 via 0.0.0.0, metric 16, tag 0
Jun 26 13:44:04.134:    31.0.0.0/14 via 0.0.0.0, metric 16, tag 0
```

Copyright © 2010 Internetwork Expert

R5 receives the invalid route from R4, withdraws the route from the routing table and database, and poisons the route back to R4.

```
Jun 26 13:44:04.149: RIP: received v2 update from 155.1.45.4 on Serial0/1/0
Jun 26 13:44:04.149:      30.0.0.0/14 via 0.0.0.0 in 16 hops  (inaccessible)
Jun 26 13:44:04.149: RT: del 30.0.0.0/14 via 155.1.45.4, rip metric [120/2]
Jun 26 13:44:04.149: RT: delete subnet route to 30.0.0.0/14
Jun 26 13:44:04.149: RT: NET-RED 30.0.0.0/14
Jun 26 13:44:04.153: RT: delete network route to 30.0.0.0
Jun 26 13:44:04.153: RT: NET-RED 30.0.0.0/8
Jun 26 13:44:04.153: RIP-DB: Remove 30.0.0.0/14, (metric 4294967295) via
155.1.45.4, Serial0/1/0
Jun 26 13:44:04.153:      31.0.0.0/14 via 0.0.0.0 in 16 hops  (inaccessible)
Jun 26 13:44:04.153: RT: del 31.0.0.0/14 via 155.1.45.4, rip metric [120/2]
Jun 26 13:44:04.157: RT: delete subnet route to 31.0.0.0/14
Jun 26 13:44:04.157: RT: NET-RED 31.0.0.0/14
Jun 26 13:44:04.157: RT: delete network route to 31.0.0.0
Jun 26 13:44:04.157: RT: NET-RED 31.0.0.0/8
Jun 26 13:44:04.157: RIP-DB: Remove 31.0.0.0/14, (metric 4294967295) via
155.1.45.4, Serial0/1/0
Jun 26 13:44:04.217: RIP: received v2 update from 155.1.0.4 on Serial0/0/0.1
Jun 26 13:44:06.156: RIP: sending v2 flash update to 224.0.0.9 via
Serial0/0/0.1 (155.1.0.5)
Jun 26 13:44:06.156: RIP: build flash update entries
Jun 26 13:44:06.156:    30.0.0.0/14 via 0.0.0.0, metric 16, tag 0
Jun 26 13:44:06.156:    31.0.0.0/14 via 0.0.0.0, metric 16, tag 0
Jun 26 13:44:06.156: RIP: sending v2 flash update to 224.0.0.9 via Serial0/1/0
(155.1.45.5)
Jun 26 13:44:06.156: RIP: build flash update entries
Jun 26 13:44:06.160:    30.0.0.0/14 via 0.0.0.0, metric 16, tag 0
Jun 26 13:44:06.160:    31.0.0.0/14 via 0.0.0.0, metric 16, tag 0
```

R4 receives the invalid route back from R5 again, and likewise withdraws the route from the routing table and database. The process then continues indefinitely.

```
Jun 26 13:44:06.173: RIP: received v2 update from 155.1.45.5 on Serial0/1/0
Jun 26 13:44:06.177:      30.0.0.0/14 via 0.0.0.0 in 16 hops  (inaccessible)
Jun 26 13:44:06.177:      31.0.0.0/14 via 0.0.0.0 in 16 hops  (inaccessible)
Jun 26 13:44:06.281: RIP: received v2 update from 155.1.0.5 on Serial0/0/0.1
Jun 26 13:44:06.285:      30.0.0.0/14 via 0.0.0.0 in 16 hops  (inaccessible)
Jun 26 13:44:06.285: RT: del 30.0.0.0/14 via 155.1.0.5, rip metric [120/3]
Jun 26 13:44:06.285: RT: delete subnet route to 30.0.0.0/14
Jun 26 13:44:06.285: RT: NET-RED 30.0.0.0/14
Jun 26 13:44:06.285: RIP-DB: Remove 30.0.0.0/14, (metric 4294967295) via
155.1.0.5, Serial0/0/0.1
Jun 26 13:44:06.289:      31.0.0.0/14 via 0.0.0.0 in 16 hops  (inaccessible)
Jun 26 13:44:06.289: RT: del 31.0.0.0/14 via 155.1.0.5, rip metric [120/3]
Jun 26 13:44:06.289: RT: delete subnet route to 31.0.0.0/14
Jun 26 13:44:06.289: RT: NET-RED 31.0.0.0/14
Jun 26 13:44:06.289: RIP-DB: Remove 31.0.0.0/14, (metric 4294967295) via
155.1.0.5, Serial0/0/0.1
```

Copyright © 2010 Internetwork Expert

The fix for this problem is to ensure that R4 does not install its own summary back into the routing table. OSPF, EIGRP, and BGP already prevent this problem by design through the usage of a route to Null0 being automatically generated every time a summary is created. Multiple solutions are valid to solve this problem, as long as R4 does not install the route.

In the above solution the fix is to configure static routes to Null0, mimicking the behavior of the other protocols.

Another solution would be to configure distribute-list filtering inbound on R4 to prevent the route from being installed.

Another valid solution would be to offset the route to 16 out on R5, or enable split-horizon, however these two specific solutions would stop the routes from being advertised to the rest of the neighbors on the Frame Relay cloud.

```
Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#ip route 30.0.0.0 255.252.0.0 Null0
Rack1R4(config)#ip route 31.0.0.0 255.252.0.0 Null0
Rack1R4(config)#end
Rack1R4#

Rack1R2#ping 30.0.0.1 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max =
100/112/341 ms
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

4.11 RIPv2 Filtering with Standard Access-Lists

- Configure a one line standard access-list on R6 to filter out all routes coming from BB1 that have an even number in the third octet.

Configuration

```
R6:
router rip
 distribute-list 1 in Serial0/0/0
!
access-list 1 permit 0.0.1.0 255.255.254.255
```

Verification

```
Rack1R6#show ip route | include Serial0/0/0
C       54.1.1.0 is directly connected, Serial0/0/0
R    212.18.1.0/24 [120/1] via 54.1.1.254, 00:00:24, Serial0/0/0
R    212.18.3.0/24 [120/1] via 54.1.1.254, 00:00:24, Serial0/0/0
```

4.12 RIPv2 Filtering with Extended Access-Lists

- Remove the previous prefix-list filters on R5.
- Configure an extended access-list filter on R5 so that the routes to VLANs 7 and 9 are only received from R1, while the routes to R1's Loopback and VLAN 146 are only received from R3.
- This filter should not affect any other updates on this segment.

Configuration

```
R5:
router rip
 distribute-list 100 in Serial0/0/0.1
 no distribute-list 1 in Serial0/0/0
!
no access-list 1 permit 0.0.1.0 255.255.254.255
!
access-list 100 deny   ip host 155.1.0.3 host 155.1.7.0
access-list 100 deny   ip host 155.1.0.3 host 155.1.9.0
access-list 100 deny   ip host 155.1.0.1 host 155.1.146.0
access-list 100 deny   ip host 155.1.0.1 host 150.1.1.0
access-list 100 permit ip any any
```

Copyright © 2010 Internetwork Expert

Verification

Note

Extended access-lists when called as a distribute-list in IGP have a different meaning than in redistribution or as in BGP. With BGP and redistribution the "source" field in the ACL represents the network address, and the "destination" field represents the subnet mask. In an IGP distribute-list application the "source" field in the ACL matches the update source of the route, and the "destination" field represents the network address. This implementation allows us to control which networks we are receiving, but more importantly who we are receiving them from. Before the filter is applied, R5 routes to R3 for VLANs 7 and 9, and to R1 for VLAN 146 and R1's Loopback 0.

```
Rack1R5#show ip route rip | include via 155.1.0.(1|3)
R       155.1.146.0 [120/1] via 155.1.0.1, 00:00:04, Serial0/0/0.1
R       155.1.9.0 [120/3] via 155.1.0.3, 00:00:04, Serial0/0/0.1
R       155.1.13.0 [120/1] via 155.1.0.3, 00:00:04, Serial0/0/0.1
                   [120/1] via 155.1.0.1, 00:00:04, Serial0/0/0.1
R       155.1.7.0 [120/2] via 155.1.0.3, 00:00:04, Serial0/0/0.1
R       155.1.37.0 [120/1] via 155.1.0.3, 00:00:04, Serial0/0/0.1
R       155.1.79.0 [120/2] via 155.1.0.3, 00:00:04, Serial0/0/0.1
R       155.1.67.0 [120/2] via 155.1.0.3, 00:00:04, Serial0/0/0.1
R       54.1.1.0 [120/3] via 155.1.0.3, 00:00:04, Serial0/0/0.1
R    212.18.1.0/24 [120/4] via 155.1.0.3, 00:00:04, Serial0/0/0.1
R    212.18.3.0/24 [120/4] via 155.1.0.3, 00:00:04, Serial0/0/0.1
R       150.1.7.0 [120/2] via 155.1.0.3, 00:00:04, Serial0/0/0.1
R       150.1.6.0 [120/3] via 155.1.0.3, 00:00:04, Serial0/0/0.1
R       150.1.3.0 [120/1] via 155.1.0.3, 00:00:04, Serial0/0/0.1
R       150.1.1.0 [120/1] via 155.1.0.1, 00:00:04, Serial0/0/0.1
R       150.1.9.0 [120/3] via 155.1.0.3, 00:00:04, Serial0/0/0.1
```

Copyright © 2010 Internetwork Expert

After the filter is applied, R5 routes to R3 for VLANs 7 and 9, and to R1 for VLAN 146 and R1's Loopback0.

```
Rack1R5#show ip route rip | include via 155.1.0.(1|3)
R       155.1.146.0 [120/2] via 155.1.0.3, 00:00:02, Serial0/0/0.1
R       155.1.9.0 [120/4] via 155.1.0.1, 00:00:02, Serial0/0/0.1
R       155.1.13.0 [120/1] via 155.1.0.3, 00:00:02, Serial0/0/0.1
                   [120/1] via 155.1.0.1, 00:00:02, Serial0/0/0.1
R       155.1.7.0 [120/3] via 155.1.0.1, 00:00:02, Serial0/0/0.1
R       155.1.37.0 [120/1] via 155.1.0.3, 00:00:02, Serial0/0/0.1
R       155.1.79.0 [120/2] via 155.1.0.3, 00:00:02, Serial0/0/0.1
R       155.1.67.0 [120/2] via 155.1.0.3, 00:00:02, Serial0/0/0.1
R       54.1.1.0 [120/3] via 155.1.0.3, 00:00:02, Serial0/0/0.1
R    212.18.1.0/24 [120/4] via 155.1.0.3, 00:00:02, Serial0/0/0.1
R    212.18.3.0/24 [120/4] via 155.1.0.3, 00:00:02, Serial0/0/0.1
R       150.1.7.0 [120/2] via 155.1.0.3, 00:00:02, Serial0/0/0.1
R       150.1.6.0 [120/3] via 155.1.0.3, 00:00:02, Serial0/0/0.1
R       150.1.3.0 [120/1] via 155.1.0.3, 00:00:02, Serial0/0/0.1
R       150.1.1.0 [120/2] via 155.1.0.3, 00:00:02, Serial0/0/0.1
R       150.1.9.0 [120/3] via 155.1.0.3, 00:00:02, Serial0/0/0.1
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

4.13 RIPv2 Filtering with Offset Lists

- Configure an offset-list on SW1 so that SW3 does not install a route to VLAN 5.
- This filter should not affect any other updates on this segment.

Configuration

```
SW1:
router rip
 offset-list 1 out 16 Vlan79
!
access-list 1 permit 155.1.5.0
```

Verification

> **Note**
>
> Before offset-list is applied.

```
Rack1SW3#show ip route 155.1.5.0
Routing entry for 155.1.5.0/24
  Known via "rip", distance 120, metric 3
  Redistributing via rip
  Last update from 155.1.79.7 on Vlan79, 00:00:05 ago
  Routing Descriptor Blocks:
  * 155.1.79.7, from 155.1.79.7, 00:00:05 ago, via Vlan79
      Route metric is 3, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

After offset-list is applied.

```
Rack1SW3#show ip route 155.1.5.0
% Subnet not in table

Rack1SW3#debug ip rip
RIP protocol debugging is on
Rack1SW3#
RIP: received v2 update from 155.1.79.7 on Vlan79
     30.0.0.0/14 via 0.0.0.0 in 5 hops
     31.0.0.0/14 via 0.0.0.0 in 5 hops
     54.1.1.0/24 via 0.0.0.0 in 2 hops
     150.1.1.0/24 via 0.0.0.0 in 3 hops
     150.1.2.0/24 via 0.0.0.0 in 4 hops
     150.1.3.0/24 via 0.0.0.0 in 2 hops
     150.1.4.0/24 via 0.0.0.0 in 4 hops
     150.1.5.0/24 via 0.0.0.0 in 3 hops
     150.1.6.0/24 via 0.0.0.0 in 2 hops
     150.1.7.0/24 via 0.0.0.0 in 1 hops
     150.1.8.0/24 via 0.0.0.0 in 4 hops
     155.1.0.0/24 via 0.0.0.0 in 2 hops
     155.1.5.0/24 via 0.0.0.0 in 16 hops  (inaccessible)
     155.1.7.0/24 via 0.0.0.0 in 1 hops
     155.1.8.0/24 via 0.0.0.0 in 4 hops
     155.1.10.0/24 via 0.0.0.0 in 5 hops
     155.1.13.0/24 via 0.0.0.0 in 2 hops
     155.1.37.0/24 via 0.0.0.0 in 1 hops
     155.1.45.0/24 via 0.0.0.0 in 3 hops
     155.1.58.0/24 via 0.0.0.0 in 3 hops
     155.1.67.0/24 via 0.0.0.0 in 1 hops
     155.1.108.0/24 via 0.0.0.0 in 4 hops
     155.1.146.0/24 via 0.0.0.0 in 2 hops
```

Copyright © 2010 Internetwork Expert

4.14 RIPv2 Filtering with Administrative Distance

- Configure administrative distance filtering on R5 so that devices within the network cannot reach R4's Loopback 0 network.
- This filter should not affect any other networks in the topology.

Configuration

```
R5:
router rip
 distance 255 0.0.0.0 255.255.255.255 1
!
access-list 1 permit 150.1.4.0
```

Verification

> **Note**
>
> Before distance is applied.

```
Rack1R5#show ip route 150.1.4.0
Routing entry for 150.1.4.0/24
  Known via "rip", distance 120, metric 1
  Redistributing via rip
  Last update from 155.1.45.4 on Serial0/1/0, 00:00:01 ago
  Routing Descriptor Blocks:
  * 155.1.45.4, from 155.1.45.4, 00:00:01 ago, via Serial0/1/0
      Route metric is 1, traffic share count is 1
    155.1.0.4, from 155.1.0.4, 00:00:07 ago, via Serial0/0/0.1
      Route metric is 1, traffic share count is 1

Rack1R2#ping 150.1.4.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/116/124
ms
```

Copyright © 2010 Internetwork Expert

Even though administrative distance is locally significant to the router, the RIP process, like the EIGRP process, cannot advertise a route that is not actually installed in the routing table. By setting the distance of the route 150.1.4.0 to 255, it is invalidated from being installed in the routing table, and hence invalidated from being advertised to any neighbors. Once complete R2 not only does not have reachability to the destination, but it does not have the route in its routing table or RIP database.

```
Rack1R5#show ip route 150.1.4.0
% Subnet not in table

Rack1R2#ping 150.1.4.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)

Rack1R2#show ip route 150.1.4.0
% Subnet not in table

Rack1R2#show ip rip database 150.1.4.0 255.255.255.0
%Route not in database
```

Copyright © 2010 Internetwork Expert

4.15 RIPv2 Filtering with Per Neighbor AD

- Configure administrative distance filtering on SW1 so that traffic destined for R3's Loopback 0 network is sent towards R6.
- This configuration should not affect any other networks in the topology.

Configuration

```
SW1:
router rip
 distance 255 155.1.37.3 0.0.0.0 2
!
access-list 2 permit 150.1.3.0
```

Verification

> **Note**
>
> Before distance filtering.

```
Rack1SW1#show ip route 150.1.3.0
Routing entry for 150.1.3.0/24
  Known via "rip", distance 120, metric 1
  Redistributing via rip
  Last update from 155.1.37.3 on FastEthernet0/3, 00:00:01 ago
  Routing Descriptor Blocks:
  * 155.1.37.3, from 155.1.37.3, 00:00:01 ago, via FastEthernet0/3
      Route metric is 1, traffic share count is 1
```

The distance command can be used to change the administrative distance globally for the routing process, globally for the routing process per route type (i.e. external vs. internal), on a per-prefix basis, or on a per-neighbor per-prefix basis. The two fields after the distance value are the source of the route and a wildcard mask to match the source. The value needed for the source is seen in the above output as the **`from 155.1.37.3`** field.

```
Rack1SW1#show ip route 150.1.3.0
Routing entry for 150.1.3.0/24
  Known via "rip", distance 120, metric 3
  Redistributing via rip
  Last update from 155.1.67.6 on Vlan67, 00:00:05 ago
  Routing Descriptor Blocks:
  * 155.1.67.6, from 155.1.67.6, 00:00:05 ago, via Vlan67
      Route metric is 3, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

4.16 RIPv2 Default Routing

- Configure R6 to advertise a default route only to R1 via RIP.
- R6 should not send this default route directly to SW1.
- Do not use any access-list or prefix-lists on R6 to accomplish this.

Configuration

```
R6:
router rip
 default-information originate route-map DEFAULT_TO_R1
!
route-map DEFAULT_TO_R1 permit 10
 set interface FastEthernet0/0.146

SW1:
router rip
 distribute-list prefix NO_DEFAULT out Vlan67
!
ip prefix-list NO_DEFAULT seq 5 deny 0.0.0.0/0
ip prefix-list NO_DEFAULT seq 10 permit 0.0.0.0/0 le 32
```

Verification

```
Rack1R6#show ip rip database 0.0.0.0 0.0.0.0
0.0.0.0/0    redistributed
    [1] via 0.0.0.0,

Rack1R1#show ip route | include ( 0.0.0.0)
Gateway of last resort is 155.1.146.6 to network 0.0.0.0
R*   0.0.0.0/0 [120/6] via 155.1.146.6, 00:00:06, FastEthernet0/0

Rack1R3#show ip route | include ( 0.0.0.0)
Gateway of last resort is 155.1.13.1 to network 0.0.0.0
R*   0.0.0.0/0 [120/7] via 155.1.13.1, 00:00:06, Serial1/2

Rack1R6#show ip route | include ( 0.0.0.0)

Rack1SW1#show ip route | include ( 0.0.0.0)
Gateway of last resort is 155.1.37.3 to network 0.0.0.0
R*   0.0.0.0/0 [120/8] via 155.1.37.3, 00:00:00, FastEthernet0/3
```

Pitfall

Note in the above output that R6 does not have a default route installed in the routing table. Unlike OSPF, RIP does not require that a default route actually be installed in the routing table before originating one. Similarly to the previous problem in this topology with the route feedback of R4's summary route, route feedback of R6's default origination will occur in this topology.

Copyright © 2010 Internetwork Expert

Due to the route-map attached to the default origination of R6, the default advertisement is only sent out VLAN 146 towards R1. When R1 receives this it is installed in the routing table and advertised on to R3. Note that the metric of the default route is 6 when it reaches R1 due to the previous offset-list with a metric of 5 configured outbound on R6's link to VLAN 146.

```
Rack1R1#debug ip rip
RIP protocol debugging is on
RIP: received packet with text authentication CCIE
RIP: received v2 update from 155.1.146.6 on FastEthernet0/0
     0.0.0.0/0 via 0.0.0.0 in 6 hops
<output omitted>
RIP: sending v2 update to 224.0.0.9 via Serial0/1 (155.1.13.1)
RIP: build update entries
        0.0.0.0/0 via 0.0.0.0, metric 7, tag 0
<output omitted>
```

R3 now receives the update in from R1 with a metric of 7, and forwards the announcement to SW1 with a metric of 8.

```
Rack1R3#debug ip rip
RIP protocol debugging is on
RIP: received v2 update from 155.1.13.1 on Serial1/2
     0.0.0.0/0 via 0.0.0.0 in 7 hops
<output omitted>
RIP: sending v2 update to 224.0.0.9 via FastEthernet0/0 (155.1.37.3)
RIP: build update entries
        0.0.0.0/0 via 0.0.0.0, metric 8, tag 0
<output omitted>
```

SW1 receives the default route from R3 with a metric of 8 and installs it. Normally SW1 would continue to advertise this route on to R6, but in the above example the default route is filtered out from this advertisement with a prefix-list applied as a distribute-list.

```
Rack1SW1#debug ip rip
RIP protocol debugging is on
RIP: received v2 update from 155.1.37.3 on FastEthernet0/3
     0.0.0.0/0 via 0.0.0.0 in 8 hops
<output omitted>
RIP: sending v2 update to 224.0.0.9 via Vlan67 (155.1.67.7)
RIP: build update entries
        30.0.0.0/14 via 0.0.0.0, metric 5, tag 0
        31.0.0.0/14 via 0.0.0.0, metric 5, tag 0
        150.1.2.0/24 via 0.0.0.0, metric 4, tag 0
        150.1.5.0/24 via 0.0.0.0, metric 3, tag 0
        150.1.7.0/24 via 0.0.0.0, metric 1, tag 0
        150.1.8.0/24 via 0.0.0.0, metric 4, tag 0
        150.1.9.0/24 via 0.0.0.0, metric 2, tag 0
        155.1.0.0/24 via 0.0.0.0, metric 2, tag 0
```

Copyright © 2010 Internetwork Expert

```
155.1.5.0/24 via 0.0.0.0, metric 3, tag 0
155.1.7.0/24 via 0.0.0.0, metric 1, tag 0
155.1.8.0/24 via 0.0.0.0, metric 4, tag 0
155.1.9.0/24 via 0.0.0.0, metric 2, tag 0
155.1.10.0/24 via 0.0.0.0, metric 5, tag 0
155.1.13.0/24 via 0.0.0.0, metric 2, tag 0
155.1.37.0/24 via 0.0.0.0, metric 1, tag 0
155.1.45.0/24 via 0.0.0.0, metric 3, tag 0
155.1.58.0/24 via 0.0.0.0, metric 3, tag 0
155.1.79.0/24 via 0.0.0.0, metric 1, tag 0
155.1.108.0/24 via 0.0.0.0, metric 4, tag 0
192.10.1.0/24 via 0.0.0.0, metric 4, tag 0
204.12.1.0/24 via 0.0.0.0, metric 4, tag 0
205.90.31.0/24 via 0.0.0.0, metric 11, tag 0
220.20.3.0/24 via 0.0.0.0, metric 11, tag 0
222.22.2.0/24 via 0.0.0.0, metric 11, tag 0
```

Since R6 does not receive the default route from SW1 there is no feedback. Now let's look at what happens in the topology when there is no filter configured on SW1.

```
Rack1SW1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#router rip
Rack1SW1(config-router)#no distribute-list prefix NO_DEFAULT out Vlan67
Rack1SW1(config-router)#end
Rack1SW1#
```

R6 originates the default route to R1 with a metric of 6.

```
Rack1R6#debug ip rip
RIP protocol debugging is on
Rack1R6#debug ip routing
IP routing debugging is on

Jun 27 05:19:03.009: RIP: sending v2 update to 224.0.0.9 via
FastEthernet0/0.146 (155.1.146.6)
Jun 27 05:19:03.009: RIP: build update entries
Jun 27 05:19:03.009:    0.0.0.0/0 via 0.0.0.0, metric 6, tag 0
```

Copyright © 2010 Internetwork Expert

The route is sent from R1, to R3, to SW1, and then fed back to R6. Since R6 does not actually have the default installed in the routing table it thinks that SW1's advertisement is valid.

```
Jun 27 05:19:09.059: RIP: received v2 update from 155.1.67.7 on
FastEthernet0/0.67
Jun 27 05:19:09.059:      0.0.0.0/0 via 0.0.0.0 in 9 hops
Jun 27 05:19:09.059: RT: SET_LAST_RDB for 0.0.0.0/0
  NEW rdb: via 155.1.67.7

Jun 27 05:19:09.059: RT: add 0.0.0.0/0 via 155.1.67.7, rip metric
[120/9]
Jun 27 05:19:09.063: RT: NET-RED 0.0.0.0/0
Jun 27 05:19:09.067: RT: default path is now 0.0.0.0 via 155.1.67.7
Jun 27 05:19:09.067: RT: new default network 0.0.0.0
Jun 27 05:19:09.067: RT: NET-RED 0.0.0.0/0
Jun 27 05:19:09.203: RIP: received v2 update from 155.1.67.7 on
FastEthernet0/0.67
Jun 27 05:19:09.203:      0.0.0.0/0 via 0.0.0.0 in 9 hops
```

R6 now sends a triggered update to R1 with the new default route. The metric is incremented to 10, and then offset 5 for a total of 15 as the route is sent to R1. Even without the offset list on R6 the problem will still occur, however the offset list makes the loop occur more quickly as R6 is feeding the route back with metric increments of 5 instead of 1.

```
Jun 27 05:19:11.070: RIP: sending v2 flash update to 224.0.0.9 via
FastEthernet0/0.146 (155.1.146.6)
Jun 27 05:19:11.074: RIP: build flash update entries
Jun 27 05:19:11.074:    0.0.0.0/0 via 0.0.0.0, metric 15, tag 0
```

The result of this problem can also be viewed in the rest of the topology through the **`debug ip routing`** output.

```
Rack1R1#debug ip routing
IP routing debugging is on
RT: SET_LAST_RDB for 0.0.0.0/0
  NEW rdb: via 155.1.146.6

RT: add 0.0.0.0/0 via 155.1.146.6, rip metric [120/6]
RT: NET-RED 0.0.0.0/0
RT: default path is now 0.0.0.0 via 155.1.146.6
RT: new default network 0.0.0.0
RT: NET-RED 0.0.0.0/0
RT: rip's 0.0.0.0/0 (via 155.1.146.6) metric changed from
distance/metric [120/6] to [120/15]
RT: NET-RED 0.0.0.0/0
RT: del 0.0.0.0 via 155.1.146.6, rip metric [120/15]
RT: delete network route to 0.0.0.0
RT: NET-RED 0.0.0.0/0
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#debug ip routing
IP routing debugging is on
RT: SET_LAST_RDB for 0.0.0.0/0
  NEW rdb: via 155.1.13.1

RT: add 0.0.0.0/0 via 155.1.13.1, rip metric [120/7]
RT: NET-RED 0.0.0.0/0
RT: default path is now 0.0.0.0 via 155.1.13.1
RT: new default network 0.0.0.0
RT: NET-RED 0.0.0.0/0
RT: NET-RED 0.0.0.0/0
RT: del 0.0.0.0 via 155.1.13.1, rip metric [120/7]
RT: delete network route to 0.0.0.0
RT: NET-RED 0.0.0.0/0
RT: NET-RED 0.0.0.0/0

Rack1SW1#debug ip routing
IP routing debugging is on
RT: SET_LAST_RDB for 0.0.0.0/0
  NEW rdb: via 155.1.37.3

RT: add 0.0.0.0/0 via 155.1.37.3, rip metric [120/8]
RT: default path is now 0.0.0.0 via 155.1.37.3
RT: new default network 0.0.0.0
RT: del 0.0.0.0 via 155.1.37.3, rip metric [120/8]
RT: delete network route to 0.0.0.0
```

To fix this problem we need to ensure that R6 does not install a default route via SW1. One way to fix this, as seen above, is to filter the default route from being sent from SW1 to R6. Another solution would be to filter the default in on R6 from SW1. Yet another solution would be to actually configure a static default route on R6, either to Null0 or to another interface, so the RIP route could not be installed in the routing table from SW1.

The simplest solution, although it does not meet the task requirements, is to have R6 advertise the default route out all interfaces. If R6 sends the default route directly to SW1, SW1 cannot send it back to R6 due to split-horizon, and the loop is avoided.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

4.17 RIPv2 Conditional Default Routing

- Remove the previous default route advertisement on R6.
- Configure R4 to originate a default route into the RIP domain.
- If the link to BB3 goes down R4 should withdraw its default advertisement.

Configuration

```
R4:
router rip
 default-information originate route-map TRACK_LINK_TO_BB3
!
ip prefix-list LINK_TO_BB3 seq 5 permit 204.12.1.0/24
!
route-map TRACK_LINK_TO_BB3 permit 10
 match ip address prefix-list LINK_TO_BB3
```

Verification

> **Note**
>
> As long as R4 has a route to the network 204.12.1.0/24 installed in the routing table it will advertise a default route.

```
Rack1R4#show ip route 204.12.1.0 255.255.255.0
Routing entry for 204.12.1.0/24
  Known via "connected", distance 0, metric 0 (connected, via
interface)
  Redistributing via rip
  Advertised by rip
  Routing Descriptor Blocks:
  * directly connected, via FastEthernet0/0
      Route metric is 0, traffic share count is 1

Rack1R5#show ip route | include ( 0.0.0.0)
Gateway of last resort is 155.1.45.4 to network 0.0.0.0
R*   0.0.0.0/0 [120/1] via 155.1.45.4, 00:00:05, Serial0/1/0
```

Copyright © 2010 Internetwork Expert

If 204.12.1.0/24 leaves the routing table, the default route is withdrawn.

```
Rack1SW2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW2(config)#interface Fa0/4
Rack1SW2(config-if)#shutdown
%LINK-5-CHANGED: Interface FastEthernet0/4, changed state to
administratively down
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/4,
changed state to down

Rack1R4#show ip route 204.12.1.0 255.255.255.0
% Network not in table

Rack1R5#show ip route | include ( 0.0.0.0)
```

Pitfall

Remember that the link status of a connected interface is not always a good indication of end-to-end reachability on the segment. In this design if the trunk links in the network fail, if the link from SW3 to BB3 fails, or if BB3 itself fails, R4 has no way to know this. Since the installation of the 204.12.1.0/24 route on R4 is solely dependent on the link-local status between R4 and SW2, this is not a truly fault tolerant design.

Copyright © 2010 Internetwork Expert

4.18 RIPv2 Reliable Conditional Default Routing

- Configure the IP SLA feature on R4 to track ICMP reachability to BB3.
- Modify the previous default route origination on R4 so that if an ICMP echo-reply is not received from BB3 R4 withdraws its default advertisement.

Configuration

```
R4:
ip sla 1
 icmp-echo 204.12.1.254 source-interface FastEthernet0/0
 timeout 50
 frequency 1
ip sla schedule 1 start-time now
!
track 1 rtr 1
!
router rip
 default-information originate route-map RELIABLY_TRACK_LINK_TO_BB3
!
ip route 169.254.0.1 255.255.255.255 Null0 track 1
!
ip prefix-list DUMMY_ROUTE_TRACKED_VIA_SLA seq 5 permit 169.254.0.1/32
!
route-map RELIABLY_TRACK_LINK_TO_BB3 permit 10
 match ip address prefix-list DUMMY_ROUTE_TRACKED_VIA_SLA
```

Verification

> **Note**
>
> In the previous example R4 could not detect an indirect failure in the transit path to BB3. With this example an IP SLA instance is introduced into conditional default origination. The SLA instance checks reachability to BB3 via ICMP every five seconds, with a timeout of two seconds. The SLA instance is then called from an enhanced object, which is called from a static route. This link local route of 169.254.0.1/32 could be any arbitrary dummy prefix. The dummy prefix is then called from a route-map which is tied to the default route origination. Therefore if R4 loses ICMP reachability to BB3 the default route is withdrawn.
>
> With the network properly functioning R4 advertises its default route to R5.

```
Rack1R5#show ip route | in ( 0.0.0.0)
Gateway of last resort is 155.1.45.4 to network 0.0.0.0
R*   0.0.0.0/0 [120/1] via 155.1.45.4, 00:00:06, Serial0/1/0
```

Copyright © 2010 Internetwork Expert

Next an indirect link failure is simulated by disabling the link between SW3 and BB3.

```
Rack1R4#debug ip sla trace
IP SLA Monitor TRACE debugging for all operation is on

Rack1R4#debug track

Rack1R4#debug ip routing
IP routing debugging is on

Rack1R5#debug ip rip
RIP protocol debugging is on

Rack1R5#debug ip routing
IP routing debugging is on

Rack1SW3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW3(config)#interface Fa0/24
Rack1SW3(config-if)#shutdown
Rack1SW3(config-if)#
Jun 27 06:44:19.657: %LINK-5-CHANGED: Interface FastEthernet0/24,
changed state to administratively down
Jun 27 06:44:20.657: %LINEPROTO-5-UPDOWN: Line protocol on Interface
FastEthernet0/24, changed state to down
```

SW3 declares the link to BB3 down. R4 immediately detects this and declares its object down.

```
Rack1R4#
Jun 27 06:44:21.092: IP SLA Monitor(1) Scheduler: Starting an operation
Jun 27 06:44:21.092: IP SLA Monitor(1) echo operation: Sending an echo
operation
Jun 27 06:44:21.144: IP SLA Monitor(1) echo operation: Timeout
Jun 27 06:44:21.144: IP SLA Monitor(1) Scheduler: Updating result
Jun 27 06:44:22.094: IP SLA Monitor(1) Scheduler: Starting an operation
Jun 27 06:44:22.094: IP SLA Monitor(1) echo operation: Sending an echo
operation
Jun 27 06:44:22.146: IP SLA Monitor(1) echo operation: Timeout
Jun 27 06:44:22.146: IP SLA Monitor(1) Scheduler: Updating result
Jun 27 06:44:22.406: Track: 1 Change #16 rtr 1, state Up->Down
```

This causes R4 to withdraw the dummy route from the routing table.

```
Rack1R4#
Jun 27 06:44:22.406: RT: del 169.254.0.1/32 via 0.0.0.0, static metric [1/0]
Jun 27 06:44:22.406: RT: delete subnet route to 169.254.0.1/32
```

Copyright © 2010 Internetwork Expert

This causes the route-map condition to fail, which in turn causes R4 to poison its default route advertisement.

```
Rack1R5#
Jun 27 06:44:24.422: RIP: received v2 update from 155.1.45.4 on Serial0/1/0
Jun 27 06:44:24.422:      0.0.0.0/0 via 0.0.0.0 in 16 hops
(inaccessible)
Jun 27 06:44:24.426: RT: del 0.0.0.0 via 155.1.45.4, rip metric [120/1]
Jun 27 06:44:24.426: RT: SET_LAST_RDB for 0.0.0.0/0
  OLD rdb: via 11.13.11.13
  NEW rdb: via 155.1.0.4, Serial0/0/0.1

Jun 27 06:44:24.426: RT: NET-RED 0.0.0.0/0
Jun 27 06:44:24.510: RIP: received v2 update from 155.1.0.4 on Serial0/0/0.1
Jun 27 06:44:24.510:      0.0.0.0/0 via 0.0.0.0 in 16 hops  (inaccessible)
Jun 27 06:44:24.510: RT: del 0.0.0.0 via 155.1.0.4, rip metric [120/1]
Jun 27 06:44:24.510: RT: delete network route to 0.0.0.0
Jun 27 06:44:24.510: RT: NET-RED 0.0.0.0/0
Jun 27 06:44:24.510: RT: NET-RED 0.0.0.0/0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

4.19 RIPv2 Unicast Updates

- Configure R5 and SW2 so that RIPv2 updates sent over VLAN 58 use unicasts instead of multicasts.

Configuration

```
R5:
router rip
 passive-interface FastEthernet0/0
 neighbor 155.1.58.8

SW2:
router rip
 passive-interface Vlan58
 neighbor 155.1.58.5
```

Verification

> **Note**
>
> Like EIGRP and OSPF the **neighbor** statement in RIP is used to send updates out an interface as unicast. Unlike EIGRP however the **neighbor** statement does not automatically suppress the sending of the broadcast or multicast update. The additional **passive-interface** command is required to accomplish this.

```
Rack1SW2#debug ip packet detail
IP packet debugging is on (detailed)
IP: s=155.1.58.8 (local), d=155.1.58.5 (Vlan58), len 112, sending
    UDP src=520, dst=520
IP: s=155.1.58.8 (local), d=155.1.58.5 (Vlan58), len 112, sending full
packet
    UDP src=520, dst=520
IP: s=155.1.58.5 (Vlan58), d=155.1.58.8, len 532, rcvd 2
    UDP src=520, dst=520
IP: s=155.1.58.5 (Vlan58), d=155.1.58.8, len 532, stop process pak for
forus packet
    UDP src=520, dst=520
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

4.20 RIPv2 Broadcast Updates

- Configure R1 and R6 so that RIPv2 updates sent over VLAN 146 use broadcasts instead of multicasts.

Configuration

```
R1:
interface FastEthernet0/0
 ip rip v2-broadcast

R6:
interface FastEthernet0/0.146
 ip rip v2-broadcast
```

Verification

> **Note**
>
> Normally RIPv2 updates are sent as multicast. The interface level command ip rip v2-broadcast reverts back to using the all host broadcast address 255.255.255.255 for updates.

```
Rack1R1#debug ip packet detail
IP packet debugging is on (detailed)
IP: s=155.1.146.1 (local), d=255.255.255.255 (FastEthernet0/0), len
532, sending broad/multicast
    UDP src=520, dst=520
IP: s=155.1.146.1 (local), d=255.255.255.255 (FastEthernet0/0), len
212, sending broad/multicast
    UDP src=520, dst=520
IP: s=155.1.13.1 (local), d=224.0.0.9 (Serial0/1), len 432, sending
broad/multicast
    UDP src=520, dst=520
IP: s=155.1.146.6 (FastEthernet0/0), d=255.255.255.255, len 272, rcvd 2
    UDP src=520, dst=520
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

4.21 RIPv2 Triggered Updates

- Configure R4 and R5 so that RIPv2 updates are only exchanged over the low speed point-to-point Serial link between them when there is a change in the RIP topology.

Configuration

```
R4:
interface Serial0/1/0
 ip rip triggered

R5:
interface Serial0/1/0
 ip rip triggered
```

Verification

> **Note**
>
> RFC 2091 - Triggered Extensions to RIP to Support Demand Circuits, defines a mechanism similar to OSPF demand circuit in which periodic updates are suppressed out a link. Updates are only sent out triggered links when there is a change in the RIP database. The below output indicates that R5 is sending updates out all its interfaces except the Serial0/1/0 link to R4.

```
Rack1R5#debug ip rip
RIP protocol debugging is on
RIP: sending v2 update to 224.0.0.9 via Serial0/0/0.1 (155.1.0.5)
RIP: build update entries
        0.0.0.0/0 via 155.1.0.4, metric 2, tag 0
        30.0.0.0/14 via 155.1.0.4, metric 3, tag 0
        31.0.0.0/14 via 155.1.0.4, metric 3, tag 0
<output omitted>
RIP: sending v2 update to 224.0.0.9 via FastEthernet0/1 (155.1.5.5)
RIP: build update entries
        0.0.0.0/0 via 0.0.0.0, metric 2, tag 0
        30.0.0.0/14 via 0.0.0.0, metric 3, tag 0
        31.0.0.0/14 via 0.0.0.0, metric 3, tag 0
<output omitted>
RIP: sending v2 update to 155.1.58.8 via FastEthernet0/0 (155.1.58.5)
RIP: build update entries
        0.0.0.0/0 via 0.0.0.0, metric 2, tag 0
        30.0.0.0/14 via 0.0.0.0, metric 3, tag 0
        31.0.0.0/14 via 0.0.0.0, metric 3, tag 0
<output omitted>
```

Copyright © 2010 Internetwork Expert

Note

Another useful show command is “show ip rip database” please note the word permanent associated with all routes learned from R5 via the serial 0/1/0 link. The word permanent when it is seen in the RIP Database is an indicator that rip triggered has been configured.

```
Rack1R5#sh ip rip database | sec via 155.1.45.4
    [1] via 155.1.45.4, 00:05:29 (permanent), Serial0/1/0
   * Triggered Routes:
     - [1] via 155.1.45.4, Serial0/1/0
    [2] via 155.1.45.4, 00:05:29 (permanent), Serial0/1/0
   * Triggered Routes:
     - [2] via 155.1.45.4, Serial0/1/0
    [2] via 155.1.45.4, 00:05:29 (permanent), Serial0/1/0
   * Triggered Routes:
     - [2] via 155.1.45.4, Serial0/1/0
     - [1] via 155.1.45.4, Serial0/1/0
     - [1] via 155.1.45.4, Serial0/1/0
    [1] via 155.1.45.4, 00:05:29 (permanent), Serial0/1/0
   * Triggered Routes:
     - [1] via 155.1.45.4, Serial0/1/0
```

Copyright © 2010 Internetwork Expert

4.22 RIPv2 Source Validation

- Configure R1 and R3 to use PPP on the Serial link between them.
- Remove R1's IP address on this segment.
- Configure R3 to assign R1 the IP address 155.1.13.1/32 via IPCP.
- Ensure that RIPv2 updates sent across this link can be installed in the routing tables of these two devices.

Configuration

```
R1:
interface Serial0/1
 ip address negotiated
encapsulation ppp
!
router rip
 no validate-update-source

R3:
interface Serial1/2
 encapsulation ppp
 peer default ip address 155.1.13.1
```

Verification

> **Note**
>
> Prior to IP address modification on R1, RIP routes from R3 are installed.

```
Rack1R1#show ip route | include via 155.1.13.3
R       155.1.9.0 [120/3] via 155.1.13.3, 00:00:03, Serial0/1
R       155.1.7.0 [120/2] via 155.1.13.3, 00:00:03, Serial0/1
R       155.1.37.0 [120/1] via 155.1.13.3, 00:00:03, Serial0/1
R       155.1.79.0 [120/2] via 155.1.13.3, 00:00:03, Serial0/1
R       155.1.67.0 [120/2] via 155.1.13.3, 00:00:03, Serial0/1
R       54.1.1.0 [120/3] via 155.1.13.3, 00:00:03, Serial0/1
R    212.18.1.0/24 [120/4] via 155.1.13.3, 00:00:03, Serial0/1
R    212.18.3.0/24 [120/4] via 155.1.13.3, 00:00:03, Serial0/1
R       150.1.7.0 [120/2] via 155.1.13.3, 00:00:03, Serial0/1
R       150.1.6.0 [120/3] via 155.1.13.3, 00:00:03, Serial0/1
R       150.1.3.0 [120/1] via 155.1.13.3, 00:00:03, Serial0/1
R       150.1.9.0 [120/3] via 155.1.13.3, 00:00:03, Serial0/1
```

Copyright © 2010 Internetwork Expert

After the IP address is negotiated on R1, it installs a peer-neighbor host route to R3 as 155.1.13.3/32. Since the local link, 155.1.13.1/32, and R3's link are not on the same subnet, R1 cannot accept RIP updates from R3. However, R3 can receive updates from R1, because R3 still sees its own interface as 155.1.13.0/24, which 155.1.13.1 is a part of.

```
Rack1R1#show ip interface brief
Interface                  IP-Address      OK? Method Status
Protocol
FastEthernet0/0            155.1.146.1     YES manual up                    up
Serial0/0                  unassigned      YES unset  up                    up
Serial0/0.1                155.1.0.1       YES manual up                    up
Serial0/1                  155.1.13.1      YES IPCP   up                    up
Loopback0                  150.1.1.1       YES manual up                    up

Rack1R1#show ip route connected
     155.1.0.0/16 is variably subnetted, 16 subnets, 2 masks
C       155.1.146.0/24 is directly connected, FastEthernet0/0
C       155.1.13.3/32 is directly connected, Serial0/1
C       155.1.13.1/32 is directly connected, Serial0/1
C       155.1.0.0/24 is directly connected, Serial0/0.1
     150.1.0.0/24 is subnetted, 8 subnets
C       150.1.1.0 is directly connected, Loopback0

Rack1R3#show ip route connected
     155.1.0.0/16 is variably subnetted, 15 subnets, 2 masks
C       155.1.13.1/32 is directly connected, Serial1/2
C       155.1.13.0/24 is directly connected, Serial1/2
C       155.1.0.0/24 is directly connected, Serial1/0.1
C       155.1.37.0/24 is directly connected, FastEthernet0/0
     150.1.0.0/24 is subnetted, 8 subnets
C       150.1.3.0 is directly connected, Loopback0

Rack1R1#show ip route | include via 155.1.13.3

Rack1R1#debug ip rip
RIP protocol debugging is on
RIP: ignored v2 update from bad source 155.1.13.3 on Serial0/1

Rack1R3#debug ip rip
RIP protocol debugging is on
RIP: received v2 update from 155.1.13.1 on Serial1/2
     155.1.146.0/24 via 0.0.0.0 in 1 hops
     192.10.1.0/24 via 0.0.0.0 in 3 hops
     204.12.1.0/24 via 0.0.0.0 in 3 hops
     205.90.31.0/24 via 0.0.0.0 in 10 hops
     212.18.1.0/24 via 0.0.0.0 in 6 hops
     212.18.3.0/24 via 0.0.0.0 in 6 hops
     220.20.3.0/24 via 0.0.0.0 in 10 hops
     222.22.2.0/24 via 0.0.0.0 in 10 hops
```

Copyright © 2010 Internetwork Expert

Once R1 is configured to ignore update source validation routes from R3 are installed.

```
Rack1R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#router rip
Rack1R1(config-router)#no validate-update-source
Rack1R1(config-router)#end

Rack1R1#show ip route rip | include via 155.1.13.3
R       155.1.9.0/24 [120/3] via 155.1.13.3, 00:00:03
R       155.1.7.0/24 [120/2] via 155.1.13.3, 00:00:03
R       155.1.37.0/24 [120/1] via 155.1.13.3, 00:00:03
R       155.1.79.0/24 [120/2] via 155.1.13.3, 00:00:03
R       155.1.67.0/24 [120/2] via 155.1.13.3, 00:00:03
R       54.1.1.0 [120/3] via 155.1.13.3, 00:00:03
R    212.18.1.0/24 [120/4] via 155.1.13.3, 00:00:03
R    212.18.3.0/24 [120/4] via 155.1.13.3, 00:00:03
R       150.1.7.0 [120/2] via 155.1.13.3, 00:00:03
R       150.1.6.0 [120/3] via 155.1.13.3, 00:00:03
R       150.1.3.0 [120/1] via 155.1.13.3, 00:00:03
R       150.1.9.0 [120/3] via 155.1.13.3, 00:00:03
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

EIGRP Solutions

Note

Load the *Initial EIGRP* initial configurations prior to starting. Note that R4's link to VLAN 146 and the point-to-point link between R2 and R3 are disabled.

5.1 EIGRP Network Statement

- Configure EIGRP AS 100 on all devices in the internal topology.
- Enable EIGRP on all interfaces in the 150.X.0.0 and 155.X.0.0 networks on all devices.
- Any new interfaces added should not automatically have EIGRP enabled on them regardless of their IP addresses.
- Do not disable auto-summary.
- Note any reachability problems throughout the network.

Configuration

```
R1:
router eigrp 100
 network 150.1.1.1 0.0.0.0
 network 155.1.0.1 0.0.0.0
 network 155.1.146.1 0.0.0.0
 network 155.1.13.1 0.0.0.0

R2:
router eigrp 100
 network 150.1.2.2 0.0.0.0
 network 155.1.0.2 0.0.0.0

R3:
router eigrp 100
 network 150.1.3.3 0.0.0.0
 network 155.1.0.3 0.0.0.0
 network 155.1.13.3 0.0.0.0
 network 155.1.37.3 0.0.0.0

R4:
router eigrp 100
 network 150.1.4.4 0.0.0.0
 network 155.1.0.4 0.0.0.0
 network 155.1.45.4 0.0.0.0

R5:
router eigrp 100
 network 150.1.5.5 0.0.0.0
```

Copyright © 2010 Internetwork Expert

```
 network 155.1.0.5 0.0.0.0
 network 155.1.5.5 0.0.0.0
 network 155.1.45.5 0.0.0.0
 network 155.1.58.5 0.0.0.0

R6:
router eigrp 100
 network 150.1.6.6 0.0.0.0
 network 155.1.67.6 0.0.0.0
 network 155.1.146.6 0.0.0.0

SW1:
ip routing
!
router eigrp 100
 network 150.1.7.7 0.0.0.0
 network 155.1.7.7 0.0.0.0
 network 155.1.37.7 0.0.0.0
 network 155.1.67.7 0.0.0.0
 network 155.1.79.7 0.0.0.0

SW2:
ip routing
!
router eigrp 100
 network 150.1.8.8 0.0.0.0
 network 155.1.8.8 0.0.0.0
 network 155.1.58.8 0.0.0.0
 network 155.1.108.8 0.0.0.0

SW3:
ip routing
!
router eigrp 100
 network 150.1.9.9 0.0.0.0
 network 155.1.9.9 0.0.0.0
 network 155.1.79.9 0.0.0.0

SW4:
ip routing
!
router eigrp 100
 network 150.1.10.10 0.0.0.0
 network 155.1.10.10 0.0.0.0
 network 155.1.108.10 0.0.0.0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> The network statement in EIGRP, like in OSPF, does not control what networks are being advertised, but instead controls what interfaces are running the EIGRP process. By using a wildcard address of 0.0.0.0 in the EIGRP network statement this means that only the interface with that particular IP address will have the EIGRP process enabled. By using all zeros in the wildcard mask there is no question as to which interfaces are running the process, and new interfaces added to the device will not automatically be running the EIGRP process.
>
> Once the network statement is configured the first verification you should always do is to check the neighbor adjacencies with the **`show ip eigrp neighbors`** command. A "Q Cnt" (queue count) of zero means that there are no updates waiting to be sent and the network is converged.

```
Rack1R1#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
2   155.1.146.6             Fa0/0             13 00:05:10    4   200  0  17
1   155.1.0.5               Se0/0.1          177 00:05:19   44   264  0  22
0   155.1.13.3              Se0/1             13 00:05:31   23   200  0  44

Rack1R2#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   155.1.0.5               Se0/0.1          174 00:05:22   44   264  0  22

Rack1R3#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
2   155.1.37.7              Fa0/0             12 00:05:08    3   200  0  17
1   155.1.0.5               Se1/0.1          167 00:05:29   43  1140  0  22
0   155.1.13.1              Se1/2             11 00:05:41   20  1140  0  37

Rack1R4#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
1   155.1.0.5               Se0/0.1          163 00:02:59  830  4980  0  28
0   155.1.45.5              Se0/1             10 00:02:59   54   324  0  27

Rack1R5#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
5   155.1.0.4               Se0/0             12 00:03:02   88   528  0  10
4   155.1.45.4              Se0/1             13 00:03:02   43   258  0  8
3   155.1.58.8              Fa0/0             14 00:05:00    9   200  0  7
2   155.1.0.1               Se0/0             11 00:05:36  240  1440  0  35
1   155.1.0.2               Se0/0             12 00:05:36  242  1452  0  3
0   155.1.0.3               Se0/0             14 00:05:36  207  1242  0  46
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
1   155.1.67.7              Fa0/0.67          14 00:05:20    7   200  0  16
0   155.1.146.1             Fa0/0.146         10 00:05:33    2   200  0  36

Rack1SW1#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
Type
                                            (sec)         (ms)       Cnt Num
2   155.1.79.9              Vl79              13 00:05:01   13   200  0  3
1   155.1.67.6              Vl67              12 00:05:24  521  3126  0  18
0   155.1.37.3              Fa0/3             13 00:05:24  418  2508  0  45

Rack1SW2#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
Type
                                            (sec)         (ms)       Cnt Num
1   155.1.108.10            Po1               14 00:04:48    8   200  0  2
0   155.1.58.5              Vl58              13 00:05:16  814  4884  0  23

Rack1SW3#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
Type
                                            (sec)         (ms)       Cnt Num
0   155.1.79.7              Vl79              11 00:05:11  509  3054  0  18

Rack1SW4#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
Type
                                            (sec)         (ms)       Cnt Num
0   155.1.108.8             Po1               12 00:04:54 1417  5000  0  6
```

Two separate design problems exist in the current network topology as it's configured in this section. The first is discontiguous networks, and the second is split-horizon.

The first problem can be seen from the routing table output of any device in the network. Each device has a Loopback0 interface in the major network 150.1.0.0/16, while the transit network between the devices is 155.1.0.0/16. When auto-summary is on in EIGRP networks are summarized as they pass through the major network boundary. This is unlike RIP, which automatically summarizes networks anytime there is a different major network or different subnet masks.

The result of auto-summary being on and the discontiguous networks is that the subnets of the 150.1.0.0/16 network cannot be advertised. This is due to the fact that all routers generate a local route for 150.1.0.0/16 to Null0, and advertise this into the EIGRP topology. When another device receive the advertisement 150.1.0.0/16, which it already has installed locally, the local route is preferred over the received route. This means that since the subnets of 150.1.0.0/16 are not exchanged, no device will have reachability to the Loopback 0 networks of the other devices in the topology.

```
Rack1SW4#show ip route eigrp
     155.1.0.0/16 is variably subnetted, 15 subnets, 2 masks
D       155.1.146.0/24 [90/2175232] via 155.1.108.8, 00:05:01, Port-channel1
D       155.1.8.0/24 [90/15616] via 155.1.108.8, 00:05:01, Port-channel1
D       155.1.9.0/24 [90/2175744] via 155.1.108.8, 00:05:01, Port-channel1
D       155.1.13.0/24 [90/2684672] via 155.1.108.8, 00:05:01, Port-channel1
D       155.1.0.0/24 [90/2172672] via 155.1.108.8, 00:05:01, Port-channel1
D       155.1.0.0/16 is a summary, 00:05:06, Null0
D       155.1.7.0/24 [90/2175488] via 155.1.108.8, 00:05:01, Port-channel1
D       155.1.5.0/24 [90/30976] via 155.1.108.8, 00:05:01, Port-channel1
D       155.1.58.0/24 [90/15616] via 155.1.108.8, 00:05:01, Port-channel1
D       155.1.45.0/24 [90/2172672] via 155.1.108.8, 00:05:01, Port-channel1
D       155.1.37.0/24 [90/2175232] via 155.1.108.8, 00:05:01, Port-channel1
D       155.1.79.0/24 [90/2175488] via 155.1.108.8, 00:05:01, Port-channel1
D       155.1.67.0/24 [90/2175488] via 155.1.108.8, 00:05:01, Port-channel1
     150.1.0.0/16 is variably subnetted, 2 subnets, 2 masks
D       150.1.0.0/16 is a summary, 00:05:02, Null0
```

Copyright © 2010 Internetwork Expert

The second problem related to auto-summary can be seen on R2. Since R2's only connection to the rest of the EIGRP network is through the Frame Relay network, all advertisements that R5 receives in the Frame Relay interface cannot be sent back out to R2. This is similar to the RIP split-horizon problem previously introduced, however EIGRP split-horizon is enabled on all interfaces, regardless if they are main interfaces or subinterfaces. To resolve this issue R5 needs to disable split-horizon for this EIGRP process by using the command no ip split-horizon eigrp 100 under the Frame Relay interface.

```
Rack1R2#show ip route eigrp
     155.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
D       155.1.10.0/24 [90/2175232] via 155.1.0.5, 00:06:17, Serial0/0.1
D       155.1.8.0/24 [90/2172672] via 155.1.0.5, 00:06:49, Serial0/0.1
D       155.1.0.0/16 is a summary, 00:07:50, Null0
D       155.1.5.0/24 [90/2172416] via 155.1.0.5, 00:07:24, Serial0/0.1
D       155.1.58.0/24 [90/2172416] via 155.1.0.5, 00:07:24, Serial0/0.1
D       155.1.45.0/24 [90/2681856] via 155.1.0.5, 00:07:24, Serial0/0.1
D       155.1.108.0/24 [90/2174976] via 155.1.0.5, 00:06:47, Serial0/0.1
     150.1.0.0/16 is variably subnetted, 2 subnets, 2 masks
D       150.1.0.0/16 is a summary, 00:07:24, Null0
```

Copyright © 2010 Internetwork Expert

5.2 EIGRP Auto-Summary

- Disable auto-summary on all devices running EIGRP.
- Note any changes in reachability throughout the network.

Configuration

```
R1 - R6, SW1 - SW4:
router eigrp 100
 no auto-summary
```

Verification

> **Note**
>
> With EIGRP auto-summary disabled the subnets of the discontiguous network 150.1.0.0/16 can be advertised to all neighbors. The result of this can be seen anywhere in the topology through the **`show ip route eigrp`** output, as now the individual /24 subnets of 150.1.0.0 are installed.
>
> As a general rule auto-summary for EIGRP would always be disabled in a real design in order for the protocol to function in a truly "classless" fashion. Within the scope of the lab exam it would be safe to assume that auto-summary for EIGRP could be disabled unless there is a specific question telling you to leave auto-summary on.

```
Rack1SW4#show ip route eigrp
     155.1.0.0/24 is subnetted, 14 subnets
D       155.1.146.0 [90/2175232] via 155.1.108.8, 00:00:15, Port-channel1
D       155.1.8.0 [90/15616] via 155.1.108.8, 00:00:15, Port-channel1
D       155.1.9.0 [90/2175744] via 155.1.108.8, 00:00:11, Port-channel1
D       155.1.13.0 [90/2684672] via 155.1.108.8, 00:00:15, Port-channel1
D       155.1.0.0 [90/2172672] via 155.1.108.8, 00:00:15, Port-channel1
D       155.1.7.0 [90/2175488] via 155.1.108.8, 00:00:11, Port-channel1
D       155.1.5.0 [90/30976] via 155.1.108.8, 00:00:15, Port-channel1
D       155.1.58.0 [90/15616] via 155.1.108.8, 00:00:15, Port-channel1
D       155.1.45.0 [90/2172672] via 155.1.108.8, 00:00:15, Port-channel1
D       155.1.37.0 [90/2175232] via 155.1.108.8, 00:00:15, Port-channel1
D       155.1.79.0 [90/2175488] via 155.1.108.8, 00:00:12, Port-channel1
D       155.1.67.0 [90/2175488] via 155.1.108.8, 00:00:15, Port-channel1
     150.1.0.0/24 is subnetted, 10 subnets
D       150.1.7.0 [90/2303232] via 155.1.108.8, 00:00:12, Port-channel1
D       150.1.6.0 [90/2303232] via 155.1.108.8, 00:00:16, Port-channel1
D       150.1.5.0 [90/143616] via 155.1.108.8, 00:00:16, Port-channel1
D       150.1.4.0 [90/2300672] via 155.1.108.8, 00:00:16, Port-channel1
D       150.1.3.0 [90/2300672] via 155.1.108.8, 00:00:16, Port-channel1
D       150.1.2.0 [90/2300672] via 155.1.108.8, 00:00:16, Port-channel1
D       150.1.1.0 [90/2300672] via 155.1.108.8, 00:00:16, Port-channel1
D       150.1.9.0 [90/2303488] via 155.1.108.8, 00:00:13, Port-channel1
D       150.1.8.0 [90/143360] via 155.1.108.8, 00:00:16, Port-channel1
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

5.3 EIGRP Split Horizon

- Disable split-horizon for EIGRP on R5's connection to the Frame Relay network.
- Note any changes in reachability throughout the network.

Configuration

```
R5:
interface Serial0/0/0
 no ip split-horizon eigrp 100
```

Verification

> **Note**
>
> Once split-horizon for EIGRP is disabled on R5's link to the Frame Relay network updates can be exchanged to R2 as they come from other neighbors on the Frame Relay segment, such as R1. The result of this can be seen in the routing table of R2.

```
Rack1R2#show ip route eigrp
     155.1.0.0/24 is subnetted, 14 subnets
D       155.1.146.0 [90/2684416] via 155.1.0.5, 00:00:40, Serial0/0.1
D       155.1.10.0 [90/2175232] via 155.1.0.5, 00:04:15, Serial0/0.1
D       155.1.8.0 [90/2172672] via 155.1.0.5, 00:04:19, Serial0/0.1
D       155.1.9.0 [90/2684928] via 155.1.0.5, 00:00:40, Serial0/0.1
D       155.1.13.0 [90/3193856] via 155.1.0.5, 00:00:40, Serial0/0.1
D       155.1.7.0 [90/2684672] via 155.1.0.5, 00:00:40, Serial0/0.1
D       155.1.5.0 [90/2172416] via 155.1.0.5, 00:27:31, Serial0/0.1
D       155.1.58.0 [90/2172416] via 155.1.0.5, 00:27:31, Serial0/0.1
D       155.1.45.0 [90/2681856] via 155.1.0.5, 00:27:31, Serial0/0.1
D       155.1.37.0 [90/2684416] via 155.1.0.5, 00:00:40, Serial0/0.1
D       155.1.79.0 [90/2684672] via 155.1.0.5, 00:00:40, Serial0/0.1
D       155.1.67.0 [90/2684672] via 155.1.0.5, 00:00:40, Serial0/0.1
D       155.1.108.0 [90/2174976] via 155.1.0.5, 00:04:19, Serial0/0.1
     150.1.0.0/24 is subnetted, 10 subnets
D       150.1.7.0 [90/2812416] via 155.1.0.5, 00:00:42, Serial0/0.1
D       150.1.6.0 [90/2812416] via 155.1.0.5, 00:00:42, Serial0/0.1
D       150.1.5.0 [90/2297856] via 155.1.0.5, 00:04:23, Serial0/0.1
D       150.1.4.0 [90/2809856] via 155.1.0.5, 00:00:42, Serial0/0.1
D       150.1.3.0 [90/2809856] via 155.1.0.5, 00:00:42, Serial0/0.1
D       150.1.1.0 [90/2809856] via 155.1.0.5, 00:00:42, Serial0/0.1
D       150.1.10.0 [90/2302976] via 155.1.0.5, 00:04:17, Serial0/0.1
D       150.1.9.0 [90/2812672] via 155.1.0.5, 00:00:42, Serial0/0.1
D       150.1.8.0 [90/2300416] via 155.1.0.5, 00:04:21, Serial0/0.1
```

Copyright © 2010 Internetwork Expert

Although disabling split-horizon on R5 does not cause a routing loop, it does add additional route replication into the topology. For example if we look at the EIGRP topology table on R2 for the prefix 150.1.2.0/24, which is directly connected, we can see that this prefix is originated from the connected route, but is also learned back in from R5.

```
Rack1R2#show ip eigrp topology 150.1.2.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 150.1.2.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 128256
  Routing Descriptor Blocks:
  0.0.0.0 (Loopback0), from Connected, Send flag is 0x0
      Composite metric is (128256/0), Route is Internal
      Vector metric:
        Minimum bandwidth is 10000000 Kbit
        Total delay is 5000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1514
        Hop count is 0
  155.1.0.5 (Serial0/0.1), from 155.1.0.5, Send flag is 0x0
      Composite metric is (2809856/2297856), Route is Internal
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 45000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
```

Even though there are multiple paths to the same destination a loop cannot occur based on the EIGRP feasibility condition. The feasibility condition determines which routes from the EIGRP topology will actually be used for forwarding in the IP routing table.

First off the complete end-to-end composite metric is compared between routes. In this case R2's local route has a metric of 128,256, while R5's route has a metric of 2,809,856. This value is seen as the first value in parenthesis before the slash. The lower of these values, 128,256, is considered the Feasible Distance, and is the end-to-end metric of the active forwarding path. This active forwarding path is called the Successor.

Next, R2 compares the metric that the upstream neighbor is advertising for the destination. In this case R5 is advertising a metric of 2,297,856. This value is seen as the second number inside the parenthesis, the Advertised Distance, is used to find alternate loop-free forwarding paths. If the Advertised Distance of a neighbor's route is lower than the current Feasible Distance of the Successor, the route is considered an alternate path, or a Feasible Successor.

In this case R5's Advertised Distance is higher than R2's Feasible Distance, so the route is discarded and not considered a valid path.

Copyright © 2010 Internetwork Expert

5.4 EIGRP MD5 Authentication

- Configure EIGRP 10 on the link between R6 and BB1.
- Authenticate this adjacency with the MD5 key 1 using the password CISCO.
- Use a key-chain named MD5_KEYS.

Configuration

```
R6:
key chain MD5_KEYS
 key 1
  key-string CISCO
!
interface Serial0/0/0
 ip authentication mode eigrp 10 md5
 ip authentication key-chain eigrp 10 MD5_KEYS
!
router eigrp 10
 network 54.1.1.6 0.0.0.0
 no auto-summary
```

Verification

> **Note**
>
> Without authentication configured hello packets from BB1 are ignored.

```
Rack1R6#debug eigrp packet
EIGRP Packets debugging is on
    (UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK, STUB,
SIAQUERY, SIAREPLY)

EIGRP: Serial0/0/0: ignored packet from 54.1.1.254, opcode = 5
(authentication off)
```

Copyright © 2010 Internetwork Expert

If authentication is successful the neighbor relationship should appear immediately.

```
Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#
Rack1R6(config)#interface Serial0/0/0
Rack1R6(config-if)#ip authentication mode eigrp 10 md5
Rack1R6(config-if)#end
Rack1R6#
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 10: Neighbor 54.1.1.254 (Serial0/0/0) is
up: new adjacency

Rack1R6#show ip route eigrp 10
D    200.0.0.0/24 [90/2297856] via 54.1.1.254, 00:00:10, Serial0/0/0
D    200.0.1.0/24 [90/2297856] via 54.1.1.254, 00:00:10, Serial0/0/0
D    200.0.2.0/24 [90/2297856] via 54.1.1.254, 00:00:10, Serial0/0/0
D    200.0.3.0/24 [90/2297856] via 54.1.1.254, 00:00:10, Serial0/0/0
```

Pitfall

Like EIGRP a white space in the key-string can cause authentication failure.

```
Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#key chain MD5_KEYS
Rack1R6(config-keychain)#key 1
Rack1R6(config-keychain-key)#key-string CISCO ?
LINE     <cr>

Rack1R6(config-keychain-key)#key-string CISCO
Rack1R6(config-keychain-key)#interface Serial0/0/0
Rack1R6(config-if)#ip authentication mode eigrp 10 md5
Rack1R6(config-if)#ip authentication key-chain eigrp 10 MD5_KEYS
Rack1R6(config-if)#end
Rack1R6#
%SYS-5-CONFIG_I: Configured from console by console
Rack1R6#show key chain
Key-chain MD5_KEYS:
    key 1 -- text "CISCO "
        accept lifetime (always valid) - (always valid) [valid now]
        send lifetime (always valid) - (always valid) [valid now]

Rack1R6#debug eigrp packets
EIGRP Packets debugging is on
    (UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK, STUB,
SIAQUERY, SIAREPLY)
Rack1R6#
EIGRP: pkt key id = 1, authentication mismatch
EIGRP: Serial0/0/0: ignored packet from 54.1.1.254, opcode = 5 (invalid
authentication)
```

Copyright © 2010 Internetwork Expert

5.5 EIGRP Key Chain Rotation

- Authenticate the EIGRP adjacencies on the Frame Relay network between R1, R2, R3, R4, and R5 using key 10 and the password CISCO10.
- Key 10 should be sent until 12:05 AM on Jan 1st 2030, and should be accepted for 10 minutes past this time.
- Configure a new key 20 with the password CISCO20 that is sent starting 12:00 AM on Jan 1st 2030, and is accepted any time after this time.
- Use a key-chain named KEY_ROTATION.

Configuration

```
R1, R2:
key chain KEY_ROTATION
 key 10
  key-string CISCO10
  accept-lifetime 00:00:00 Jan 1 1993 00:15:00 Jan 1 2030
  send-lifetime 00:00:00 Jan 1 1993 00:05:00 Jan 1 2030
 key 20
  key-string CISCO20
  accept-lifetime 00:00:00 Jan 1 2030 infinite
  send-lifetime 00:00:00 Jan 1 2030 infinite
!
interface Serial0/0.1 point-to-point
 ip authentication mode eigrp 100 md5
 ip authentication key-chain eigrp 100 KEY_ROTATION

R4:
key chain KEY_ROTATION
 key 10
  key-string CISCO10
  accept-lifetime 00:00:00 Jan 1 1993 00:15:00 Jan 1 2030
  send-lifetime 00:00:00 Jan 1 1993 00:05:00 Jan 1 2030
 key 20
  key-string CISCO20
  accept-lifetime 00:00:00 Jan 1 2030 infinite
  send-lifetime 00:00:00 Jan 1 2030 infinite
!
interface Serial0/0/0.1 point-to-point
 ip authentication mode eigrp 100 md5
 ip authentication key-chain eigrp 100 KEY_ROTATION

R3:
key chain KEY_ROTATION
 key 10
  key-string CISCO10
  accept-lifetime 00:00:00 Jan 1 1993 00:15:00 Jan 1 2030
  send-lifetime 00:00:00 Jan 1 1993 00:05:00 Jan 1 2030
 key 20
  key-string CISCO20
  accept-lifetime 00:00:00 Jan 1 2030 infinite
  send-lifetime 00:00:00 Jan 1 2030 infinite
```

Copyright © 2010 Internetwork Expert

```
interface Serial1/0.1 point-to-point
 ip authentication mode eigrp 100 md5
 ip authentication key-chain eigrp 100 KEY_ROTATION

R5:
key chain KEY_ROTATION
 key 10
  key-string CISCO10
  accept-lifetime 00:00:00 Jan 1 1993 00:15:00 Jan 1 2030
  send-lifetime 00:00:00 Jan 1 1993 00:05:00 Jan 1 2030
 key 20
  key-string CISCO20
  accept-lifetime 00:00:00 Jan 1 2030 infinite
  send-lifetime 00:00:00 Jan 1 2030 infinite
!
interface Serial0/0/0
 ip authentication mode eigrp 100 md5
 ip authentication key-chain eigrp 100 KEY_ROTATION
```

Verification

Pitfall

Anytime time based authentication is configured ensure that all devices agree on the same time. This can be manually configured with the **`clock set`** command or through NTP. Also the additional overlap of sending/receiving keys ensures that a drift away from the accurate time will not cause routing adjacencies to be lost.

```
Rack1R2#show clock
00:04:55.223 UTC Tue Jan 1 2030

Rack1R2#show key chain KEY_ROTATION
Key-chain KEY_ROTATION:
    key 10 -- text "CISCO10"
        accept lifetime (00:00:00 UTC Jan 1 1993) - (00:15:00 UTC Jan 1 2030)
[valid now]
        send lifetime (00:00:00 UTC Jan 1 1993) - (00:05:00 UTC Jan 1 2030)
[valid now]
    key 20 -- text "CISCO20"
        accept lifetime (00:00:00 UTC Jan 1 2030) - (infinite) [valid now]
        send lifetime (00:00:00 UTC Jan 1 2030) - (infinite) [valid now]

Rack1R2#show clock
00:05:30.954 UTC Tue Jan 1 2030

Rack1R2#show key chain KEY_ROTATION
Key-chain KEY_ROTATION:
    key 10 -- text "CISCO10"
        accept lifetime (00:00:00 UTC Jan 1 1993) - (00:15:00 UTC Jan 1 2030)
[valid now]
        send lifetime (00:00:00 UTC Jan 1 1993) - (00:05:00 UTC Jan 1 2030)
    key 20 -- text "CISCO20"
        accept lifetime (00:00:00 UTC Jan 1 2030) - (infinite) [valid now]
        send lifetime (00:00:00 UTC Jan 1 2030) - (infinite) [valid now]
```

Copyright © 2010 Internetwork Expert

5.6 EIGRP Unicast Updates

- Configure R5 and SW2 so that they exchange EIGRP packets only as unicasts on their connection to VLAN 58.

Configuration

```
R5:
router eigrp 100
 neighbor 155.1.58.8 FastEthernet0/0

SW2:
router eigrp 100
 neighbor 155.1.58.5 Vlan58
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> By default EIGRP hello packets are sent to the multicast address 224.0.0.10, while topology synchronization between two neighbors is unicast. Like RIP the **neighbor** statement under the EIGRP process is used to send hello packets as unicasts. However unlike RIP the **passive-interface** command is not needed to suppress the sending of the multicast hellos. This means that if the **neighbor** statement is configured on one end of the adjacency it is required that the neighbor statement be configured on the other end.

```
Rack1R5#debug ip packet detail
IP packet debugging is on (detailed)
IP: s=155.1.58.5 (local), d=224.0.0.10 (FastEthernet0/0), len 60,
sending broad/multicast, proto=88
IP: s=155.1.58.8 (FastEthernet0/0), d=224.0.0.10, len 60, rcvd 2,
proto=88

Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#router eigrp 100
Rack1R5(config-router)#neighbor 155.1.58.8 FastEthernet0/0
Rack1R5(config-router)#end
Rack1R5#

Rack1SW2#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW2(config)#router eigrp 100
Rack1SW2(config-router)#neighbor 155.1.58.5 Vlan58
Rack1SW2(config-router)#end
Rack1SW2#

Rack1R5#debug ip packet detail
IP packet debugging is on (detailed)
IP: s=155.1.58.5 (local), d=155.1.58.8 (FastEthernet0/0), len 60,
sending, proto=88
IP: tableid=0, s=155.1.58.8 (FastEthernet0/0), d=155.1.58.5
(FastEthernet0/0), routed via RIB
IP: s=155.1.58.8 (FastEthernet0/0), d=155.1.58.5 (FastEthernet0/0), len
60, rcvd 3, proto=88
```

Copyright © 2010 Internetwork Expert

5.7 EIGRP Default Network

- Redistribute between EIGRP AS 10 and EIGRP AS 100 on R6.
- Configure R6 to advertise the network 200.0.0.0/24 as the default network to all devices in EIGRP AS 100.

Configuration

```
R6:
router eigrp 100
 redistribute eigrp 10
!
router eigrp 10
 redistribute eigrp 100
!
ip default-network 200.0.0.0
```

Verification

> **Note**
>
> The original implementation of IGRP did not support the advertisement of the network 0.0.0.0/0, so the **`ip default-network`** command was used as a workaround. Although EIGRP does support the direct advertisement of 0.0.0.0/0, it also inherits the default network behavior from IGRP. A default network is a classful major network that is advertised as the candidate destination for unknown traffic to be forwarded towards. This network is denoted with an * in the routing table, as seen from the below output on SW4.

```
Rack1SW4#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 155.1.108.8 to network 200.0.0.0

     155.1.0.0/24 is subnetted, 14 subnets
D       155.1.146.0 [90/2175232] via 155.1.108.8, 00:16:23, Port-channel1
C       155.1.10.0 is directly connected, Vlan10
D       155.1.7.0 [90/2175488] via 155.1.108.8, 00:16:24, Port-channel1
D       155.1.5.0 [90/30976] via 155.1.108.8, 00:16:24, Port-channel1
D       155.1.58.0 [90/15616] via 155.1.108.8, 01:34:41, Port-channel1
D       155.1.45.0 [90/2172672] via 155.1.108.8, 00:16:24, Port-channel1
D       155.1.37.0 [90/2175232] via 155.1.108.8, 00:16:24, Port-channel1
D       155.1.79.0 [90/2175488] via 155.1.108.8, 00:16:24, Port-channel1
D       155.1.67.0 [90/2175488] via 155.1.108.8, 00:16:24, Port-channel1
C       155.1.108.0 is directly connected, Port-channel1
D*EX 200.0.0.0/24 [170/2815232] via 155.1.108.8, 00:08:51, Port-channel1
<output omitted>
```

Copyright © 2010 Internetwork Expert

If we trace the path of the default network back to the source we can see that the gateway of last resort (the default next-hop) changes on a per router basis.

```
Rack1SW4#show ip route | include last resort|D\*
Gateway of last resort is 155.1.108.8 to network 200.0.0.0
D*EX 200.0.0.0/24 [170/2815232] via 155.1.108.8, 00:16:04, Port-channel1

Rack1SW2#show ip route | include last resort|D\*
Gateway of last resort is 155.1.58.5 to network 200.0.0.0
D*EX 200.0.0.0/24 [170/2812672] via 155.1.58.5, 00:16:15, Vlan58

Rack1R5#show ip route | include last resort|D\*
Gateway of last resort is 155.1.0.1 to network 200.0.0.0
D*EX 200.0.0.0/24 [170/2812416] via 155.1.0.1, 00:16:18, Serial0/0/0

Rack1R1#show ip route | include last resort|D\*
Gateway of last resort is 155.1.146.6 to network 200.0.0.0
D*EX 200.0.0.0/24 [170/2300416] via 155.1.146.6, 00:16:21, FastEthernet0/0

Rack1R6#show ip route | include last resort|D\*
Gateway of last resort is 54.1.1.254 to network 200.0.0.0
D*   200.0.0.0/24 [90/2297856] via 54.1.1.254, 00:53:29, Serial0/0/0
```

While technically not a “default route”, the result of the default network is the same. Traffic for unknown destinations is forwarded towards the device that originates the default network.

```
Rack1SW4#show ip route 1.2.3.4
% Network not in table

Rack1SW4#traceroute 1.2.3.4

Type escape sequence to abort.
Tracing the route to 1.2.3.4

  1 155.1.108.8 0 msec 0 msec 0 msec
  2 155.1.58.5 0 msec 4 msec 0 msec
  3 155.1.0.1 28 msec 28 msec 28 msec
  4 155.1.146.6 28 msec 28 msec 28 msec
  5 54.1.1.254 44 msec 48 msec 44 msec
  6 54.1.1.254 !H  *  !H
```

Copyright © 2010 Internetwork Expert

5.8 EIGRP Summarization

- Redistribute between RIP and EIGRP AS 100 on R4.
- Use the metric of R4's connection to VLAN 43 to translate RIP metrics into EIGRP metrics.
- Configure R4 to summarize the 30.0.0.0 subnets to R5 out the Frame Relay link, and the 31.0.0.0 subnets out the point-to-point link.
- Do not overlap any address space that R4 does not have a longer match to.
- If R4's point-to-point link is down traffic for the 30.0.0.0 subnets should be rerouted out the Frame Relay link.
- If R4's Frame Relay link is down traffic for the 31.0.0.0 subnets should be rerouted out the point-to-point link.

Configuration

```
R4:
interface Serial0/0/0.1 point-to-point
 ip summary-address eigrp 100 30.0.0.0 255.252.0.0 5
!
interface Serial0/1/0
 ip summary-address eigrp 100 31.0.0.0 255.252.0.0 5
!
router eigrp 100
 redistribute rip metric 100000 10 255 1 1500
!
router rip
 redistribute eigrp 100 metric 1
```

Verification

Note

Like RIP, EIGRP supports summarization at the interface level anywhere throughout the topology, but does not have the limitation of not being able to summarize beyond the classful boundary. When a summary is configured in EIGRP all subnets that make up the summary are suppressed from being advertised out the link. Design-wise this feature can be used to both reduce the size of the routing table and to limit the scope of EIGRP query messages.

Copyright © 2010 Internetwork Expert

In the below output we can see that R5 learns the summary 30.0.0.0/14 in the Frame Relay network, and the /16 subnets in the point-to-point link. Based on longest match routing we can infer that R5 will send traffic for any subnet of the aggregate out the point-to-point link.

```
Rack1R5#show ip route | include 30\.|31\.
     31.0.0.0/8 is variably subnetted, 5 subnets, 2 masks
D EX    31.3.0.0/16 [170/2172416] via 155.1.0.4, 00:01:22, Serial0/0/0
D EX    31.2.0.0/16 [170/2172416] via 155.1.0.4, 00:01:22, Serial0/0/0
D EX    31.1.0.0/16 [170/2172416] via 155.1.0.4, 00:01:22, Serial0/0/0
D EX    31.0.0.0/16 [170/2172416] via 155.1.0.4, 00:01:22, Serial0/0/0
D       31.0.0.0/14 [90/2172416] via 155.1.45.4, 00:01:21, Serial0/1/0
     30.0.0.0/8 is variably subnetted, 5 subnets, 2 masks
D EX    30.2.0.0/16 [170/2172416] via 155.1.45.4, 00:01:20, Serial0/1/0
D EX    30.3.0.0/16 [170/2172416] via 155.1.45.4, 00:01:20, Serial0/1/0
D EX    30.0.0.0/16 [170/2172416] via 155.1.45.4, 00:01:20, Serial0/1/0
D       30.0.0.0/14 [90/2172416] via 155.1.0.4, 00:01:27, Serial0/0/0
D EX    30.1.0.0/16 [170/2172416] via 155.1.45.4, 00:01:20, Serial0/1/0

Rack1R5#traceroute 30.0.0.1

Type escape sequence to abort.
Tracing the route to 30.0.0.1

  1 155.1.45.4 16 msec 12 msec 16 msec
  2 204.12.1.254 16 msec *  16 msec
```

If the point-to-point link fails the longest match for these destinations becomes the /14 summary, and traffic is routed out the Frame Relay network.

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface Serial0/1/0
Rack1R4(config-if)#shutdown
Rack1R4(config-if)#

Rack1R5#traceroute 30.0.0.1

Type escape sequence to abort.
Tracing the route to 30.0.0.1

  1 155.1.0.4 32 msec 28 msec 32 msec
  2 204.12.1.254 32 msec *  32 msec

Rack1R5#show ip route | include 30\.|31\.
     31.0.0.0/16 is subnetted, 4 subnets
D EX    31.3.0.0 [170/2172416] via 155.1.0.4, 00:01:55, Serial0/0/0
D EX    31.2.0.0 [170/2172416] via 155.1.0.4, 00:01:55, Serial0/0/0
D EX    31.1.0.0 [170/2172416] via 155.1.0.4, 00:01:55, Serial0/0/0
D EX    31.0.0.0 [170/2172416] via 155.1.0.4, 00:01:55, Serial0/0/0
     30.0.0.0/14 is subnetted, 1 subnets
D       30.0.0.0 [90/2172416] via 155.1.0.4, 00:02:00, Serial0/0/0
```

Copyright © 2010 Internetwork Expert

5.9 EIGRP Summarization with Default Routing

- Remove R6's default network advertisement.
- Remove R4's previous summarization.
- Configure summarization on R4's connections to R5 so that it only advertises a default route out to R5 via EIGRP.

Configuration

```
R4:
interface Serial0/0/0.1 point-to-point
 ip summary-address eigrp 100 0.0.0.0 0.0.0.0 5
!
interface Serial0/1/0
 ip summary-address eigrp 100 0.0.0.0 0.0.0.0 5
```

Verification

> **Note**
>
> Summarization can also be used to originate a default route in EIGRP. The disadvantage of this configuration however is that all subnets previously advertised out an interface will be suppressed, since all IPv4 networks are a subnet of the aggregate 0.0.0.0/0.

```
Rack1R5#show ip route | include via 155.1.(0|45).4
D*    0.0.0.0/0 [90/2172416] via 155.1.45.4, 00:00:36, Serial0/1/0
                [90/2172416] via 155.1.0.4, 00:00:36, Serial0/0/0

Rack1R5#show ip route 30.0.0.1
% Network not in table

Rack1R5#traceroute 30.0.0.1

Type escape sequence to abort.
Tracing the route to 30.0.0.1

  1 155.1.45.4 24 msec
    155.1.0.4 28 msec
    155.1.45.4 20 msec
  2 204.12.1.254 32 msec *  32 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

5.10 EIGRP Summarization with Leak Map

- Configure a leak-map on R4 so that traffic going to R4's Loopback 0 network is routed out the point-to-point link between R4 and R5.
- If this link is down traffic should still be rerouted out the Frame Relay connection between these devices.

Configuration

```
R4:
interface Serial0/1/0
 ip summary-address eigrp 100 0.0.0.0 0.0.0.0 5 leak-map LEAK_LOOPBACK0
!
ip prefix-list LOOPBACK0 seq 5 permit 150.1.4.0/24
!
route-map LEAK_LOOPBACK0 permit 10
 match ip address prefix-list LOOPBACK0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> The EIGRP **leak-map** feature of the **summary-address** allows the advertisement of specific subnets encompassed by the interface level summary, similar to the **unsuppress-map** feature of BGP aggregation. Routes matched in the leak-map route-map will be advertised in addition to the summary. If the route-map matches all routes, all subnets of the aggregate will be advertised in addition to the aggregate. This is useful in cases where you want to originate a default route with the interface summary-address, but don't want to stop the advertisement of any subnets.
>
> In this particular design the leak-map is used to enforce longest match routing traffic engineering. Since R5 has a longer match for the prefix 150.1.4.0/24 via the Serial0/1/0 interface, traffic for this prefix will never get routed over the Frame Relay network unless the point-to-point link is down.

```
Rack1R5#show ip route | include via 155.1.(0|45).4
D       150.1.4.0 [90/2297856] via 155.1.45.4, 00:00:04, Serial0/1/0
D*   0.0.0.0/0 [90/2172416] via 155.1.45.4, 00:00:04, Serial0/1/0
                [90/2172416] via 155.1.0.4, 00:00:04, Serial0/0/0

Rack1R5#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.45.4 16 msec *  12 msec

Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface Serial0/1/0
Rack1R4(config-if)#shutdown
Rack1R4(config-if)#

Rack1R5#show ip route | include via 155.1.(0|45).4
D*   0.0.0.0/0 [90/2172416] via 155.1.0.4, 00:00:03, Serial0/0/0

Rack1R5#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.0.4 32 msec *  28 msec
```

Copyright © 2010 Internetwork Expert

5.11 EIGRP Floating Summarization

- Shutdown the point-to-point link between R4 and R5.
- Configure R5 to summarize the Loopback 0 networks of R4 and R5 out to SW2; this route should not overlap any additional networks.
- Configure an equal longest match static route on R5 so that SW2 has reachability to both Loopback 0 networks of R4 and R5.

Configuration

```
R5:
interface FastEthernet0/0
 ip summary-address eigrp 100 150.1.4.0 255.255.254.0 5
!
ip route 150.1.4.0 255.255.254.0 155.1.0.4
```

Verification

> **Note**
>
> When summaries are created in EIGRP, OSPF, and BGP the router automatically installs a route to Null0 to match the summary. This is used to prevent the router from forwarding traffic for destinations inside the summary that it does not have a longer match for. However in certain designs this can be an undesirable behavior. To resolve this EIGRP sets its interface level summaries to have an administrative distance of 5 by default. This means that any other route with a distance of 1 – 4 will take precedence over the summary.
>
> In this particular case before summarization is configured on R5, SW2 has the subnet route 150.1.5.0/24, and a default route to reach 150.1.4.4. This is because R4 is generating a default route and suppressing its subnet advertisements.

```
Rack1SW2#show ip route 150.1.4.4
% Subnet not in table

Rack1SW2#show ip route 150.1.5.5
Routing entry for 150.1.5.0/24
  Known via "eigrp 100", distance 90, metric 130816, type internal
  Redistributing via eigrp 100
  Last update from 155.1.58.5 on Vlan58, 01:19:03 ago
  Routing Descriptor Blocks:
  * 155.1.58.5, from 155.1.58.5, 01:19:03 ago, via Vlan58
      Route metric is 130816, traffic share count is 1
      Total delay is 5010 microseconds, minimum bandwidth is 1000000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
```

Likewise R5 only has a default route to 150.1.4.4, while 150.1.5.5 is directly connected.

```
Rack1R5#show ip route 150.1.4.4
% Subnet not in table

Rack1R5#show ip route 150.1.5.5
Routing entry for 150.1.5.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Redistributing via eigrp 100
  Routing Descriptor Blocks:
  * directly connected, via Loopback0
      Route metric is 0, traffic share count is 1
```

Based on this current routing information SW2 has reachability to both of these destinations.

```
Rack1SW2#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.58.5 0 msec 0 msec 0 msec
  2 155.1.0.4 34 msec *  25 msec

Rack1SW2#traceroute 150.1.5.5

Type escape sequence to abort.
Tracing the route to 150.1.5.5

  1 155.1.58.5 0 msec *  0 msec
```

Copyright © 2010 Internetwork Expert

Once R5 advertises the summary 150.1.4.0/23 SW2 loses its more specific route to 150.1.5.0/24, but gains a longer match to 150.1.4.4.

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface FastEthernet0/0
Rack1R5(config-if)#ip summary-address eigrp 100 150.1.4.0 255.255.254.0

Rack1SW2#show ip route 150.1.4.4
Routing entry for 150.1.4.0/23
  Known via "eigrp 100", distance 90, metric 130816, type internal
  Redistributing via eigrp 100
  Last update from 155.1.58.5 on Vlan58, 00:00:12 ago
  Routing Descriptor Blocks:
  * 155.1.58.5, from 155.1.58.5, 00:00:12 ago, via Vlan58
      Route metric is 130816, traffic share count is 1
      Total delay is 5010 microseconds, minimum bandwidth is 1000000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1

Rack1SW2#show ip route 150.1.5.5
Routing entry for 150.1.4.0/23
  Known via "eigrp 100", distance 90, metric 130816, type internal
  Redistributing via eigrp 100
  Last update from 155.1.58.5 on Vlan58, 00:00:16 ago
  Routing Descriptor Blocks:
  * 155.1.58.5, from 155.1.58.5, 00:00:16 ago, via Vlan58
      Route metric is 130816, traffic share count is 1
      Total delay is 5010 microseconds, minimum bandwidth is 1000000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
```

Since R5 previously only had a default route to reach 150.1.4.4, the longer match is now the summary to Null0.

```
Rack1R5#show ip route 150.1.4.4
Routing entry for 150.1.4.0/23
  Known via "eigrp 100", distance 5, metric 128256, type internal
  Redistributing via eigrp 100
  Routing Descriptor Blocks:
  * directly connected, via Null0
      Route metric is 128256, traffic share count is 1
      Total delay is 5000 microseconds, minimum bandwidth is 10000000 Kbit
      Reliability 255/255, minimum MTU 1514 bytes
      Loading 1/255, Hops 0
```

Copyright © 2010 Internetwork Expert

The longer match for 150.1.5.5 remains the connected interface.

```
Rack1R5#show ip route 150.1.5.5
Routing entry for 150.1.5.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Redistributing via eigrp 100
  Routing Descriptor Blocks:
  * directly connected, via Loopback0
      Route metric is 0, traffic share count is 1
```

This implies that R5 can forward traffic for 150.1.5.5, but traffic for 150.1.4.4 will be Null routed (dropped).

```
Rack1SW2#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.58.5 0 msec 0 msec 8 msec
  2 155.1.58.5 !H  *  !H

Rack1SW2#traceroute 150.1.5.5

Type escape sequence to abort.
Tracing the route to 150.1.5.5

  1 155.1.58.5 8 msec *  0 msec
```

To resolve this a static route with a lower administrative distance than the summary is installed in the routing table of R5. This static route tells R5 to forward traffic that matches the summary towards R4.

```
Rack1R5#show ip route | include 150.1.4.0
D       150.1.4.0/23 is a summary, 00:01:14, Null0

Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#ip route 150.1.4.0 255.255.254.0 155.1.0.4
Rack1R5(config)#end

Rack1R5#show ip route | include 150.1.4.0
S       150.1.4.0/23 [1/0] via 155.1.0.4

Rack1SW2#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.58.5 0 msec 8 msec 0 msec
  2 155.1.0.4 25 msec *  25 msec
```

Copyright © 2010 Internetwork Expert

5.12 EIGRP Poisoned Floating Summarization

- Remove the previously configured static route on R5.
- Modify the administrative distance of the summary that R5 is generating to SW2 so that a route to Null0 is not installed.

Configuration

```
R5:
interface FastEthernet0/0
 ip summary-address eigrp 100 150.1.4.0 255.255.254.0 255
```

Verification

> **Note**
>
> Routes with an administrative distance of 255 are not candidate to be installed in the routing table. By poisoning the interface level summary on R5 with a distance of 255, the route to Null0 cannot be installed locally in the routing table, but the summary itself can be advertised out the interface. The use of this configuration design-wise is in cases where you want the router to forward traffic for destinations inside the summary that it does not have a longer match for.
>
> In this case we can see that SW2 has the route 150.1.4.0/23 to reach 150.1.4.4. However since R4 is only advertising a default route to R5, R5 has no longer match for 150.1.4.4. In the previous case R5's longer match to 150.1.4.4 was its own summary to Null0, meaning that all traffic going to 150.1.4.4 was dropped. By poisoning the summary with a distance of 255 R5 can now use the default route to reach 150.1.4.4.

```
Rack1SW2#show ip route 150.1.4.4
Routing entry for 150.1.4.0/23
  Known via "eigrp 100", distance 90, metric 130816, type internal
  Redistributing via eigrp 100
  Last update from 155.1.58.5 on Vlan58, 00:00:46 ago
  Routing Descriptor Blocks:
  * 155.1.58.5, from 155.1.58.5, 00:00:46 ago, via Vlan58
      Route metric is 130816, traffic share count is 1
      Total delay is 5010 microseconds, minimum bandwidth is 1000000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1

Rack1R5#show ip route 150.1.4.4
% Subnet not in table

Rack1SW2#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.58.5 0 msec 0 msec 9 msec
  2 155.1.0.4 25 msec *  25 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Note

Erase and reload all devices and load the *Basic EIGRP Routing* initial configurations before continuing.

5.13 EIGRP Metric Weights

- Configure all devices in EIGRP AS 100 so that only delay is used in the composite metric calculation.

Configuration

```
R1 - R6, SW1 - SW4:
router eigrp 100
 metric weights 0 0 0 1 0 0
```

Verification

Note

By default EIGRP uses bandwidth and delay to calculate its composite metric. Load and reliability can also be used, or the ratio at which bandwidth and delay are used can be changed, by modifying the **`metric weights`**. The default weighting of K1 and K3 mean that only bandwidth and delay are used. Specifically the calculation is as follows:

```
metric = [k1 * bandwidth + (k2 * bandwidth)/(256 - load) +
k3 * delay] * [k5/(reliability + k4)]
```

If k5 equals zero the second half of the equation is ignored. Bandwidth is the inverse minimum bandwidth along the path scaled by $2.56 * 10^{12}$. Delay is 10s of microseconds scaled by 256.

The weighting of the metrics can be seen from the **`show ip protocols`** command.

Copyright © 2010 Internetwork Expert

```
Rack1SW3#show ip protocols
*** IP Routing is NSF aware ***

Routing Protocol is "eigrp 100"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Default networks flagged in outgoing updates
  Default networks accepted from incoming updates
  EIGRP metric weight K1=0, K2=0, K3=1, K4=0, K5=0
  EIGRP maximum hopcount 100
  EIGRP maximum metric variance 1
  Redistributing: eigrp 100
  EIGRP NSF-aware route hold timer is 240s
  Automatic network summarization is not in effect
<output omitted>
```

show ip eigrp topology shows the individual vector metrics that are used in the composite calculation.

```
Rack1SW3#show ip eigrp topology 150.1.9.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 150.1.9.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is
128000
  Routing Descriptor Blocks:
  0.0.0.0 (Loopback0), from Connected, Send flag is 0x0
      Composite metric is (128000/0), Route is Internal
      Vector metric:
        Minimum bandwidth is 10000000 Kbit
        Total delay is 5000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1514
        Hop count is 0
```

For the connected Loopback 0 network of SW3 the total delay is 5000 microseconds. 500 tens of microseconds scaled by 256 equals the total composite metric of 128,000. This indicates that only delay is weighted in the calculation.

Copyright © 2010 Internetwork Expert

Pitfall

The metric weights must match in order for EIGRP adjacency to form.

```
Rack1SW3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW3(config)#router eigrp 100
Rack1SW3(config-router)#metric weights 0 1 1 1 1 1
Rack1SW3(config-router)#end
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.79.7 (Vlan79) is
down: metric changed
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.79.7 (Vlan79) is
down: K-value mismatch
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.79.7 (Vlan79) is
down: K-value mismatch
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

5.14 EIGRP Traffic Engineering with Metric

- Configure a metric manipulation on SW1 so that traffic from SW3 to the Loopback 0 network of R6 transits the link between R3 and R1.

Configuration

```
SW1:
interface Vlan67
 delay 100000
```

Verification

> **Note**
>
> Before any metric manipulation, SW3's traffic to R6 is sent to SW1, then directly to R6.

```
Rack1SW3#traceroute 150.1.6.6

Type escape sequence to abort.
Tracing the route to 150.1.6.6

  1 155.1.79.7 4 msec 0 msec 0 msec
  2 155.1.67.6 4 msec *  0 msec
```

This is based on the fact that SW1 installs the route to 150.1.6.6 via 155.1.67.6.

```
Rack1SW1#show ip route 150.1.6.6
Routing entry for 150.1.6.0/24
  Known via "eigrp 100", distance 90, metric 128256, type internal
  Redistributing via eigrp 100
  Last update from 155.1.67.6 on Vlan67, 00:14:50 ago
  Routing Descriptor Blocks:
  * 155.1.67.6, from 155.1.67.6, 00:14:50 ago, via Vlan67
      Route metric is 128256, traffic share count is 1
      Total delay is 5010 microseconds, minimum bandwidth is 1000000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
```

Copyright © 2010 Internetwork Expert

Also note that SW1 does not know the alternate route through R3.

```
Rack1SW1#show ip eigrp topology 150.1.6.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 150.1.6.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 128256
  Routing Descriptor Blocks:
  155.1.67.6 (Vlan67), from 155.1.67.6, Send flag is 0x0
      Composite metric is (128256/128000), Route is Internal
      Vector metric:
        Minimum bandwidth is 1000000 Kbit
        Total delay is 5010 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
```

This is due to the fact that R3 is choosing SW1's route as the successor. Since only the successor is candidate to be advertised, and since split-horizon is enabled on R3's link to SW1, this route will not advertise this route back to SW1.

```
Rack1R3#show ip eigrp topology 150.1.6.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 150.1.6.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 130816
  Routing Descriptor Blocks:
  155.1.37.7 (FastEthernet0/0), from 155.1.37.7, Send flag is 0x0
      Composite metric is (130816/128256), Route is Internal
      Vector metric:
        Minimum bandwidth is 100000 Kbit
        Total delay is 5110 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
  155.1.0.5 (Serial1/0.1), from 155.1.0.5, Send flag is 0x0
      Composite metric is (1154560/642560), Route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 45100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 3
  155.1.13.1 (Serial1/2), from 155.1.13.1, Send flag is 0x0
      Composite metric is (642560/130560), Route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 25100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
```

Copyright © 2010 Internetwork Expert

In order for R3 to advertise the alternate path to SW1, R3 must see a better composite metric through R1 than it does through SW1. This can be accomplished by altering the advertised distance of the route from SW1 to R3 by changing the delay.

```
Rack1SW1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface Vlan 67
Rack1SW1(config-if)#delay 100000
Rack1SW1(config-if)#end
```

The EIGRP neighbors must then be cleared to recalculate DUAL.

```
Rack1SW1#clear ip eigrp neighbors
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.37.3
(FastEthernet0/3) is down: manually cleared
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.67.6 (Vlan67) is
down: manually cleared
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.79.9 (Vlan79) is
down: manually cleared
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.79.9 (Vlan79) is up:
new adjacency
```

SW1 now chooses R3's route as the successor, as the composite result 645,120 is lower than 25,728,000.

```
Rack1SW1#show ip eigrp topology 150.1.6.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 150.1.6.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 645120
  Routing Descriptor Blocks:
  155.1.37.3 (FastEthernet0/3), from 155.1.37.3, Send flag is 0x0
      Composite metric is (645120/642560), Route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 25200 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 3
  155.1.67.6 (Vlan67), from 155.1.67.6, Send flag is 0x0
      Composite metric is (25728000/128000), Route is Internal
      Vector metric:
        Minimum bandwidth is 1000000 Kbit
        Total delay is 1005000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
```

Copyright © 2010 Internetwork Expert

SW1 now routes through R3 to reach 150.1.6.6, which is reflected in both the routing table output of SW1 and the traceroute output of SW3.

```
Rack1SW1#show ip route 150.1.6.6
Routing entry for 150.1.6.0/24
  Known via "eigrp 100", distance 90, metric 645120, type internal
  Redistributing via eigrp 100
  Last update from 155.1.37.3 on FastEthernet0/3, 00:02:14 ago
  Routing Descriptor Blocks:
  * 155.1.37.3, from 155.1.37.3, 00:02:14 ago, via FastEthernet0/3
      Route metric is 645120, traffic share count is 1
      Total delay is 25200 microseconds, minimum bandwidth is 128 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 3


Rack1SW3#traceroute 150.1.6.6

Type escape sequence to abort.
Tracing the route to 150.1.6.6

  1 155.1.79.7 0 msec 0 msec 0 msec
  2 155.1.37.3 4 msec 0 msec 0 msec
  3 155.1.13.1 4 msec 8 msec 8 msec
  4 155.1.146.6 8 msec *  4 msec
```

Copyright © 2010 Internetwork Expert

5.15 EIGRP Unequal Cost Load Balancing

- Configure unequal cost load balancing so that traffic from R6 going to VLAN 9 is load balanced between R1 and SW1.
- The traffic share should be configured in such a way that the link to SW1 is used five times as much as the link to R1.
- Verify this by configuring per-packet load balancing on R6.

Configuration

```
R1:
interface Serial0/1
 delay 1

R3:
interface FastEthernet0/0
 delay 1

R6:
interface FastEthernet0/0.146
 delay 56
!
router eigrp 100
 variance 128
```

Verification

> **Note**
>
> Previously the **`metric weights`** command was updated on all devices in EIGRP AS 100 so that only delay was weighted. Therefore based on the interface delay values from R6 outbound towards VLAN 9 we can calculate how traffic will be routed. Recall that the delay value used in the composite calculation is *tens of microseconds scaled by 256*.

Copyright © 2010 Internetwork Expert

To start, without any configuration changes, the path from R6 to SW3 has the following delays.

```
Rack1R6#show interface FastEthernet0/0.67 | include DLY
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,

Rack1SW1#show interface Vlan79 | include DLY
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,

Rack1SW3#show interface Vlan9 | include DLY
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
```

The path from R6 -> SW1 -> SW3 therefore has a total delay of 120 microseconds. 12 *tens of microseconds* scaled by 256 gives us a composite metric of 3,072.

This path is then compared to the one R6 -> R1 -> R3 -> SW1 -> SW3 with the following delays.

```
Rack1R6#show interface FastEthernet0/0.146 | include DLY
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,

Rack1R1#show interface Serial0/1 | include DLY
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,

Rack1R3#show interface FastEthernet0/0 | include DLY
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,

Rack1SW1#show interface Vlan79 | include DLY
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,

Rack1SW3#show interface Vlan9 | include DLY
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
```

This path has a total delay of 20,220 microseconds. 2,022 *tens of microseconds* scaled by 256 gives us a composite metric of 517,632. Since 3,072 is lower than 517,632, the Successor is the route from R6 to SW1. This can be verified from the topology view of R6.

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip eigrp topology 155.1.9.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 155.1.9.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 3072
  Routing Descriptor Blocks:
  155.1.67.7 (FastEthernet0/0.67), from 155.1.67.7, Send flag is 0x0
      Composite metric is (3072/512), Route is Internal
      Vector metric:
        Minimum bandwidth is 100000 Kbit
        Total delay is 120 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
```

In order to consider the route from R6 to R1 for load balancing, the route first must pass the Feasibility Condition. Again the Feasibility Condition states that if the Advertised Distance of an alternate route is lower than the Feasible Distance of the Successor, the route is a loop free path and can be considered for load balancing. In other words if R1's metric to reach SW3 is lower than R6's metric to reach SW3, R6 can assume that R1 is closer to SW3, and is a loop free path.

The Advertised Distance that R1 would be sending to R6 is based on these interfaces in the transit path.

```
Rack1R1#show interface Serial0/1 | include DLY
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,

Rack1R3#show interface FastEthernet0/0 | include DLY
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,

Rack1SW1#show interface Vlan79 | include DLY
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,

Rack1SW3#show interface Vlan9 | include DLY
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
```

The total delay of this path is 20,120 microseconds, or 2,012 *tens of microseconds*. Scaled by 256 R1 would be advertising 515,072. Since 515,072 is greater than 3,072, R6's Feasible Distance, this path cannot be considered a Feasible Successor. Therefore the first step in doing unequal cost load balancing is to lower what R1 is advertising as its metric. In this example this is accomplished by changing the delay of R1's link to R3 and R3's link to SW1 to 10 microseconds.

Copyright © 2010 Internetwork Expert

```
Rack1R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#interface Serial0/1
Rack1R1(config-if)#delay 1
Rack1R1(config-if)#

Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#interface FastEthernet0/0
Rack1R3(config-if)#delay 1
Rack1R3(config-if)#
```

These new delay values update the entire path as follows.

```
Rack1R6#show interface FastEthernet0/0.146 | include DLY
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,

Rack1R1#show interface Serial0/1 | include DLY
  MTU 1500 bytes, BW 1544 Kbit, DLY 10 usec,

Rack1R3#show interface FastEthernet0/0 | include DLY
  MTU 1500 bytes, BW 100000 Kbit, DLY 10 usec,

Rack1SW1#show interface Vlan79 | include DLY
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,

Rack1SW3#show interface Vlan9 | include DLY
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
```

The total delay of the path is now 140 microseconds. 14 *tens of microseconds* scaled by 256 equals a total composite of 3,584. Since this is still higher than 3,072, this path is not the Successor. However R1's Advertised Distance is now a total delay of 40 microseconds. 4 *tens of microseconds* scaled by 256 equals a total Advertised Distance of 1024. Since 1,024 is lower than 3,072, R6's Feasible Distance, this route is now a Feasible Successor.

Copyright © 2010 Internetwork Expert

If the variance command were configured on R6 this path would now be installed in the routing table for load balancing. The actual value of variance is arbitrary, as long as the Feasible Distance, 3,072, times the variance is greater than the total composite metric through R1.

```
Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#router eigrp 100
Rack1R6(config-router)#variance 128
Rack1R6(config-router)#end

Rack1R6#show ip route 155.1.9.9
Routing entry for 155.1.9.0/24
  Known via "eigrp 100", distance 90, metric 3072, type internal
  Redistributing via eigrp 100, eigrp 10
  Advertised by eigrp 10
  Last update from 155.1.146.1 on FastEthernet0/0.146, 00:00:03 ago
  Routing Descriptor Blocks:
    155.1.146.1, from 155.1.146.1, 00:00:03 ago, via FastEthernet0/0.146
      Route metric is 3584, traffic share count is 103
      Total delay is 140 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 4
  * 155.1.67.7, from 155.1.67.7, 00:00:03 ago, via FastEthernet0/0.67
      Route metric is 3072, traffic share count is 120
      Total delay is 120 microseconds, minimum bandwidth is 100000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
```

These paths are now balanced 103:120. To achieve the desired 1:5 traffic share, R6's delay on the link to R1 must be updated. The actual values used on R1, R3, and R6 for delay can have multiple valid options as long as two conditions are true. First, the Advertised Distance R1 sends to R6 must be lower than R6's Feasible Distance. Secondly the entire composite result R6 calculates through R1 should be five times the Feasible Distance.

In our case R1's Advertised Distance is 40 microseconds, or 4 *tens of microseconds*. This specifically means the following must be true if we want a traffic share of 1:5.

```
3072 * 5 = (R6_TO_R1_DLY + 4) * 256
```

Therefore R6's delay to R1 should be 56 *tens of microseconds*.

Copyright © 2010 Internetwork Expert

```
Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface FastEthernet0/0.146
Rack1R6(config-subif)#delay 56
Rack1R6(config-subif)#end

Rack1R6#show ip route 155.1.9.9
Routing entry for 155.1.9.0/24
  Known via "eigrp 100", distance 90, metric 3072, type internal
  Redistributing via eigrp 100, eigrp 10
  Advertised by eigrp 10
  Last update from 155.1.146.1 on FastEthernet0/0.146, 00:00:05 ago
  Routing Descriptor Blocks:
    155.1.146.1, from 155.1.146.1, 00:00:05 ago, via FastEthernet0/0.146
      Route metric is 15360, traffic share count is 1
      Total delay is 600 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 4
  * 155.1.67.7, from 155.1.67.7, 00:00:05 ago, via FastEthernet0/0.67
      Route metric is 3072, traffic share count is 5
      Total delay is 120 microseconds, minimum bandwidth is 100000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
```

To test that traffic is actually sent in this distribution CEF is disabled on R6, and traffic is load balanced per packet. Next access-lists are used inbound on R1 and SW1 to count how many ICMP packets they receive from R6 going to VLAN 9.

```
Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface FastEthernet0/0.67
Rack1R6(config-subif)#no ip route-cache
Rack1R6(config-subif)#ip load-sharing per-packet
Rack1R6(config-subif)#interface FastEthernet0/0.146
Rack1R6(config-subif)#no ip route-cache
Rack1R6(config-subif)#ip load-sharing per-packet

Rack1R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#access-list 100 permit icmp any host 155.1.9.9 log
Rack1R1(config)#access-list 100 permit ip any any
Rack1R1(config)#interface FastEthernet0/0
Rack1R1(config-if)#ip access-group 100 in

Rack1SW1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#access-list 100 permit icmp any host 155.1.9.9 log
Rack1SW1(config)#access-list 100 permit ip any any
Rack1SW1(config)#interface Vlan67
Rack1SW1(config-if)#ip access-group 100 in
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#ping 155.1.9.9 repeat 600

Type escape sequence to abort.
Sending 600, 100-byte ICMP Echos to 155.1.9.9, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Copyright © 2010 Internetwork Expert

5.16 EIGRP Convergence Timers

- Configure R1 through R6 so that EIGRP hellos are sent every one second; these devices should inform their neighbors to declare them down if subsequent hellos are not received within three seconds.
- Configure SW1 through SW4 so that EIGRP hellos are sent every ten seconds; these devices should inform their neighbors to declare them down if subsequent hellos are not received within thirty seconds.
- Additionally configure AS 100 so that lost routes are considered Stuck In Active if a query response has not been heard within one minute.

Configuration

```
R1:
interface FastEthernet0/0
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
interface Serial0/0.1
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
interface Serial0/1
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
router eigrp 100
 timers active-time 1

R2:
interface FastEthernet0/0
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
interface Serial0/0.1
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
interface Serial0/1
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
router eigrp 100
 timers active-time 1
```

Copyright © 2010 Internetwork Expert

```
R3:
interface FastEthernet0/0
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
interface Serial1/0.1
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
interface Serial1/2
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
interface Serial1/3
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
router eigrp 100
 timers active-time 1

R4:
interface FastEthernet0/1
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
interface Serial0/0/0.1
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
interface Serial0/1/0
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
router eigrp 100
 timers active-time 1

R5:
interface FastEthernet0/0
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
interface Serial0/0/0
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
interface Serial0/1/0
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
```

Copyright © 2010 Internetwork Expert

```
R6:
interface FastEthernet0/0.67
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
interface FastEthernet0/0.146
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 3
!
router eigrp 100
 timers active-time 1

SW1:
interface FastEthernet0/3
 ip hello-interval eigrp 100 10
 ip hold-time eigrp 100 30
!
interface Vlan67
 ip hello-interval eigrp 100 10
 ip hold-time eigrp 100 30
!
interface Vlan79
 ip hello-interval eigrp 100 10
 ip hold-time eigrp 100 30
!
router eigrp 100
 timers active-time 1

SW2:
interface Vlan58
 ip hello-interval eigrp 100 10
 ip hold-time eigrp 100 30
!
interface Port-channel1
 ip hello-interval eigrp 100 10
 ip hold-time eigrp 100 30
!
router eigrp 100
 timers active-time 1

SW3:
interface Vlan79
 ip hello-interval eigrp 100 10
 ip hold-time eigrp 100 30
!
router eigrp 100
 timers active-time 1

SW4:
interface Port-channel1
 ip hello-interval eigrp 100 10
 ip hold-time eigrp 100 30
!
router eigrp 100
 timers active-time 1
```

Copyright © 2010 Internetwork Expert

Verification

Note

Unlike OSPF, EIGRP hello intervals do not need to match in order to form adjacencies. Instead, the neighbor sending the hello packet tells the adjacent router what its hold time is for that particular hello.

In this case R3 has its hello and dead intervals configured as 1 and 3, while SW1 has them configured as 10 and 30. This means that R3 will be expecting a hello to come in from SW1 within 3 seconds, while R3 will be expecting a hello to come in from SW1 within 30 seconds. In most designs the hello and dead intervals will be set identical on both ends of the link, however as we can see from this example it is not technically required.

```
Rack1R3#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   155.1.0.5               Se1/0.1           2 00:40:50  422  2532  0  343
1   155.1.13.1              Se1/2             2 00:42:22   41  1140  0  208
3   155.1.23.2              Se1/3             2 00:45:58  101  1140  0  133
2   155.1.37.7              Fa0/0            23 00:46:01    3   200  0  194

Rack1SW1#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
Type
                                            (sec)         (ms)       Cnt Num
2   155.1.67.6              Vl67              2 00:00:53   17   200  0  162
0   155.1.79.9              Vl79             28 00:46:06    3   200  0  62
1   155.1.37.3              Fa0/3             2 00:46:08    1   200  0  248
```

Copyright © 2010 Internetwork Expert

The **`timers active-time`** command controls how long an EIGRP router will wait for a reply to a query message before considering the route Stuck In Active (SIA), and declaring the neighbor down that a reply was not received from.

The query and reply process is used to discover alternate paths to a route for which the successor is lost. In the below case SW4 loses the successor for 155.1.10.0/24 when the Vlan10 interface is shutdown. This causes it to send an EIGRP query message out to its neighbor, SW2.

```
Rack1SW3#debug eigrp packet terse
EIGRP Packets debugging is on
    (UPDATE, REQUEST, QUERY, REPLY, IPXSAP, PROBE, ACK, STUB, SIAQUERY,
SIAREPLY)

Rack1SW4#debug eigrp packet terse
EIGRP Packets debugging is on
    (UPDATE, REQUEST, QUERY, REPLY, IPXSAP, PROBE, ACK, STUB, SIAQUERY,
SIAREPLY)
Rack1SW4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW4(config)#interface Vlan10
Rack1SW4(config-if)#shutdown
Rack1SW4(config-if)#
Jul 10 10:22:16: EIGRP: Enqueueing QUERY on Port-channel1 iidbQ un/rely 0/1
serno 453-453
Jul 10 10:22:16: EIGRP: Sending QUERY on Port-channel1
Jul 10 10:22:16:   AS 100, Flags 0x0, Seq 69/0 idbQ 0/0 iidbQ un/rely 0/0 serno
453-453
```

SW2 acknowledges the reception of the query with an ACK to SW4, and the query is continued to be forwarded.

```
Rack1SW4(config-if)#
Jul 10 10:22:16: EIGRP: Received ACK on Port-channel1 nbr 155.1.108.8
Jul 10 10:22:16:   AS 100, Flags 0x0, Seq 0/69 idbQ 0/0 iidbQ un/rely 0/0 peerQ
un/rely 0/1
Jul 10 10:22:16: EIGRP: Port-channel1 multicast flow blocking cleared
```

Within one second the query reaches the far end of the network at SW3. SW3 then acknowledges to SW1 that it received the query.

```
Rack1SW3#
Jul 10 10:22:17: EIGRP: Received QUERY on Vlan79 nbr 155.1.79.7
Jul 10 10:22:17:   AS 100, Flags 0x0, Seq 209/0 idbQ 0/0 iidbQ un/rely 0/0
peerQ un/rely 0/0
Jul 10 10:22:17: EIGRP: Enqueueing ACK on Vlan79 nbr 155.1.79.7
Jul 10 10:22:17:   Ack seq 209 iidbQ un/rely 0/0 peerQ un/rely 1/0
Jul 10 10:22:17: EIGRP: Sending ACK on Vlan79 nbr 155.1.79.7
Jul 10 10:22:17:   AS 100, Flags 0x0, Seq 0/209 idbQ 0/0 iidbQ un/rely 0/0
peerQ un/rely 1/0
```

Copyright © 2010 Internetwork Expert

Since SW3 does not have any other neighbors to send the query to, and it does not have an alternate route to 155.1.10.0/24, it replies to SW1 telling it that it does not have another path. SW1 then acknowledges the query reply to SW3, and sends its own replies back to its other neighbors.

```
Rack1SW3#
Jul 10 10:22:17: EIGRP: Enqueueing REPLY on Vlan79 nbr 155.1.79.7 iidbQ un/rely
0/1 peerQ un/rely 0/0 serno 542-542
Jul 10 10:22:17: EIGRP: Requeued unicast on Vlan79
Jul 10 10:22:17: EIGRP: Sending REPLY on Vlan79 nbr 155.1.79.7
Jul 10 10:22:17:   AS 100, Flags 0x0, Seq 66/209 idbQ 0/0 iidbQ un/rely 0/0
peerQ un/rely 0/1 serno 542-542
Jul 10 10:22:17: EIGRP: Received ACK on Vlan79 nbr 155.1.79.7
Jul 10 10:22:17:   AS 100, Flags 0x0, Seq 0/66 idbQ 0/0 iidbQ un/rely 0/0 peerQ
un/rely 0/1
```

Within two seconds the entire query process is completed on SW4 with the reply coming back from SW2. Incredibly the query and replies are received even before the link up/down message is generated, indicating the immensely fast convergence capability of EIGRP. If a reply had not come back from SW2, SW4 would wait for the **`timers active-time`** to expire. If this timer had expired the route would have been considered SIA, and the neighbor relationship to SW2 would have been reset.

```
Rack1SW4(config-if)#
Jul 10 10:22:17: EIGRP: Received REPLY on Port-channel1 nbr 155.1.108.8
Jul 10 10:22:17:   AS 100, Flags 0x0, Seq 131/69 idbQ 0/0 iidbQ un/rely 0/0
peerQ un/rely 0/0
Jul 10 10:22:17: EIGRP: Enqueueing ACK on Port-channel1 nbr 155.1.108.8
Jul 10 10:22:17:   Ack seq 131 iidbQ un/rely 0/0 peerQ un/rely 1/0
Jul 10 10:22:17: EIGRP: Sending ACK on Port-channel1 nbr 155.1.108.8
Jul 10 10:22:17:   AS 100, Flags 0x0, Seq 0/131 idbQ 0/0 iidbQ un/rely 0/0
peerQ un/rely 1/0
Jul 10 10:22:18: %LINK-5-CHANGED: Interface Vlan10, changed state to
administratively down
Jul 10 10:22:19: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan10,
changed state to down
```

Copyright © 2010 Internetwork Expert

5.17 EIGRP Stub Routing

- Configure the EIGRP stub feature in such a way that SW2 does not receive EIGRP query messages.
- Ensure that all devices in AS 100 still have IP reachability to VLAN 8.

Configuration

```
SW2:
router eigrp 100
 eigrp stub connected
```

Verification

> **Note**
>
> The EIGRP stub feature is used to limit the scope of EIGRP query messages, and to limit what routes a neighbor advertises.

```
Rack1R5#show ip eigrp neighbors detail
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
4   155.1.58.8              Fa0/0             27 00:19:42    6   200  0  172
   Version 12.2/1.2, Retrans: 4, Retries: 0, Prefixes: 3
   Stub Peer Advertising ( CONNECTED ) Routes
   Suppressing queries
5   155.1.45.4              Se0/1/0            225 02:10:18   22   200  0  355
   Version 12.4/1.2, Retrans: 0, Retries: 0, Prefixes: 33
3   155.1.0.4               Se0/0/0              2 02:41:35   41   246  0  356
   Version 12.4/1.2, Retrans: 0, Retries: 0, Prefixes: 33
2   155.1.0.2               Se0/0/0              2 02:41:35   50   300  0  189
   Version 12.4/1.2, Retrans: 0, Retries: 0, Prefixes: 13
1   155.1.0.3               Se0/0/0              2 02:41:35   55   330  0  346
   Version 12.4/1.2, Retrans: 0, Retries: 0, Prefixes: 32
0   155.1.0.1               Se0/0/0              2 02:41:35   44   264  0  295
   Version 12.4/1.2, Retrans: 1, Retries: 0, Prefixes: 33
```

Copyright © 2010 Internetwork Expert

In this case SW2 is configured to only advertise its connected routes to other EIGRP neighbors. This implies that SW4 will not have reachability to any destinations behind SW2, and destinations behind SW2 will not have reachability to SW4.

```
Rack1SW4#show ip route eigrp
     155.1.0.0/24 is subnetted, 4 subnets
D       155.1.8.0 [90/2816] via 155.1.108.8, 00:02:00, Port-channel1
D       155.1.58.0 [90/2816] via 155.1.108.8, 00:02:00, Port-channel1
     150.1.0.0/24 is subnetted, 2 subnets
D       150.1.8.0 [90/130560] via 155.1.108.8, 00:02:00, Port-channel1

Rack1R5#show ip route | include via 155.1.58.8
D       155.1.8.0 [90/2816] via 155.1.58.8, 00:02:28, FastEthernet0/0
D       155.1.108.0 [90/5120] via 155.1.58.8, 00:02:28, FastEthernet0/0
D       150.1.8.0 [90/130560] via 155.1.58.8, 00:02:28, FastEthernet0/0
```

Output from the **debug eigrp packet terse** shows the progression of a QUERY message in the network and its REPLY. First, SW3 shuts down its Vlan9 interface, withdrawing 155.1.9.0/24 and generating a QUERY.

```
Rack1SW3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW3(config)#interface Vlan9
Rack1SW3(config-if)#shutdown
Rack1SW3(config-if)#
EIGRP: Enqueueing QUERY on Vlan79 iidbQ un/rely 0/1 serno 574-574
EIGRP: Sending QUERY on Vlan79
   AS 100, Flags 0x0, Seq 84/0 idbQ 0/0 iidbQ un/rely 0/0 serno 574-574
```

R5 receives the QUERY in from R1, R2, R3, and R4.

```
Rack1R5#
EIGRP: Received QUERY on Serial0/0 nbr 155.1.0.3
  AS 100, Flags 0x0, Seq 344/437 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
EIGRP: Received QUERY on Serial0/0 nbr 155.1.0.2
  AS 100, Flags 0x0, Seq 187/437 idbQ 1/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
EIGRP: Received QUERY on Serial0/1 nbr 155.1.45.4
  AS 100, Flags 0x0, Seq 353/438 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
EIGRP: Received QUERY on Serial0/0 nbr 155.1.0.1
  AS 100, Flags 0x0, Seq 292/437 idbQ 1/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
EIGRP: Received QUERY on Serial0/0 nbr 155.1.0.4
  AS 100, Flags 0x0, Seq 352/437 idbQ 1/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
```

Copyright © 2010 Internetwork Expert

QUERY is forwarded on to all neighbors except the stub neighbor, SW2.

```
Rack1R5#
EIGRP: Sending QUERY on Serial0/0 nbr 155.1.0.4
  AS 100, Flags 0x0, Seq 449/352 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
serno 1262-1263
EIGRP: Sending QUERY on Serial0/0 nbr 155.1.0.2
  AS 100, Flags 0x0, Seq 449/187 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
serno 1262-1263
EIGRP: Sending QUERY on Serial0/0 nbr 155.1.0.3
  AS 100, Flags 0x0, Seq 449/344 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
serno 1262-1263
EIGRP: Sending QUERY on Serial0/1 nbr 155.1.45.4
  AS 100, Flags 0x0, Seq 445/353 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
serno 1262-1263
EIGRP: Sending QUERY on Serial0/0 nbr 155.1.0.1
  AS 100, Flags 0x0, Seq 449/292 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
serno 1262-1263
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

5.18 EIGRP Stub Routing with Leak Map

- Configure the EIGRP stub feature in such a way that R5 does not receive EIGRP query messages.
- R5 should continue to advertise all learned routes with the exception of SW2's Loopback 0 network.

Configuration

```
R5:
ip prefix-list SW2_LOOPBACK seq 5 permit 150.1.8.0/24
!
route-map STUB_LEAK_MAP deny 10
 match ip address prefix-list SW2_LOOPBACK
!
route-map STUB_LEAK_MAP permit 20
!
router eigrp 100
 eigrp stub connected leak-map STUB_LEAK_MAP
```

Verification

Note

The **leak-map** feature of EIGRP stub, like the leak-map for EIGRP summarization, allows the advertisement of routes that would normally be suppressed. When R5 is configured with only the **eigrp stub** command, it cannot be used as transit. This can be seen from the routing table views of SW2 and R3.

```
Rack1R3#show ip route | include via 155.1.0.5
D       155.1.5.0 [90/514560] via 155.1.0.5, 00:00:16, Serial1/0.1
D       155.1.58.0 [90/514560] via 155.1.0.5, 00:00:16, Serial1/0.1
D       155.1.45.0 [90/1024000] via 155.1.0.5, 00:00:17, Serial1/0.1
D       150.1.5.0 [90/640000] via 155.1.0.5, 00:00:16, Serial1/0.1

Rack1SW2#show ip route | include via 155.1.58.5
D       155.1.0.0 [90/512256] via 155.1.58.5, 00:00:19, Vlan58
D       155.1.5.0 [90/2816] via 155.1.58.5, 00:00:19, Vlan58
D       155.1.45.0 [90/512256] via 155.1.58.5, 00:00:19, Vlan58
D       150.1.5.0 [90/128256] via 155.1.58.5, 00:00:19, Vlan58
```

Copyright © 2010 Internetwork Expert

In this design R5 is configured to leak all dynamically learned routes with the exception of SW2's Loopback. If a failure in the network occurs however, R5 will not receive EIGRP QUERY messages.

```
Rack1R3#show ip eigrp neighbors detail
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   155.1.0.5               Se1/0.1           2 00:00:05  911   5000  0  626
   Version 12.4/1.2, Retrans: 0, Retries: 0, Prefixes: 37
   Stub Peer Advertising ( CONNECTED ) Routes
   Suppressing queries
1   155.1.13.1              Se1/2             2 02:57:20   27   1140  0  383
   Version 12.4/1.2, Retrans: 0, Retries: 0, Prefixes: 24
3   155.1.23.2              Se1/3             2 03:00:57   30   1140  0  285
   Version 12.4/1.2, Retrans: 0, Retries: 0, Prefixes: 20
2   155.1.37.7              Fa0/0            29 03:01:00    1    200  0  407
   Version 12.2/1.2, Retrans: 5, Retries: 0, Prefixes: 5
Rack1R3#

Rack1R3#show ip route | include via 155.1.0.5
D       155.1.8.0 [90/514816] via 155.1.0.5, 00:00:00, Serial1/0.1
D       155.1.5.0 [90/514560] via 155.1.0.5, 00:00:02, Serial1/0.1
D       155.1.58.0 [90/514560] via 155.1.0.5, 00:00:02, Serial1/0.1
D       155.1.45.0 [90/1024000] via 155.1.0.5, 00:00:02, Serial1/0.1
D       155.1.108.0 [90/517120] via 155.1.0.5, 00:00:00, Serial1/0.1
D       150.1.5.0 [90/640000] via 155.1.0.5, 00:00:02, Serial1/0.1

Rack1R3#show ip route 150.1.8.8
% Subnet not in table
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show ip eigrp neighbors detail
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq Type
                                              (sec)         (ms)       Cnt Num
0   155.1.58.5              Vl58             2 00:00:25    1   200  0  633
   Version 12.4/1.2, Retrans: 3, Retries: 0
   Stub Peer Advertising ( CONNECTED REDISTRIBUTED ) Routes
   Suppressing queries
1   155.1.108.10            Po1             20 00:34:21    1   450  0  90
   Version 12.2/1.2, Retrans: 12, Retries: 0

Rack1SW2#show ip route | include via 155.1.58.5
D EX 222.22.2.0/24 [170/537856] via 155.1.58.5, 00:00:18, Vlan58
D EX 204.12.1.0/24 [170/537856] via 155.1.58.5, 00:00:18, Vlan58
D       155.1.146.0 [90/514816] via 155.1.58.5, 00:00:18, Vlan58
D       155.1.23.0 [90/1024256] via 155.1.58.5, 00:00:18, Vlan58
D       155.1.13.0 [90/512512] via 155.1.58.5, 00:00:18, Vlan58
D       155.1.0.0 [90/512256] via 155.1.58.5, 00:00:18, Vlan58
D       155.1.7.0 [90/512768] via 155.1.58.5, 00:00:18, Vlan58
D       155.1.5.0 [90/2816] via 155.1.58.5, 00:00:18, Vlan58
D       155.1.45.0 [90/512256] via 155.1.58.5, 00:00:19, Vlan58
D       155.1.37.0 [90/512512] via 155.1.58.5, 00:00:19, Vlan58
D       155.1.79.0 [90/512768] via 155.1.58.5, 00:00:19, Vlan58
D       155.1.67.0 [90/517376] via 155.1.58.5, 00:00:14, Vlan58
D EX 220.20.3.0/24 [170/537856] via 155.1.58.5, 00:00:19, Vlan58
D EX 200.0.0.0/24 [170/1154816] via 155.1.58.5, 00:00:14, Vlan58
D EX    54.1.1.0 [170/1026816] via 155.1.58.5, 00:00:14, Vlan58
D EX 200.0.1.0/24 [170/1154816] via 155.1.58.5, 00:00:14, Vlan58
D EX 200.0.2.0/24 [170/1154816] via 155.1.58.5, 00:00:14, Vlan58
D EX 200.0.3.0/24 [170/1154816] via 155.1.58.5, 00:00:14, Vlan58
D EX 192.10.1.0/24 [170/537856] via 155.1.58.5, 00:00:19, Vlan58
D EX    31.3.0.0 [170/537856] via 155.1.58.5, 00:00:14, Vlan58
D EX    31.2.0.0 [170/537856] via 155.1.58.5, 00:00:14, Vlan58
D EX    31.1.0.0 [170/537856] via 155.1.58.5, 00:00:14, Vlan58
D EX    31.0.0.0 [170/537856] via 155.1.58.5, 00:00:14, Vlan58
D       150.1.7.0 [90/640512] via 155.1.58.5, 00:00:21, Vlan58
D       150.1.6.0 [90/642816] via 155.1.58.5, 00:00:16, Vlan58
D       150.1.5.0 [90/128256] via 155.1.58.5, 00:00:21, Vlan58
D       150.1.4.0 [90/640256] via 155.1.58.5, 00:00:16, Vlan58
D       150.1.3.0 [90/640256] via 155.1.58.5, 00:00:21, Vlan58
D       150.1.2.0 [90/640256] via 155.1.58.5, 00:00:21, Vlan58
D       150.1.1.0 [90/640256] via 155.1.58.5, 00:00:21, Vlan58
D       150.1.9.0 [90/640768] via 155.1.58.5, 00:00:21, Vlan58
D EX 205.90.31.0/24 [170/537856] via 155.1.58.5, 00:00:21, Vlan58
D EX    30.2.0.0 [170/537856] via 155.1.58.5, 00:00:21, Vlan58
D EX    30.3.0.0 [170/537856] via 155.1.58.5, 00:00:21, Vlan58
D EX    30.0.0.0 [170/537856] via 155.1.58.5, 00:00:21, Vlan58
D EX    30.1.0.0 [170/537856] via 155.1.58.5, 00:00:21, Vlan58
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

5.19 EIGRP Filtering with Passive Interface

- Configure the passive-interface feature on R5, SW2, and SW4 so that EIGRP hello packets are not sent out the LAN segments without routers attached.
- Configure the passive-interface default feature on SW1 and SW3 so that EIGRP hello packets are not sent out the LAN segments without routers attached; ensure that full reachability is maintained after this change is made.

Configuration

```
R5:
router eigrp 100
 passive-interface FastEthernet0/1

SW1:
router eigrp 100
 passive-interface default
 no passive-interface Vlan67
 no passive-interface Vlan79
 no passive-interface FastEthernet0/3

SW2:
router eigrp 100
 passive-interface Vlan8

SW3:
router eigrp 100
 passive-interface default
 no passive-interface Vlan79

SW4:
router eigrp 100
 passive-interface Vlan10
```

Verification

Note

The **passive-interface** command in EIGRP, like in RIPv2, stops the sending of updates out an interface. Unlike RIPv2 however, passive-interface in EIGRP will stop the forming of an adjacency on the interface, and hence the learning of any updates on the link.

The **passive-interface default** command can be used to make all interfaces passive, and then interfaces can have the passive feature selectively disabled with the no passive-interface command.

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show ip protocols
*** IP Routing is NSF aware ***

Routing Protocol is "eigrp 100"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Default networks flagged in outgoing updates
  Default networks accepted from incoming updates
  EIGRP metric weight K1=0, K2=0, K3=1, K4=0, K5=0
  EIGRP maximum hopcount 100
  EIGRP maximum metric variance 1
  Redistributing: eigrp 100
  EIGRP NSF-aware route hold timer is 240s
  Automatic network summarization is not in effect
  Maximum path: 4
  Routing for Networks:
    150.1.0.0
    155.1.0.0
  Passive Interface(s):
    Vlan1
    Vlan7
    FastEthernet0/1
    FastEthernet0/2
    FastEthernet0/4
    FastEthernet0/5
    FastEthernet0/6
    FastEthernet0/7
    FastEthernet0/8
    FastEthernet0/9
    FastEthernet0/10
    FastEthernet0/11
    FastEthernet0/12
    FastEthernet0/13
    FastEthernet0/14
    FastEthernet0/15
    FastEthernet0/16
    FastEthernet0/17
    FastEthernet0/18
    FastEthernet0/19
    FastEthernet0/20
    FastEthernet0/21
    FastEthernet0/22
    FastEthernet0/23
    FastEthernet0/24
    GigabitEthernet0/1
    GigabitEthernet0/2
    Loopback0
  Routing Information Sources:
    Gateway         Distance      Last Update
    155.1.37.3            90      00:01:39
    155.1.79.9            90      00:01:39
    155.1.67.6            90      00:01:39
  Distance: internal 90 external 170
```

5.20 EIGRP Filtering with Prefix-Lists

- Configure a prefix-list on R4 so that it does not advertise the 30.0.0.0 and 31.0.0.0 subnets learned from BB3 out the point-to-point link to R5; use the most efficient list to accomplish this that will not deny any other networks than those subnets R4 is learning.
- Configure a prefix-list on R1 so that it does not install any updates received from R4 on the VLAN 146 segment.

Configuration

```
R1:
router eigrp 100
 distribute-list prefix PERMIT_ALL gateway NOT_FROM_R4 in
!
ip prefix-list NOT_FROM_R4 seq 5 deny 155.1.146.4/32
ip prefix-list NOT_FROM_R4 seq 10 permit 0.0.0.0/0 le 32
!
ip prefix-list PERMIT_ALL seq 5 permit 0.0.0.0/0 le 32

R4:
router eigrp 100
 distribute-list prefix STOP_RIP_SUBNETS out Serial0/1
!
ip prefix-list STOP_RIP_SUBNETS seq 5 deny 30.0.0.0/14 ge 16 le 16
ip prefix-list STOP_RIP_SUBNETS seq 10 deny 31.0.0.0/14 ge 16 le 16
ip prefix-list STOP_RIP_SUBNETS seq 15 permit 0.0.0.0/0 le 32
```

Verification

Note

Before filtering:

```
Rack1R1#show ip route | include 3(0|1).[0-3].0.0
     31.0.0.0/16 is subnetted, 4 subnets
D EX    31.3.0.0 [170/28160] via 155.1.146.4, 00:17:02, FastEthernet0/0
D EX    31.2.0.0 [170/28160] via 155.1.146.4, 00:17:02, FastEthernet0/0
D EX    31.1.0.0 [170/28160] via 155.1.146.4, 00:17:02, FastEthernet0/0
D EX    31.0.0.0 [170/28160] via 155.1.146.4, 00:17:02, FastEthernet0/0
     30.0.0.0/16 is subnetted, 4 subnets
D EX    30.2.0.0 [170/28160] via 155.1.146.4, 00:17:02, FastEthernet0/0
D EX    30.3.0.0 [170/28160] via 155.1.146.4, 00:17:02, FastEthernet0/0
D EX    30.0.0.0 [170/28160] via 155.1.146.4, 00:17:02, FastEthernet0/0
D EX    30.1.0.0 [170/28160] via 155.1.146.4, 00:17:02, FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip route | include via 155.1.(0|45).4
D EX 204.12.1.0/24 [170/537600] via 155.1.45.4, 00:21:37, Serial0/1/0
                   [170/537600] via 155.1.0.4, 00:21:37, Serial0/0/0
D       155.1.146.0 [90/514560] via 155.1.45.4, 00:21:37, Serial0/1/0
                    [90/514560] via 155.1.0.4, 00:21:37, Serial0/0/0
D       155.1.67.0 [90/517120] via 155.1.45.4, 00:21:36, Serial0/1/0
                   [90/517120] via 155.1.0.4, 00:21:36, Serial0/0/0
D EX 200.0.0.0/24 [170/1154560] via 155.1.45.4, 00:21:36, Serial0/1/0
                  [170/1154560] via 155.1.0.4, 00:21:36, Serial0/0/0
D EX    54.1.1.0 [170/1026560] via 155.1.45.4, 00:21:37, Serial0/1/0
                 [170/1026560] via 155.1.0.4, 00:21:37, Serial0/0/0
D EX 200.0.1.0/24 [170/1154560] via 155.1.45.4, 00:21:37, Serial0/1/0
                  [170/1154560] via 155.1.0.4, 00:21:37, Serial0/0/0
D EX 200.0.2.0/24 [170/1154560] via 155.1.45.4, 00:21:37, Serial0/1/0
                  [170/1154560] via 155.1.0.4, 00:21:37, Serial0/0/0
D EX 200.0.3.0/24 [170/1154560] via 155.1.45.4, 00:21:37, Serial0/1/0
                  [170/1154560] via 155.1.0.4, 00:21:37, Serial0/0/0
D EX    31.3.0.0 [170/537600] via 155.1.45.4, 00:03:52, Serial0/1/0
                 [170/537600] via 155.1.0.4, 00:03:52, Serial0/0/0
D EX    31.2.0.0 [170/537600] via 155.1.45.4, 00:03:52, Serial0/1/0
                 [170/537600] via 155.1.0.4, 00:03:52, Serial0/0/0
D EX    31.1.0.0 [170/537600] via 155.1.45.4, 00:03:52, Serial0/1/0
                 [170/537600] via 155.1.0.4, 00:03:52, Serial0/0/0
D EX    31.0.0.0 [170/537600] via 155.1.45.4, 00:03:52, Serial0/1/0
                 [170/537600] via 155.1.0.4, 00:03:52, Serial0/0/0
D       150.1.6.0 [90/642560] via 155.1.45.4, 00:21:40, Serial0/1/0
                  [90/642560] via 155.1.0.4, 00:21:40, Serial0/0/0
D       150.1.4.0 [90/640000] via 155.1.45.4, 00:21:41, Serial0/1/0
                  [90/640000] via 155.1.0.4, 00:21:41, Serial0/0/0
D EX    30.2.0.0 [170/537600] via 155.1.45.4, 00:03:55, Serial0/1/0
                 [170/537600] via 155.1.0.4, 00:03:55, Serial0/0/0
D EX    30.3.0.0 [170/537600] via 155.1.45.4, 00:03:55, Serial0/1/0
                 [170/537600] via 155.1.0.4, 00:03:55, Serial0/0/0
D EX    30.0.0.0 [170/537600] via 155.1.45.4, 00:03:55, Serial0/1/0
                 [170/537600] via 155.1.0.4, 00:03:55, Serial0/0/0
D EX    30.1.0.0 [170/537600] via 155.1.45.4, 00:03:55, Serial0/1/0
                 [170/537600] via 155.1.0.4, 00:03:55, Serial0/0/0
```

After filtering:

```
Rack1R1#show ip route | include 3(0|1).[0-3].0.0
     31.0.0.0/16 is subnetted, 4 subnets
D EX    31.3.0.0 [170/1049600] via 155.1.0.5, 00:00:04, Serial0/0.1
D EX    31.2.0.0 [170/1049600] via 155.1.0.5, 00:00:04, Serial0/0.1
D EX    31.1.0.0 [170/1049600] via 155.1.0.5, 00:00:04, Serial0/0.1
D EX    31.0.0.0 [170/1049600] via 155.1.0.5, 00:00:04, Serial0/0.1
     30.0.0.0/16 is subnetted, 4 subnets
D EX    30.2.0.0 [170/1049600] via 155.1.0.5, 00:00:04, Serial0/0.1
D EX    30.3.0.0 [170/1049600] via 155.1.0.5, 00:00:04, Serial0/0.1
D EX    30.0.0.0 [170/1049600] via 155.1.0.5, 00:00:04, Serial0/0.1
D EX    30.1.0.0 [170/1049600] via 155.1.0.5, 00:00:04, Serial0/0.1
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip route | include via 155.1.(0|45).4
D EX 204.12.1.0/24 [170/537600] via 155.1.45.4, 00:04:25, Serial0/1/0
                   [170/537600] via 155.1.0.4, 00:04:25, Serial0/0/0
D       155.1.146.0 [90/514560] via 155.1.45.4, 00:04:25, Serial0/1/0
                    [90/514560] via 155.1.0.4, 00:04:25, Serial0/0/0
D       155.1.67.0 [90/517120] via 155.1.45.4, 00:04:24, Serial0/1/0
                   [90/517120] via 155.1.0.4, 00:04:24, Serial0/0/0
D EX 200.0.0.0/24 [170/1154560] via 155.1.45.4, 00:04:24, Serial0/1/0
                  [170/1154560] via 155.1.0.4, 00:04:25, Serial0/0/0
D EX    54.1.1.0 [170/1026560] via 155.1.45.4, 00:04:25, Serial0/1/0
                 [170/1026560] via 155.1.0.4, 00:04:25, Serial0/0/0
D EX 200.0.1.0/24 [170/1154560] via 155.1.45.4, 00:04:25, Serial0/1/0
                  [170/1154560] via 155.1.0.4, 00:04:25, Serial0/0/0
D EX 200.0.2.0/24 [170/1154560] via 155.1.45.4, 00:04:25, Serial0/1/0
                  [170/1154560] via 155.1.0.4, 00:04:25, Serial0/0/0

D EX 200.0.3.0/24 [170/1154560] via 155.1.45.4, 00:04:25, Serial0/1/0
                  [170/1154560] via 155.1.0.4, 00:04:25, Serial0/0/0
D EX    31.3.0.0 [170/537600] via 155.1.0.4, 00:04:26, Serial0/0/0
D EX    31.2.0.0 [170/537600] via 155.1.0.4, 00:04:26, Serial0/0/0
D EX    31.1.0.0 [170/537600] via 155.1.0.4, 00:04:26, Serial0/0/0
D EX    31.0.0.0 [170/537600] via 155.1.0.4, 00:04:26, Serial0/0/0
D       150.1.6.0 [90/642560] via 155.1.45.4, 00:04:26, Serial0/1/0
                  [90/642560] via 155.1.0.4, 00:04:26, Serial0/0/0
D       150.1.4.0 [90/640000] via 155.1.45.4, 00:04:26, Serial0/1/0
                  [90/640000] via 155.1.0.4, 00:04:26, Serial0/0/0
D EX    30.2.0.0 [170/537600] via 155.1.0.4, 00:04:28, Serial0/0/0
D EX    30.3.0.0 [170/537600] via 155.1.0.4, 00:04:28, Serial0/0/0
D EX    30.0.0.0 [170/537600] via 155.1.0.4, 00:04:28, Serial0/0/0
D EX    30.1.0.0 [170/537600] via 155.1.0.4, 00:04:28, Serial0/0/0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

5.21 EIGRP Filtering with Standard Access-Lists

- Configure a one line standard access-list on R6 to filter out all routes coming from BB1 that have an odd number in the third octet.

Configuration

```
R6:
router eigrp 10
 distribute-list 1 in Serial0/0/0
!
access-list 1 permit 0.0.0.0 255.255.254.255
```

Verification

> **Note**
>
> Before filter.

```
Rack1R6#show ip route eigrp 10
D    200.0.0.0/24 [90/2297856] via 54.1.1.254, 00:00:04, Serial0/0/0
D    200.0.1.0/24 [90/2297856] via 54.1.1.254, 00:00:04, Serial0/0/0
D    200.0.2.0/24 [90/2297856] via 54.1.1.254, 00:00:04, Serial0/0/0
D    200.0.3.0/24 [90/2297856] via 54.1.1.254, 00:00:04, Serial0/0/0
```

After filter.

```
Rack1R6#show ip route eigrp 10
D    200.0.0.0/24 [90/2297856] via 54.1.1.254, 00:00:03, Serial0/0/0
D    200.0.2.0/24 [90/2297856] via 54.1.1.254, 00:00:03, Serial0/0/0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

5.22 EIGRP Filtering with Extended Access-Lists

- Shutdown R5's point-to-point link to R4.
- Configure an extended access-list filter on R5 so that traffic for the Loopback 0 networks of R4 and R6 is sent to R2.
- Traffic for the Loopback 0 networks of R1 and R2 should be sent to R3.
- Traffic for the Loopback 0 networks of SW1 and SW3 should be sent to R1.
- This filter should not affect any other updates on this segment.

Configuration

```
R5:
access-list 100 deny   ip host 155.1.0.2 host 150.1.7.0
access-list 100 deny   ip host 155.1.0.3 host 150.1.7.0
access-list 100 deny   ip host 155.1.0.4 host 150.1.7.0
access-list 100 deny   ip host 155.1.0.2 host 150.1.9.0
access-list 100 deny   ip host 155.1.0.3 host 150.1.9.0
access-list 100 deny   ip host 155.1.0.4 host 150.1.9.0
access-list 100 deny   ip host 155.1.0.1 host 150.1.4.0
access-list 100 deny   ip host 155.1.0.3 host 150.1.4.0
access-list 100 deny   ip host 155.1.0.4 host 150.1.4.0
access-list 100 deny   ip host 155.1.0.1 host 150.1.6.0
access-list 100 deny   ip host 155.1.0.3 host 150.1.6.0
access-list 100 deny   ip host 155.1.0.4 host 150.1.6.0
access-list 100 deny   ip host 155.1.0.1 host 150.1.1.0
access-list 100 deny   ip host 155.1.0.2 host 150.1.1.0
access-list 100 deny   ip host 155.1.0.4 host 150.1.1.0
access-list 100 deny   ip host 155.1.0.1 host 150.1.2.0
access-list 100 deny   ip host 155.1.0.2 host 150.1.2.0
access-list 100 deny   ip host 155.1.0.4 host 150.1.2.0
access-list 100 permit ip any any
!
router eigrp 100
 distribute-list 100 in Serial0/0/0
```

Verification

Note

Like RIP, extended access-lists when called as a distribute-list in IGP have a different meaning than in redistribution or as in BGP. With BGP and redistribution the "source" field in the ACL represents the network address, and the "destination" field represents the subnet mask. In IGP distribute-list application the "source" field in the ACL matches the update source of the route, and the "destination" field represents the network address. This implementation allows us to control which networks we are receiving, but more importantly who we are receiving them from. Before the filter is applied, R5 routes as follows.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip route eigrp | include 150.1.
     150.1.0.0/24 is subnetted, 9 subnets
D       150.1.7.0 [90/640256] via 155.1.0.3, 00:43:14, Serial0/0
D       150.1.6.0 [90/642560] via 155.1.0.4, 00:00:20, Serial0/0
D       150.1.4.0 [90/640000] via 155.1.0.4, 00:00:20, Serial0/0
D       150.1.3.0 [90/640000] via 155.1.0.3, 01:07:01, Serial0/0
D       150.1.2.0 [90/640000] via 155.1.0.2, 01:07:01, Serial0/0
D       150.1.1.0 [90/640000] via 155.1.0.1, 01:07:01, Serial0/0
D       150.1.9.0 [90/640512] via 155.1.0.3, 00:43:15, Serial0/0
D       150.1.8.0 [90/130560] via 155.1.58.8, 01:46:12, FastEthernet0/0

Rack1R5#traceroute 150.1.7.7

Type escape sequence to abort.
Tracing the route to 150.1.7.7

  1 155.1.0.3 32 msec 28 msec 32 msec
  2 155.1.37.7 28 msec *  28 msec

Rack1R5#traceroute 150.1.9.9

Type escape sequence to abort.
Tracing the route to 150.1.9.9

  1 155.1.0.3 28 msec 28 msec 32 msec
  2 155.1.37.7 28 msec 32 msec 32 msec
  3 155.1.79.9 28 msec *  32 msec

Rack1R5#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.0.4 32 msec *  28 msec

Rack1R5#traceroute 150.1.6.6

Type escape sequence to abort.
Tracing the route to 150.1.6.6

  1 155.1.0.4 32 msec
    155.1.0.1 28 msec
    155.1.0.4 28 msec
  2 155.1.146.6 28 msec *  28 msec

Rack1R5#traceroute 150.1.1.1

Type escape sequence to abort.
Tracing the route to 150.1.1.1

  1 155.1.0.1 28 msec *  28 msec
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#traceroute 150.1.2.2

Type escape sequence to abort.
Tracing the route to 150.1.2.2

  1 155.1.0.2 32 msec *  28 msec
```

Once the distribute-list is implemented R5 has only one possible way to route to these destinations.

```
Rack1R5#show ip route eigrp | include 150.1.
     150.1.0.0/24 is subnetted, 9 subnets
D       150.1.7.0 [90/640512] via 155.1.0.1, 00:03:20, Serial0/0
D       150.1.6.0 [90/1666560] via 155.1.0.2, 00:03:20, Serial0/0
D       150.1.4.0 [90/26766592] via 155.1.0.2, 00:03:20, Serial0/0
D       150.1.3.0 [90/640000] via 155.1.0.3, 00:03:20, Serial0/0
D       150.1.2.0 [90/1152000] via 155.1.0.3, 00:03:20, Serial0/0
D       150.1.1.0 [90/1152000] via 155.1.0.3, 00:03:20, Serial0/0
D       150.1.9.0 [90/640768] via 155.1.0.1, 00:03:20, Serial0/0
D       150.1.8.0 [90/130560] via 155.1.58.8, 00:03:20, FastEthernet0/0

Rack1R5#traceroute 150.1.7.7

Type escape sequence to abort.
Tracing the route to 150.1.7.7

  1 155.1.0.1 32 msec 28 msec 28 msec
  2 155.1.13.3 36 msec 32 msec 32 msec
  3 155.1.37.7 36 msec *  32 msec

Rack1R5#traceroute 150.1.9.9

Type escape sequence to abort.
Tracing the route to 150.1.9.9

  1 155.1.0.1 28 msec 32 msec 93 msec
  2 155.1.13.3 36 msec 32 msec 32 msec
  3 155.1.37.7 36 msec 32 msec 36 msec
  4 155.1.79.9 36 msec *  32 msec

Rack1R5#traceroute 150.1.4.4

Type escape sequence to abort.
Tracing the route to 150.1.4.4

  1 155.1.0.2 28 msec 28 msec 32 msec
  2 155.1.23.3 32 msec 36 msec 36 msec
  3 155.1.37.7 32 msec 32 msec 32 msec
  4 155.1.67.6 37 msec 36 msec 32 msec
  5 155.1.146.4 36 msec *  32 msec
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#traceroute 150.1.6.6

Type escape sequence to abort.
Tracing the route to 150.1.6.6

  1 155.1.0.2 28 msec 32 msec 28 msec
  2 155.1.23.3 36 msec 36 msec 32 msec
  3 155.1.13.1 36 msec 40 msec 40 msec
  4 155.1.146.6 40 msec *  40 msec

Rack1R5#traceroute 150.1.1.1

Type escape sequence to abort.
Tracing the route to 150.1.1.1

  1 155.1.0.3 32 msec 28 msec 32 msec
  2 155.1.13.1 36 msec *  32 msec

Rack1R5#traceroute 150.1.2.2

Type escape sequence to abort.
Tracing the route to 150.1.2.2

  1 155.1.0.3 28 msec 32 msec 28 msec
  2 155.1.23.2 41 msec *  32 msec
```

Copyright © 2010 Internetwork Expert

5.23 EIGRP Filtering with Offset Lists

- Configure an offset-list on SW1 so traffic destined for R3's Loopback0 network is sent to R6.
- If the link to R6 is down traffic should be rerouted directly to R3.

Configuration

```
SW1:
router eigrp 100
 offset-list 1 in 2147483647 FastEthernet0/3
!
access-list 1 permit 150.1.3.0
```

Verification

> **Note**
>
> Like in RIP, the offset-list feature in EIGRP is used to modify the metric on a per-route basis or a per-interface basis. Before any metric modifications we can see that SW1 is routing directly to R3 to reach 150.1.3.0/24. There are no additional entries in the EIGRP topology table for this prefix since both R6 and SW3 are routing through SW1 to reach it.

```
Rack1SW1#show ip route 150.1.3.3
Routing entry for 150.1.3.0/24
  Known via "eigrp 100", distance 90, metric 130560, type internal
  Redistributing via eigrp 100
  Last update from 155.1.37.3 on FastEthernet0/3, 02:16:47 ago
  Routing Descriptor Blocks:
  * 155.1.37.3, from 155.1.37.3, 02:16:47 ago, via FastEthernet0/3
      Route metric is 130560, traffic share count is 1
      Total delay is 5100 microseconds, minimum bandwidth is 100000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1

Rack1SW1#show ip eigrp topology 150.1.3.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 150.1.3.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 130560
  Routing Descriptor Blocks:
  155.1.37.3 (FastEthernet0/3), from 155.1.37.3, Send flag is 0x0
      Composite metric is (130560/128000), Route is Internal
      Vector metric:
        Minimum bandwidth is 100000 Kbit
        Total delay is 5100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
```

Copyright © 2010 Internetwork Expert

```
Rack1SW1#traceroute 150.1.3.3

Type escape sequence to abort.
Tracing the route to 150.1.3.3

  1 155.1.37.3 9 msec *  0 msec
```

In order for SW1 to route through R6 to reach this destination, the metric must be offset sufficiently so that R6 computes a lower composite metric through R1 than SW1. By offsetting to the maximum value inbound on SW1 from R3 this is ensured. Also since an access-list is used to match just 150.1.3.0, no other prefixes are affected by this traffic engineering.

```
Rack1SW1#show ip eigrp topology 150.1.3.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 150.1.3.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is
25742592
  Routing Descriptor Blocks:
  155.1.67.6 (Vlan67), from 155.1.67.6, Send flag is 0x0
      Composite metric is (25742592/142592), Route is Internal
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 1005570 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 3
  155.1.37.3 (FastEthernet0/3), from 155.1.37.3, Send flag is 0x0
      Composite metric is (2147614207/2147611647), Route is Internal
      Vector metric:
        Minimum bandwidth is 100000 Kbit
        Total delay is 83891179 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1

Rack1SW1#traceroute 150.1.3.3

Type escape sequence to abort.
Tracing the route to 150.1.3.3

  1 155.1.67.6 9 msec 0 msec 0 msec
  2 155.1.146.1 0 msec 0 msec 8 msec
  3 155.1.13.3 8 msec *  8 msec
```

Copyright © 2010 Internetwork Expert

Since the route through R3 is still installed in the topology table it will be used as a backup route if the path through R6 is lost.

```
Rack1SW1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface Vlan67
Rack1SW1(config-if)#shutdown
Rack1SW1(config-if)#end
Rack1SW1#
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.67.6 (Vlan67) is
down: interface down
%SYS-5-CONFIG_I: Configured from console by console
%LINK-5-CHANGED: Interface Vlan67, changed state to administratively
down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan67, changed state
to down

Rack1SW1#show ip eigrp topology 150.1.3.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 150.1.3.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is
2147614207
  Routing Descriptor Blocks:
  155.1.37.3 (FastEthernet0/3), from 155.1.37.3, Send flag is 0x0
      Composite metric is (2147614207/2147611647), Route is Internal
      Vector metric:
        Minimum bandwidth is 100000 Kbit
        Total delay is 83891179 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1

Rack1SW1#traceroute 150.1.3.3

Type escape sequence to abort.
Tracing the route to 150.1.3.3

  1 155.1.37.3 0 msec *  0 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

5.24 EIGRP Filtering with Administrative Distance

- Configure administrative distance filtering on R6 so that it does not install the route to R4's Loopback 0 network.

Configuration

```
R6:
access-list 4 permit 150.1.4.0
!
router eigrp 100
 distance 255 0.0.0.0 255.255.255.255 4
```

Verification

> **Note**
>
> Like in the other IGP protocols, administrative distance can be set on a per-prefix basis in EIGRP. In this example the source address of the route is ignored by matching an address of 0.0.0.0 with the wildcard 255.255.255.255, while access-list 4 matches the route to change the distance for. Since the AD value of 255 is "infinite", the route in question cannot be installed in the routing table or the EIGRP topology.

```
Rack1R6#show ip route 150.1.4.4
Routing entry for 150.1.4.0/24
  Known via "eigrp 100", distance 90, metric 156160, type internal
  Redistributing via eigrp 10, eigrp 100
  Advertised by eigrp 10
  Last update from 155.1.146.4 on FastEthernet0/0.146, 00:27:44 ago
  Routing Descriptor Blocks:
  * 155.1.146.4, from 155.1.146.4, 00:27:44 ago, via FastEthernet0/0.146
      Route metric is 156160, traffic share count is 1
      Total delay is 5100 microseconds, minimum bandwidth is 100000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1

Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#access-list 4 permit 150.1.4.0
Rack1R6(config)#router eigrp 100
Rack1R6(config-router)#distance 255 0.0.0.0 255.255.255.255 4
Rack1R6(config-router)#end
Rack1R6#clear ip route *
Rack1R6#show ip route 150.1.4.4
% Subnet not in table

Rack1R6#show ip eigrp 100 topology 150.1.4.0 255.255.255.0
% IP-EIGRP (AS 100): Route not in topology table
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

5.25 EIGRP Filtering with Per Neighbor AD

- Configure administrative distance filtering on R3 so that traffic destined for SW1's Loopback 0 network is sent towards R1.

Configuration

```
R3:
access-list 7 permit 150.1.7.0
!
router eigrp 100
 distance 255 155.1.37.7 0.0.0.0 7
```

Verification

> **Note**
>
> Prior to any distance modifications R3 routes directly to SW1 to reach 150.1.7.0/24. Based on the routing table and EIGRP topology table we can see that the Feasible Distance is 128256, and the neighbor the route is learned from is 155.1.37.7.

```
Rack1R3#show ip route 150.1.7.7
Routing entry for 150.1.7.0/24
  Known via "eigrp 100", distance 90, metric 128256, type internal
  Redistributing via eigrp 100
  Last update from 155.1.37.7 on FastEthernet0/0, 00:01:32 ago
  Routing Descriptor Blocks:
  * 155.1.37.7, from 155.1.37.7, 00:01:32 ago, via FastEthernet0/0
      Route metric is 128256, traffic share count is 1
      Total delay is 5010 microseconds, minimum bandwidth is 100000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1

Rack1R3#show ip eigrp topology 150.1.7.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 150.1.7.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 128256
  Routing Descriptor Blocks:
  155.1.37.7 (FastEthernet0/0), from 155.1.37.7, Send flag is 0x0
      Composite metric is (128256/128000), Route is Internal
      Vector metric:
        Minimum bandwidth is 100000 Kbit
        Total delay is 5010 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
  155.1.0.5 (Serial1/0.1), from 155.1.0.5, Send flag is 0x0
      Composite metric is (1152512/640512), Route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 45020 microseconds
        Reliability is 255/255
        Load is 1/255
```

Copyright © 2010 Internetwork Expert

```
       Minimum MTU is 1500
Rack1R3#traceroute 150.1.7.7

Type escape sequence to abort.
Tracing the route to 150.1.7.7

  1 155.1.37.7 0 msec *  0 msec
```

As we saw in the previous example administrative distance can be changed on a per-prefix basis. Based on matching who the route is learned from the distance can also be changed on a per-prefix per-neighbor basis.

```
Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#access-list 7 permit 150.1.7.0
Rack1R3(config)#router eigrp 100
Rack1R3(config-router)#distance 255 155.1.37.7 0.0.0.0 7
Rack1R3(config-router)#end
Rack1R3#clear ip route *

Rack1R3#show ip route 150.1.7.7
Routing entry for 150.1.7.0/24
  Known via "eigrp 100", distance 90, metric 645120, type internal
  Redistributing via eigrp 100
  Last update from 155.1.13.1 on Serial1/2, 00:00:01 ago
  Routing Descriptor Blocks:
  * 155.1.13.1, from 155.1.13.1, 00:00:01 ago, via Serial1/2
      Route metric is 645120, traffic share count is 1
      Total delay is 25200 microseconds, minimum bandwidth is 128 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 3
```

Copyright © 2010 Internetwork Expert

Although the composite metric is higher through R1 than it was originally through SW1, the route through SW1 cannot be installed in the topology table because it has an infinite administrative distance. This implies that R3 must route through R1 to reach the destination.

```
Rack1R3#show ip eigrp topology 150.1.7.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 150.1.7.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 645120
  Routing Descriptor Blocks:
  155.1.13.1 (Serial1/2), from 155.1.13.1, Send flag is 0x0
      Composite metric is (645120/133120), Route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 25200 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 3
  155.1.0.5 (Serial1/0.1), from 155.1.0.5, Send flag is 0x0
      Composite metric is (1157120/645120), Route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 45200 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 4

Rack1R3#traceroute 150.1.7.7

Type escape sequence to abort.
Tracing the route to 150.1.7.7

  1 155.1.13.1 16 msec 16 msec 12 msec
  2 155.1.146.6 12 msec 16 msec 16 msec
  3 155.1.67.7 4 msec *  4 msec
```

Pitfall

The administrative distance for EIGRP internal routes can be changed on a per-prefix basis, but external EIGRP routes cannot.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

5.26 EIGRP Filtering with Route Maps

- Configure R4 to redistribute the VLAN 43 subnet into EIGRP with the tag value of 4.
- Configure a route-map filter on R2 that matches this tag value and denies the route from being installed in the routing table.
- Configure a route-map filter on R3 that denies EIGRP routes with a metric in the range of 500,000 – 750,000 from entering the routing table.
- These filters should not impact any other networks advertised by R4 or learned by R2 and R3.

Configuration

```
R2:
router eigrp 100
 distribute-list route-map FILTER_ON_TAGS in
!
route-map FILTER_ON_TAGS deny 10
 match tag 4
!
route-map FILTER_ON_TAGS permit 20

R3:
router eigrp 100
 distribute-list route-map FILTER_ON_METRIC_RANGE in
!
route-map FILTER_ON_METRIC_RANGE deny 10
 match metric 625000 +- 125000
!
route-map FILTER_ON_METRIC_RANGE permit 20

R4:
router eigrp 100
 redistribute rip metric 100000 100 255 1 1500 route-map RIP_TO_EIGRP
!
ip prefix-list VLAN_43 seq 5 permit 204.12.1.0/24
!
route-map RIP_TO_EIGRP permit 10
 match ip address prefix-list VLAN_43
 set tag 4
!
route-map RIP_TO_EIGRP permit 20
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Unlike BGP, filtering with route-maps in IGP is usually limited to redistribution filtering only. However EIGRP now supports route-map filtering as a distribute-list with matches on metric and tag.
>
> Route tags are set at the time of redistribution, and can be used like BGP community values to group prefixes together without having to match on the actual route in a prefix-list or access-list. In this example we can see that R2 and R4 see the prefix 204.12.1.0/24 with a tag of 4 in the topology table. R2 installs this in the routing table until the distribute-list is applied which denies routes with that tag value.

```
Rack1R4#show ip eigrp topology | include tag
P 204.12.1.0/24, 1 successors, FD is 25600, tag is 4

Rack1R2#show ip eigrp topology | include tag
P 204.12.1.0/24, 1 successors, FD is 1049600, tag is 4

Rack1R2#show ip route 204.12.1.0
Routing entry for 204.12.1.0/24
  Known via "eigrp 100", distance 170, metric 1049600
  Tag 4, type external
  Redistributing via eigrp 100, rip
  Advertised by rip metric 1
  Last update from 155.1.0.5 on Serial0/0.1, 00:05:14 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 155.1.0.5, 00:05:14 ago, via Serial0/0.1
      Route metric is 1049600, traffic share count is 1
      Total delay is 41000 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
      Route tag 4

Rack1R2#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R2(config)#route-map FILTER_ON_TAGS deny 10
Rack1R2(config-route-map)#match tag 4
Rack1R2(config-route-map)#route-map FILTER_ON_TAGS permit 20
Rack1R2(config-route-map)#router eigrp 100
Rack1R2(config-router)#distribute-list route-map FILTER_ON_TAGS in
Rack1R2(config-router)#end
Rack1R2#
```

Copyright © 2010 Internetwork Expert

```
Rack1R2#show ip route 204.12.1.0
% Network not in table

Rack1R2#show ip eigrp topology 204.12.1.0 255.255.255.0
% IP-EIGRP (AS 100): Route not in topology table
```

Other routes learned by EIGRP are not affected by this filter.

```
Rack1R2#show ip route eigrp
     155.1.0.0/24 is subnetted, 13 subnets
D       155.1.146.0 [90/1026560] via 155.1.0.5, 00:05:58, Serial0/0.1
D       155.1.8.0 [90/514816] via 155.1.0.5, 00:09:19, Serial0/0.1
D       155.1.9.0 [90/512768] via 155.1.23.3, 00:09:19, Serial0/1
D       155.1.13.0 [90/1024000] via 155.1.23.3, 00:09:24, Serial0/1
D       155.1.7.0 [90/512512] via 155.1.23.3, 00:09:19, Serial0/1
D       155.1.5.0 [90/514560] via 155.1.0.5, 00:09:19, Serial0/0.1
D       155.1.58.0 [90/514560] via 155.1.0.5, 00:09:19, Serial0/0.1
D       155.1.37.0 [90/512256] via 155.1.23.3, 00:09:19, Serial0/1
D       155.1.79.0 [90/512512] via 155.1.23.3, 00:09:19, Serial0/1
D       155.1.67.0 [90/1029120] via 155.1.0.5, 00:05:58, Serial0/0.1
D       155.1.108.0 [90/517120] via 155.1.0.5, 00:09:19, Serial0/0.1
D EX 200.0.0.0/24 [170/1666560] via 155.1.23.3, 00:09:22, Serial0/1
                  [170/1666560] via 155.1.0.5, 00:09:22, Serial0/0.1
     54.0.0.0/24 is subnetted, 1 subnets
D EX    54.1.1.0 [170/1538560] via 155.1.23.3, 00:09:22, Serial0/1
                 [170/1538560] via 155.1.0.5, 00:09:23, Serial0/0.1
D EX 200.0.2.0/24 [170/1666560] via 155.1.23.3, 00:09:23, Serial0/1
                  [170/1666560] via 155.1.0.5, 00:09:23, Serial0/0.1
     31.0.0.0/16 is subnetted, 4 subnets
D EX    31.3.0.0 [170/1049600] via 155.1.0.5, 00:05:59, Serial0/0.1
D EX    31.2.0.0 [170/1049600] via 155.1.0.5, 00:05:59, Serial0/0.1
D EX    31.1.0.0 [170/1049600] via 155.1.0.5, 00:05:59, Serial0/0.1
D EX    31.0.0.0 [170/1049600] via 155.1.0.5, 00:06:01, Serial0/0.1
     150.1.0.0/24 is subnetted, 7 subnets
D       150.1.7.0 [90/1157120] via 155.1.0.5, 00:06:00, Serial0/0.1
D       150.1.6.0 [90/26240256] via 155.1.23.3, 00:06:00, Serial0/1
D       150.1.5.0 [90/640000] via 155.1.0.5, 00:09:21, Serial0/0.1
D       150.1.3.0 [90/640000] via 155.1.23.3, 00:09:21, Serial0/1
D       150.1.1.0 [90/26254592] via 155.1.23.3, 00:06:00, Serial0/1
D       150.1.9.0 [90/640512] via 155.1.23.3, 00:09:25, Serial0/1
     30.0.0.0/16 is subnetted, 4 subnets
D EX    30.2.0.0 [170/1049600] via 155.1.0.5, 00:09:21, Serial0/0.1
D EX    30.3.0.0 [170/1049600] via 155.1.0.5, 00:06:01, Serial0/0.1
D EX    30.0.0.0 [170/1049600] via 155.1.0.5, 00:09:21, Serial0/0.1
D EX    30.1.0.0 [170/1049600] via 155.1.0.5, 00:09:21, Serial0/0.1
```

Copyright © 2010 Internetwork Expert

As a filter for metrics the route-map match can on an absolute metric value, such as with the **`match metric 10`** command, or on a range of metrics. In this case metrics in the range of 500,000 – 750,000 are filtered out based on matching the value 625,000 plus or minus 125,000.

We can see from the below output the highlighted prefixes are no longer installed in the routing table via the same path once the filter is applied. Note that the route 150.1.2.0/24 is withdrawn completely as there is no other valid path to this destination than the one with a metric value matched by the range.

```
Rack1R3#show ip route eigrp
D EX 222.22.2.0/24 [170/537600] via 155.1.23.2, 00:02:51, Serial1/3
D EX 204.12.1.0/24 [170/1049600] via 155.1.0.5, 00:02:51, Serial1/0.1
     155.1.0.0/24 is subnetted, 13 subnets
D       155.1.146.0 [90/514560] via 155.1.13.1, 00:02:51, Serial1/2
D       155.1.8.0 [90/514816] via 155.1.0.5, 00:02:51, Serial1/0.1
D       155.1.9.0 [90/768] via 155.1.37.7, 00:02:51, FastEthernet0/0
D       155.1.7.0 [90/512] via 155.1.37.7, 00:02:51, FastEthernet0/0
D       155.1.5.0 [90/514560] via 155.1.0.5, 00:02:51, Serial1/0.1
D       155.1.58.0 [90/514560] via 155.1.0.5, 00:02:51, Serial1/0.1
D       155.1.79.0 [90/512] via 155.1.37.7, 00:02:51, FastEthernet0/0
D       155.1.67.0 [90/517120] via 155.1.13.1, 00:02:51, Serial1/2
D       155.1.108.0 [90/517120] via 155.1.0.5, 00:02:51, Serial1/0.1
D EX 220.20.3.0/24 [170/537600] via 155.1.23.2, 00:02:51, Serial1/3
D EX 200.0.0.0/24 [170/1154560] via 155.1.13.1, 00:02:51, Serial1/2
     54.0.0.0/24 is subnetted, 1 subnets
D EX    54.1.1.0 [170/1026560] via 155.1.13.1, 00:02:52, Serial1/2
D EX 200.0.2.0/24 [170/1154560] via 155.1.13.1, 00:02:52, Serial1/2
D EX 192.10.1.0/24 [170/537600] via 155.1.23.2, 00:02:52, Serial1/3
     31.0.0.0/16 is subnetted, 4 subnets
D EX    31.3.0.0 [170/1049600] via 155.1.0.5, 00:02:52, Serial1/0.1
D EX    31.2.0.0 [170/1049600] via 155.1.0.5, 00:02:52, Serial1/0.1
D EX    31.1.0.0 [170/1049600] via 155.1.0.5, 00:02:52, Serial1/0.1
D EX    31.0.0.0 [170/1049600] via 155.1.0.5, 00:02:52, Serial1/0.1
     150.1.0.0/24 is subnetted, 7 subnets
D       150.1.7.0 [90/645120] via 155.1.13.1, 00:02:53, Serial1/2
D       150.1.6.0 [90/642560] via 155.1.13.1, 00:02:53, Serial1/2
D       150.1.5.0 [90/640000] via 155.1.0.5, 00:02:53, Serial1/0.1
D       150.1.2.0 [90/640000] via 155.1.23.2, 00:02:52, Serial1/3
D       150.1.1.0 [90/640000] via 155.1.13.1, 00:02:52, Serial1/2
D       150.1.9.0 [90/128512] via 155.1.37.7, 00:02:53, FastEthernet0/0
D EX 205.90.31.0/24 [170/537600] via 155.1.23.2, 00:02:53, Serial1/3
     30.0.0.0/16 is subnetted, 4 subnets
D EX    30.2.0.0 [170/1049600] via 155.1.0.5, 00:02:53, Serial1/0.1
D EX    30.3.0.0 [170/1049600] via 155.1.0.5, 00:02:53, Serial1/0.1
D EX    30.0.0.0 [170/1049600] via 155.1.0.5, 00:02:53, Serial1/0.1
D EX    30.1.0.0 [170/1049600] via 155.1.0.5, 00:02:53, Serial1/0.1
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#router eigrp 100
Rack1R3(config-router)#distribute-list route-map FILTER_ON_METRIC_RANGE
IN
Rack1R3(config-router)#end

Rack1R3#show ip route eigrp
D EX 222.22.2.0/24 [170/1049600] via 155.1.0.5, 00:00:17, Serial1/0.1
D EX 204.12.1.0/24 [170/1049600] via 155.1.0.5, 00:03:38, Serial1/0.1
     155.1.0.0/24 is subnetted, 13 subnets
D       155.1.146.0 [90/1026560] via 155.1.0.5, 00:00:17, Serial1/0.1
D       155.1.8.0 [90/1026816] via 155.1.23.2, 00:00:17, Serial1/3
                  [90/1026816] via 155.1.13.1, 00:00:17, Serial1/2
D       155.1.9.0 [90/768] via 155.1.37.7, 00:03:38, FastEthernet0/0
D       155.1.7.0 [90/512] via 155.1.37.7, 00:03:38, FastEthernet0/0
D       155.1.5.0 [90/1026560] via 155.1.23.2, 00:00:17, Serial1/3
                  [90/1026560] via 155.1.13.1, 00:00:17, Serial1/2
D       155.1.58.0 [90/1026560] via 155.1.23.2, 00:00:17, Serial1/3
                   [90/1026560] via 155.1.13.1, 00:00:17, Serial1/2
D       155.1.79.0 [90/512] via 155.1.37.7, 00:03:38, FastEthernet0/0
D       155.1.67.0 [90/1029120] via 155.1.0.5, 00:00:18, Serial1/0.1
D       155.1.108.0 [90/1029120] via 155.1.23.2, 00:00:17, Serial1/3
                    [90/1029120] via 155.1.13.1, 00:00:18, Serial1/2
D EX 220.20.3.0/24 [170/1049600] via 155.1.0.5, 00:00:19, Serial1/0.1
D EX 200.0.0.0/24 [170/1154560] via 155.1.13.1, 00:00:19, Serial1/2
     54.0.0.0/24 is subnetted, 1 subnets
D EX    54.1.1.0 [170/1026560] via 155.1.13.1, 00:00:19, Serial1/2
D EX 200.0.2.0/24 [170/1154560] via 155.1.13.1, 00:00:19, Serial1/2
D EX 192.10.1.0/24 [170/1049600] via 155.1.0.5, 00:00:19, Serial1/0.1
     31.0.0.0/16 is subnetted, 4 subnets
D EX    31.3.0.0 [170/1049600] via 155.1.0.5, 00:03:41, Serial1/0.1
D EX    31.2.0.0 [170/1049600] via 155.1.0.5, 00:03:41, Serial1/0.1
D EX    31.1.0.0 [170/1049600] via 155.1.0.5, 00:03:41, Serial1/0.1
D EX    31.0.0.0 [170/1049600] via 155.1.0.5, 00:03:41, Serial1/0.1
     150.1.0.0/24 is subnetted, 6 subnets
D       150.1.7.0 [90/1157120] via 155.1.0.5, 00:00:20, Serial1/0.1
D       150.1.6.0 [90/25728256] via 155.1.37.7, 00:00:20, FastEthernet0/0
D       150.1.5.0 [90/1152000] via 155.1.23.2, 00:00:20, Serial1/3
                  [90/1152000] via 155.1.13.1, 00:00:20, Serial1/2
D       150.1.1.0 [90/25742592] via 155.1.37.7, 00:00:20, FastEthernet0/0
D       150.1.9.0 [90/128512] via 155.1.37.7, 00:03:41, FastEthernet0/0
D EX 205.90.31.0/24 [170/1049600] via 155.1.0.5, 00:00:20, Serial1/0.1
     30.0.0.0/16 is subnetted, 4 subnets
D EX    30.2.0.0 [170/1049600] via 155.1.0.5, 00:03:41, Serial1/0.1
D EX    30.3.0.0 [170/1049600] via 155.1.0.5, 00:03:41, Serial1/0.1
D EX    30.0.0.0 [170/1049600] via 155.1.0.5, 00:03:42, Serial1/0.1
D EX    30.1.0.0 [170/1049600] via 155.1.0.5, 00:03:42, Serial1/0.1
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

5.27 EIGRP Bandwidth Pacing

- Configure R2 and R3 so that EIGRP can not use more than 154Kbps of bandwidth on the point-to-point link between them, assuming that the link speed is 1544Kbps.

Configuration

```
R2:
interface Serial0/1
 bandwidth 1544
 ip bandwidth-percent eigrp 100 10

R3:
interface Serial1/3
 bandwidth 1544
 ip bandwidth-percent eigrp 100 10
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

5.28 EIGRP Default Metric

- Configure a static route on R2 for the prefix 222.22.2.2/32 that is reachable via BB2.
- Advertise this prefix into EIGRP as external routes using a default metric of 100Mbps, 100 microseconds of delay, maximum reliability, minimum load, and an MTU of 1500 bytes.

Configuration

```
R2:
ip route 222.22.2.2 255.255.255.255 192.10.1.254
!
router eigrp 100
 redistribute static
 default-metric 100000 10 255 1 1500
```

Verification

> **Note**
>
> When redistributing static into EIGRP or between EIGRP processes, metrics are automatically derived from the source prefix. For all other redistribution the metric must be manually set on the redistribute statement, under a route-map, or from the default metric.

```
Rack1R2#show ip eigrp topology 222.22.2.2/32
IP-EIGRP (AS 100): Topology entry for 222.22.2.2/32
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 2560
  Routing Descriptor Blocks:
  192.10.1.254, from Rstatic, Send flag is 0x0
      Composite metric is (2560/0), Route is External
      Vector metric:
        Minimum bandwidth is 100000 Kbit
        Total delay is 100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 0
      External data:
        Originating router is 150.1.2.2 (this system)
        AS number of route is 0
        External protocol is Static, external metric is 0
        Administrator tag is 0 (0x00000000)
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

5.29 EIGRP Neighbor Logging

- Configure SW3 so that it does not log EIGRP neighbor adjacency events.
- Additionally EIGRP warning logs should not be generated more often than every 20 seconds.

Configuration

```
SW3:
router eigrp 100
 no eigrp log-neighbor-changes
 eigrp log-neighbor-warnings 20
```

5.30 EIGRP Router-ID

- Modify the EIGRP Router-ID on SW2 so that external EIGRP routes generated by R2 are ignored.

Configuration

```
SW2:
router eigrp 100
 eigrp router-id 150.1.2.2
```

Verification

> **Note**
>
> EIGRP uses the router-id field in external routes as a loop prevention mechanism. The router who originates the external route inserts its EIGRP router-id into the update. If an update is received back in with a router-id in this field matching the local router-id, the update is dropped.
>
> In this case, SW2's router-id is 150.1.8.8, as seen in the topology table.

```
Rack1SW2#show ip eigrp topology | include ID
IP-EIGRP Topology Table for AS(100)/ID(150.1.8.8)

Rack1SW2#show ip route | include D EX
D EX    222.22.2.2/32 [170/514816] via 155.1.58.5, 00:12:47, Vlan58
D EX    222.22.2.0/24 [170/537856] via 155.1.58.5, 02:19:28, Vlan58
D EX 204.12.1.0/24 [170/537856] via 155.1.58.5, 01:41:23, Vlan58
D EX 220.20.3.0/24 [170/537856] via 155.1.58.5, 02:19:28, Vlan58
D EX 200.0.0.0/24 [170/1154816] via 155.1.58.5, 02:19:28, Vlan58
D EX    54.1.1.0 [170/1026816] via 155.1.58.5, 00:06:25, Vlan58
D EX 200.0.2.0/24 [170/1154816] via 155.1.58.5, 02:19:28, Vlan58
D EX 192.10.1.0/24 [170/514816] via 155.1.58.5, 00:12:45, Vlan58
D EX    31.3.0.0 [170/537856] via 155.1.58.5, 02:19:29, Vlan58
```

Copyright © 2010 Internetwork Expert

```
D EX    31.2.0.0 [170/537856] via 155.1.58.5, 02:19:29, Vlan58
D EX    31.1.0.0 [170/537856] via 155.1.58.5, 02:19:29, Vlan58
D EX    31.0.0.0 [170/537856] via 155.1.58.5, 02:19:29, Vlan58
D EX 205.90.31.0/24 [170/537856] via 155.1.58.5, 02:19:29, Vlan58
D EX    30.2.0.0 [170/537856] via 155.1.58.5, 02:19:29, Vlan58
D EX    30.3.0.0 [170/537856] via 155.1.58.5, 02:19:29, Vlan58
D EX    30.0.0.0 [170/537856] via 155.1.58.5, 02:19:29, Vlan58
D EX    30.1.0.0 [170/537856] via 155.1.58.5, 02:19:29, Vlan58
```

R2 is originating the external route 222.22.2.0/24, and it is tagged with R2's router-id of 150.1.2.2

```
Rack1SW2#show ip eigrp topology 222.22.2.0/24
IP-EIGRP (AS 100): Topology entry for 222.22.2.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is
537856
  Routing Descriptor Blocks:
  155.1.58.5 (Vlan58), from 155.1.58.5, Send flag is 0x0
      Composite metric is (537856/537600), Route is External
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 21010 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
      External data:
        Originating router is 150.1.2.2
        AS number of route is 0
        External protocol is RIP, external metric is 7
        Administrator tag is 0 (0x00000000)
```

If SW2 shares this same router-id, this prefix can no longer we installed, along with any other external routes originated by R2.

```
Rack1SW2#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW2(config)#router eigrp 100
Rack1SW2(config-router)#eigrp router-id 150.1.2.2
Rack1SW2(config-router)#end

Rack1SW2#show ip eigrp topology 222.22.2.0/24
% IP-EIGRP (AS 100): Route not in topology table
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show ip route | include D EX
D EX 204.12.1.0/24 [170/537856] via 155.1.58.5, 00:00:51, Vlan58
D EX 200.0.0.0/24 [170/1154816] via 155.1.58.5, 00:00:51, Vlan58
D EX    54.1.1.0 [170/1026816] via 155.1.58.5, 00:00:51, Vlan58
D EX 200.0.2.0/24 [170/1154816] via 155.1.58.5, 00:00:51, Vlan58
D EX    31.3.0.0 [170/537856] via 155.1.58.5, 00:00:51, Vlan58
D EX    31.2.0.0 [170/537856] via 155.1.58.5, 00:00:51, Vlan58
D EX    31.1.0.0 [170/537856] via 155.1.58.5, 00:00:51, Vlan58
D EX    31.0.0.0 [170/537856] via 155.1.58.5, 00:00:51, Vlan58
D EX    30.2.0.0 [170/537856] via 155.1.58.5, 00:00:52, Vlan58
D EX    30.3.0.0 [170/537856] via 155.1.58.5, 00:00:52, Vlan58
D EX    30.0.0.0 [170/537856] via 155.1.58.5, 00:00:52, Vlan58
D EX    30.1.0.0 [170/537856] via 155.1.58.5, 00:00:52, Vlan58
```

5.31 EIGRP Maximum Hops

- Configure all devices in EIGRP AS 100 so that routes with a hop count of greater than 10 are considered invalid.

Configuration

```
R1 - R6, SW1 - SW4:
router eigrp 100
 metric maximum-hops 10
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

OSPF Solutions

Note

Load the *Initial OSPF* initial configurations prior to starting.

6.1 OSPF over Broadcast Media

- Enable OSPF on all devices using Process-ID 1.
- On R1 configure OSPF area 1 on the link to VLAN 146.
- On R2 configure OSPF area 51 on the link to VLAN 22.
- On R3 configure OSPF area 2 on the link to SW1.
- On R4 configure OSPF area 1 on the link to VLAN 146.
- On R5 configure OSPF area 3 on the links to VLANs 5 and 58.
- On R6 configure OSPF area 1 on the link to VLAN 146, and area 2 on the link to VLAN 67.
- Do not use the **network** statement under the OSPF process on R1, R2, R3, R4, R5, or R6.
- On SW1 configure OSPF area 2 on all interfaces with an IP address assigned using one single **network** statement.
- On SW2 configure OSPF area 3 on interfaces with IP addresses in the range of 155.X.0.0 – 155.X.127.255 using one single **network** statement.
- On SW3 configure OSPF area 2 on interfaces with IP addresses in the range of 155.X.9.0 – 155.X.9.255 and 155.X.79.0 – 155.X.79.255 using two **network** statements.
- On SW4 configure OSPF area 3 only on interfaces with the exact IP addresses 155.X.10.10 and 155.X.108.10.
- Note any reachability problems throughout the OSPF topology.

Configuration

```
R1:
interface FastEthernet0/0
 ip ospf 1 area 1

R2:
interface FastEthernet0/0
 ip ospf 1 area 51

R3:
interface FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

```
 ip ospf 1 area 2
!
router ospf 1
 router-id 150.1.3.3

R4:
interface FastEthernet0/1
 ip ospf 1 area 1

R5:
interface FastEthernet0/0
 ip ospf 1 area 3
!
interface FastEthernet0/1
 ip ospf 1 area 3

R6:
interface FastEthernet0/0.67
 ip ospf 1 area 2
!
interface FastEthernet0/0.146
 ip ospf 1 area 1

SW1:
ip routing
!
router ospf 1
 network 0.0.0.0 255.255.255.255 area 2

SW2:
ip routing
!
router ospf 1
 network 155.1.0.0 0.0.127.255 area 3

SW3:
ip routing
!
router ospf 1
 router-id 150.1.9.9
 network 155.1.9.0 0.0.0.255 area 2
 network 155.1.79.0 0.0.0.255 area 2

SW4:
ip routing
!
router ospf 1
 network 155.1.10.10 0.0.0.0 area 3
 network 155.1.108.10 0.0.0.0 area 3
```

Copyright © 2010 Internetwork Expert

Verification

Note

As of IOS 12.4 there are two ways to enable the OSPF process on an interface. These include the legacy **network** statement under the OSPF process, and the interface level command **ip ospf [*process-id*] area [*area-id*]**. Both accomplish the same thing with one minor exception. If an interface is IP unnumbered, and there is a **network** statement that matches the IP address of the primary interface, both the primary interface and the unnumbered interface will have OSPF enabled on them in the designated area.

Despite common confusion, the **network** statement in OSPF, just like the network statement under the EIGRP process, is not used to *originate* a network advertisement; instead it simply enables the OSPF process on the interface. If multiple network statements overlap the same interface, the most specific match based on the wildcard mask wins.

In this particular example SW1 enables the OSPF process on all interfaces with the **network 0.0.0.0 255.255.255.255 area 2** command. This means that all interfaces with an IP address assigned will be placed into area 2. This does *not* mean, however, that the network 0.0.0.0/0 itself will be advertised.

Likewise on SW2 the **network 155.1.0.0 0.0.127.255 area 3** command means that any address with the first 17 contiguous bits matching the address 155.1.0.0 will be placed into area 3. This does not mean that the network 155.1.0.0/17 will be originated.

The most specific match is seen on SW4, with the **network 155.1.10.10 0.0.0.0 area 3** command. This means that only the interface with the exact IP address 155.1.10.10 will be placed into area 3. If the interfaces 155.1.10.9 or 155.1.10.11, or any other variation, exist on the device, they will not be placed into area 3.

Copyright © 2010 Internetwork Expert

Once the **network** statement or the ip ospf statement are configured this can be quickly verified with the **show ip ospf interface brief** command. Note that in the below output there is no functional difference seen between R1 – R6 who used the interface level command to enable OSPF vs. SW1 – SW4 who used the network statement.

```
Rack1R1#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Fa0/0        1     1               155.1.146.1/24     1     DROTH 2/2

Rack1R2#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Fa0/0        1     51              192.10.1.2/24      1     BDR   1/1

Rack1R3#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Fa0/0        1     2               155.1.37.3/24      1     DR    1/1

Rack1R4#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Fa0/1        1     1               155.1.146.4/24     1     DR    2/2

Rack1R5#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Fa0/1        1     3               155.1.5.5/24       1     DR    0/0
Fa0/0        1     3               155.1.58.5/24      1     BDR   1/1

Rack1R6#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Fa0/0.146    1     1               155.1.146.6/24     1     BDR   2/2
Fa0/0.67     1     2               155.1.67.6/24      1     DR    1/1

Rack1SW1#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Lo0          1     2               150.1.7.7/24       1     LOOP  0/0
Fa0/3        1     2               155.1.37.7/24      1     BDR   1/1
Vl79         1     2               155.1.79.7/24      1     BDR   1/1
Vl67         1     2               155.1.67.7/24      1     BDR   1/1
Vl7          1     2               155.1.7.7/24       1     DR    0/0

Rack1SW2#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Po1          1     3               155.1.108.8/24     1     BDR   1/1
Vl58         1     3               155.1.58.8/24      1     DR    1/1
Vl8          1     3               155.1.8.8/24       1     DR    0/0

Rack1SW3#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Vl79         1     2               155.1.79.9/24      1     DR    1/1
Vl9          1     2               155.1.9.9/24       1     DR    0/0

Rack1SW4#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Po1          1     3               155.1.108.10/24    1     DR    1/1
Vl10         1     3               155.1.10.10/24     1     DR    0/0
```

Copyright © 2010 Internetwork Expert

Once it is verified that the interfaces are configured in the correct areas, the next verification is to check the adjacency state of the OSPF neighbors with the **show ip ospf neighbor** command.

At this point it is important to note the attributes that must match as well as those that must be unique in order for an adjacency to be established. The common attributes that must match are the area, timers, authentication, stub flags, MTU, and compatible network types. The attributes that must be unique are the interface IP addresses and the router-id's. The router-id is chosen first based on the process level **router-id** command, second based on the highest active Loopback IP interface, and lastly on the highest active non-Loopback interface IP address.

In this design R3 and SW3 are configured with the duplicate Loopback IP address 222.255.255.255/32. Since this is the highest Loopback IP address it is chosen as the router-id. Since these devices are in the same area, and share the same router-id, both adjacency problems and SPF problems become apparent. This can be seen by log messages on R3 and SW3 along with SPF and routing table calculation problems on SW1.

```
Rack1SW1#debug ip routing
IP routing debugging is on
RT: add 155.1.108.0/24 via 155.1.37.3, ospf metric [110/784]
RT: SET_LAST_RDB for 155.1.146.0/24
  NEW rdb: via 155.1.37.3

RT: del 155.1.108.0/24 via 155.1.37.3, ospf metric [110/784]
RT: delete subnet route to 155.1.108.0/24
```

Since the LSA origination is based on the router-id, SW1 thinks it has two interfaces connecting to the same device. The result of this is that SW1 constantly recalculates SPF forcing it to add and remove prefixes from the routing table because it sees two neighbors with the same router-id advertising different information.

From R3 and SW3's perspective they see each other's Router LSA's (LSA 1) with the same router-id as the local device, and cannot compute SPF. This can be clearly seen from their log messages to the console.

```
Rack1R3#
%OSPF-4-DUP_RTRID_AREA: Detected router with duplicate router ID
222.255.255.255 in area 2

Rack1SW3#
%OSPF-4-DUP_RTRID_AREA: Detected router with duplicate router ID
222.255.255.255 in area 2
```

Copyright © 2010 Internetwork Expert

To fix this design problem the router-id needs to be changed to something unique. This can be accomplished by changing or removing IP addresses on the devices, or by issuing the router-id command under the OSPF process. In this solution the router-id command is used to match the unique Loopback 0 interfaces of R3 and SW1 respectively.

```
Rack1SW3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW3(config)#router ospf 1
Rack1SW3(config-router)#router-id 150.1.9.9
Reload or use "clear ip ospf process" command, for this to take effect
Rack1SW3(config-router)#end
Rack1SW3#
%SYS-5-CONFIG_I: Configured from console by console
Rack1SW3#clear ip ospf process
Reset ALL OSPF processes? [no]: yes
Rack1SW3#
```

Once modified on both R3 and SW3, SPF calculation inside area 2 succeeds, and proper routing information can be installed.

```
Rack1R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.6.6         1   FULL/BDR        00:00:37    155.1.146.6     FastEthernet0/0
223.255.255.255   1   FULL/DR         00:00:36    155.1.146.4     FastEthernet0/0

Rack1R2#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
192.10.1.254      1   FULL/DR         00:00:36    192.10.1.254    FastEthernet0/0

Rack1R3#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.7.7         1   FULL/BDR        00:00:32    155.1.37.7      FastEthernet0/0
Rack1R4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.1.1         1   FULL/DROTHER    00:00:37    155.1.146.1     FastEthernet0/1
150.1.6.6         1   FULL/BDR        00:00:37    155.1.146.6     FastEthernet0/1

Rack1R5#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.8.8         1   FULL/DR         00:00:32    155.1.58.8      FastEthernet0/0

Rack1R6#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time Address       Interface
150.1.1.1         1   FULL/DROTHER    00:00:39  155.1.146.1   FastEthernet0/0.146
223.255.255.255   1   FULL/DR         00:00:39  155.1.146.4   FastEthernet0/0.146
150.1.7.7         1   FULL/BDR        00:00:36  155.1.67.7    FastEthernet0/0.67
```

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.3.3         1   FULL/DR         00:00:32    155.1.37.3      FastEthernet0/3
150.1.9.9         1   FULL/DR         00:00:38    155.1.79.9      Vlan79
150.1.6.6         1   FULL/DR         00:00:32    155.1.67.6      Vlan67

Rack1SW2#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
223.255.255.255   1   FULL/DR         00:00:30    155.1.108.10    Port-channel1
150.1.5.5         1   FULL/BDR        00:00:38    155.1.58.5      Vlan58

Rack1SW3#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.7.7         1   FULL/BDR        00:00:37    155.1.79.7      Vlan79

Rack1SW4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.8.8         1   FULL/BDR        00:00:38    155.1.108.8     Port-channel1
```

Once adjacency has been established a fundamental underlying design issue still exists in the network. At this point only areas 1, 2, 3, and 51 are configured. The backbone area 0 is not configured on any links. This implies that the devices can route within their own area (Intra-Area), but not between areas (Inter-Area). This is due to the fact that the Area Border Router (ABR) who connects one area to area zero is responsible for generating the Network Summary LSA (LSA 3) that describes the Inter-Area routes. The result of this can be seen by viewing the routing table.

R1 and R4 do not see any routes installed because the only link advertised in area 1 is their connected VLAN 146 interface.

```
Rack1R1#show ip route ospf

Rack1R4#show ip route ospf
```

R2 sees area 51 routes from its adjacent neighbor, BB2.

```
Rack1R2#show ip route ospf
     51.0.0.0/32 is subnetted, 1 subnets
O E2    51.51.51.51 [110/20] via 192.10.1.254, 00:08:46,
```

Copyright © 2010 Internetwork Expert

R3, R6, SW1, and SW3 know about the routes in area 2, but not the routes in area 1.

```
Rack1R3#show ip route ospf
     155.1.0.0/24 is subnetted, 8 subnets
O       155.1.9.0 [110/3] via 155.1.37.7, 00:01:09, FastEthernet0/0
O       155.1.7.0 [110/2] via 155.1.37.7, 00:01:09, FastEthernet0/0
O       155.1.79.0 [110/2] via 155.1.37.7, 00:01:09, FastEthernet0/0
O       155.1.67.0 [110/2] via 155.1.37.7, 00:01:09, FastEthernet0/0
     150.1.0.0/16 is variably subnetted, 2 subnets, 2 masks
O       150.1.7.7/32 [110/2] via 155.1.37.7, 00:01:10, FastEthernet0/0

Rack1R6#show ip route ospf
     155.1.0.0/24 is subnetted, 6 subnets
O       155.1.9.0 [110/3] via 155.1.67.7, 00:01:28, FastEthernet0/0.67
O       155.1.7.0 [110/2] via 155.1.67.7, 00:01:28, FastEthernet0/0.67
O       155.1.37.0 [110/2] via 155.1.67.7, 00:01:28, FastEthernet0/0.67
O       155.1.79.0 [110/2] via 155.1.67.7, 00:01:28, FastEthernet0/0.67
     150.1.0.0/16 is variably subnetted, 2 subnets, 2 masks
O       150.1.7.7/32 [110/2] via 155.1.67.7, 00:01:28,

Rack1SW1#show ip route ospf
     155.1.0.0/24 is subnetted, 5 subnets
O       155.1.9.0 [110/2] via 155.1.79.9, 00:01:35, Vlan79

Rack1SW3#show ip route ospf
     155.1.0.0/24 is subnetted, 5 subnets
O       155.1.7.0 [110/2] via 155.1.79.7, 00:01:48, Vlan79
O       155.1.37.0 [110/2] via 155.1.79.7, 00:01:48, Vlan79
O       155.1.67.0 [110/2] via 155.1.79.7, 00:01:48, Vlan79
     150.1.0.0/16 is variably subnetted, 2 subnets, 2 masks
O       150.1.7.7/32 [110/2] via 155.1.79.7, 00:01:49, Vlan79
```

R5, SW2, and SW4 know about the routes in area 3.

```
Rack1R5#show ip route ospf
     155.1.0.0/24 is subnetted, 7 subnets
O       155.1.10.0 [110/3] via 155.1.58.8, 00:03:27, FastEthernet0/0
O       155.1.8.0 [110/2] via 155.1.58.8, 00:03:27, FastEthernet0/0
O       155.1.108.0 [110/2] via 155.1.58.8, 00:03:27, FastEthernet0/0

Rack1SW2#show ip route ospf
     155.1.0.0/24 is subnetted, 5 subnets
O       155.1.10.0 [110/2] via 155.1.108.10, 00:03:41, Port-channel1
O       155.1.5.0 [110/2] via 155.1.58.5, 00:03:41, Vlan58

Rack1SW4#show ip route ospf
     155.1.0.0/24 is subnetted, 5 subnets
O       155.1.8.0 [110/2] via 155.1.108.8, 00:03:53, Port-channel1
O       155.1.5.0 [110/3] via 155.1.108.8, 00:03:53, Port-channel1
O       155.1.58.0 [110/2] via 155.1.108.8, 00:03:53, Port-channel1
```

Copyright © 2010 Internetwork Expert

Intra-area routing is successful but inter-area routing fails.

```
Rack1SW3#ping 155.1.67.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.67.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1SW3#debug ip packet
IP packet debugging is on
Rack1SW3#ping 155.1.146.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.6, timeout is 2 seconds:

IP: s=155.1.9.9 (local), d=155.1.146.6, len 100, unroutable.
IP: s=155.1.9.9 (local), d=155.1.146.6, len 100, unroutable.
IP: s=155.1.9.9 (local), d=155.1.146.6, len 100, unroutable.
IP: s=155.1.9.9 (local), d=155.1.146.6, len 100, unroutable.
IP: s=155.1.9.9 (local), d=155.1.146.6, len 100, unroutable.
Success rate is 0 percent (0/5)
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

6.2 OSPF over Non-Broadcast Media

- Configure OSPF area 0 on R1, R2, R3, R4, and R5's connections to the Frame Relay network.
- Without modifying the OSPF network type on any of these devices ensure full reachability is obtained throughout the OSPF domain.

Configuration

```
R1:
interface Serial0/0
 ip ospf 1 area 0
 frame-relay map ip 155.1.0.2 105
 frame-relay map ip 155.1.0.3 105
 frame-relay map ip 155.1.0.4 105

R2:
interface Serial0/0
 ip ospf 1 area 0
 frame-relay map ip 155.1.0.1 205
 frame-relay map ip 155.1.0.3 205
 frame-relay map ip 155.1.0.4 205

R3:
interface Serial1/0
 ip ospf 1 area 0
 frame-relay map ip 155.1.0.1 305
 frame-relay map ip 155.1.0.2 305
 frame-relay map ip 155.1.0.4 305

R4:
interface Serial0/0/0
 ip ospf 1 area 0
 frame-relay map ip 155.1.0.1 405
 frame-relay map ip 155.1.0.2 405
 frame-relay map ip 155.1.0.3 405

R5:
interface Serial0/0/0
 ip ospf 1 area 0
!
router ospf 1
 neighbor 155.1.0.3
 neighbor 155.1.0.1
 neighbor 155.1.0.2
 neighbor 155.1.0.4
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> In this solution OSPF is enabled on the interfaces with the interface level **ip ospf** command. The **network** statement could have also been used under the OSPF process. The first goal of the section is simply to ensure that the **show ip ospf interface** output indicates that the Frame Relay interfaces are running OSPF in area 0.

```
Rack1R1#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Se0/0        1     0               155.1.0.1/24       64    BDR   1/1
Fa0/0        1     1               155.1.146.1/24     1     DROTH 2/2

Rack1R2#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Se0/0        1     0               155.1.0.2/24       64    BDR   1/1
Fa0/0        1     51              192.10.1.2/24      1     BDR   1/1

Rack1R3#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Se1/0        1     0               155.1.0.3/24       781   BDR   1/1
Fa0/0        1     2               155.1.37.3/24      1     DR    1/1

Rack1R4#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Se0/0/0      1     0               155.1.0.4/24       64    BDR   1/1
Fa0/1        1     1               155.1.146.4/24     1     DR    2/2

Rack1R5#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Se0/0/0      1     0               155.1.0.5/24       64    DR    4/4
Fa0/1        1     3               155.1.5.5/24       1     DR    0/0
Fa0/0        1     3               155.1.58.5/24      1     BDR   1/1
```

Copyright © 2010 Internetwork Expert

The next consideration for establishing reachability in this design is the difference between the default OSPF network types on NBMA Frame Relay interfaces versus broadcast Ethernet interfaces. As seen in the **show ip ospf interface** output the default network type for a multipoint Frame Relay interface is *NON_BROADCAST*.

```
Rack1R5#show ip ospf interface Serial0/0/0
Serial0/0 is up, line protocol is up
  Internet Address 155.1.0.5/24, Area 0
  Process ID 1, Router ID 150.1.5.5, Network Type NON_BROADCAST, Cost: 64
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 150.1.5.5, Interface address 155.1.0.5
  Backup Designated router (ID) 223.255.255.255, Interface address 155.1.0.4
  Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit 5
    oob-resync timeout 120
    Hello due in 00:00:16
  Supports Link-local Signaling (LLS)
  Index 1/2, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 4
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 4, Adjacent neighbor count is 4
    Adjacent with neighbor 150.1.3.3
    Adjacent with neighbor 150.1.1.1
    Adjacent with neighbor 150.1.2.2
    Adjacent with neighbor 223.255.255.255  (Backup Designated Router)

Rack1R1#show ip ospf interface Serial0/0
Serial0/0 is up, line protocol is up
  Internet Address 155.1.0.1/24, Area 0
  Process ID 1, Router ID 150.1.1.1, Network Type NON_BROADCAST, Cost: 64
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State BDR, Priority 1
  Designated Router (ID) 150.1.5.5, Interface address 155.1.0.5
  Backup Designated router (ID) 150.1.1.1, Interface address 155.1.0.1
  Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit 5
    oob-resync timeout 120
    Hello due in 00:00:12
  Supports Link-local Signaling (LLS)
  Index 1/2, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.5.5  (Designated Router)
  Suppress hello for 0 neighbor(s)
```

Copyright © 2010 Internetwork Expert

The Non-Broadcast network type means that there will be a DR/BDR election, and that hellos are exchanged as unicast. In order to unicast OSPF hellos the **neighbor** statement needs to be configured under the OSPF process of the DR. Once the DROTHERs and/or BDR hear the unicast hellos from the DR, they will automatically respond back with their own unicast hellos. This implies that the **neighbor** statement can be configured everywhere, but is only required on the DR. Once R5 is configured with the neighbor statement the show ip ospf neighbor output should be checked to verify adjacency.

```
Rack1R5#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.3.3         1   FULL/DROTHER    00:01:34    155.1.0.3       Serial0/0/0
150.1.1.1         1   FULL/DROTHER    00:01:48    155.1.0.1       Serial0/0/0
150.1.2.2         1   FULL/DROTHER    00:01:59    155.1.0.2       Serial0/0/0
223.255.255.255   1   FULL/BDR        00:01:45    155.1.0.4       Serial0/0/0
150.1.8.8         1   FULL/DR         00:00:33    155.1.58.8      FastEthernet0/0
```

Once adjacency is established in area 0, Inter-Area routing advertisements can be propagated throughout the entire topology. This is due to the fact that R1, R2, R3, R4, and R5 are now ABRs, and can originate the Network Summary LSA (LSA 3) describing Inter-Area routes to the other neighbors in their attached areas.

From the **show ip ospf database** output on R5 the *Summary Net link States (Area 0)* shows ABRs that are advertising information from other areas into area 0. For example R2 is advertising the link 192.10.1.0/24 from area 51 into area 0, and can be seen as the last entry in this category.

```
Rack1R5#show ip ospf database

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
150.1.1.1       150.1.1.1       1489        0x80000003 0x007B03 1
150.1.2.2       150.1.2.2       1670        0x80000003 0x006712 1
150.1.3.3       150.1.3.3       1452        0x80000003 0x00ECB7 1
150.1.5.5       150.1.5.5       1138        0x80000004 0x002940 1
223.255.255.255 223.255.255.255 982         0x80000006 0x008962 1

                Net Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.0.5       150.1.5.5       981         0x80000002 0x001283
```

Copyright © 2010 Internetwork Expert

```
                Summary Net Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum
150.1.7.7       150.1.3.3       1704        0x80000002 0x00DB1B
155.1.5.0       150.1.5.5       1709        0x80000002 0x00D225
155.1.7.0       150.1.3.3       1704        0x80000002 0x00E018
155.1.8.0       150.1.5.5       1709        0x80000002 0x00BB38
155.1.9.0       150.1.3.3       1706        0x80000002 0x00D421
155.1.10.0      150.1.5.5       1710        0x80000002 0x00AF41
155.1.37.0      150.1.3.3       1706        0x80000002 0x008B50
155.1.58.0      150.1.5.5       1710        0x80000002 0x008939
155.1.67.0      150.1.3.3       1706        0x80000002 0x004A72
155.1.79.0      150.1.3.3       1706        0x80000002 0x00C5EA
155.1.108.0     150.1.5.5       1710        0x80000002 0x006B24
155.1.146.0     150.1.1.1       1739        0x80000002 0x00F180
155.1.146.0     223.255.255.255 983         0x80000003 0x0071B9
192.10.1.0      150.1.2.2       1672        0x80000002 0x00D6FC

                Summary ASB Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum
192.10.1.254    150.1.2.2       1672        0x80000002 0x00D201

                Router Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum Link count
150.1.5.5       150.1.5.5       1710        0x8000002B 0x004017 2
150.1.8.8       150.1.8.8       219         0x80000031 0x007D8E 3
223.255.255.255 223.255.255.255 443         0x8000002C 0x00A0C9 2

                Net Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.58.8      150.1.8.8       219         0x80000029 0x009AA7
155.1.108.10    223.255.255.255 443         0x80000029 0x00ECAA
```

Since R5 is the ABR connecting area 0 to area 3, R5 re-originates all Inter-Area routes coming from other neighbors in area 0 as LSA type 3 routes into area 3. This is seen under the *Summary Net Link States (Area 3)* field.

```
                Summary Net Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
150.1.7.7       150.1.5.5       1135        0x80000001 0x00466D
155.1.0.0       150.1.5.5       1712        0x80000002 0x00823B
155.1.7.0       150.1.5.5       1135        0x80000001 0x004B6A
155.1.9.0       150.1.5.5       1135        0x80000001 0x003F73
155.1.37.0      150.1.5.5       1135        0x80000001 0x00F5A2
155.1.67.0      150.1.5.5       1135        0x80000001 0x00B4C4
155.1.79.0      150.1.5.5       1135        0x80000001 0x00303D
155.1.146.0     150.1.5.5       1451        0x80000002 0x0040E9
192.10.1.0      150.1.5.5       1135        0x80000001 0x00345A
```

Copyright © 2010 Internetwork Expert

```
                Summary ASB Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
192.10.1.254    150.1.5.5       1136        0x80000001 0x00305E

                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
51.51.51.51     192.10.1.254    1706        0x80000052 0x00F9BD 0
```

The result of the Inter-Area routing advertisements can be seen in the routing table via the **`show ip route ospf`** output. These new routes are denoted as *O IA* for OSPF Inter-Area. The next important design issue in this example is how the DR processes routing advertisements on the NBMA segment.

First, if we view the **`show ip ospf neighbor`** output on R1 – R5 we can see that the spokes of the Frame Relay network only form adjacency with R5, the DR. There is no direct adjacency between R1 and R2, because there is no direct layer 2 Frame Relay PVC between them. If you reached this step and R5 is *not* the DR for the segment, an additional problem will be seen that is covered in the next section regarding the DR/BDR election process. However, for the sake of this example let's assume that R5 is the DR.

```
Rack1R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         1   FULL/DR         00:01:37    155.1.0.5       Serial0/0
150.1.6.6         1   FULL/BDR        00:00:29    155.1.146.6     FastEthernet0/0
223.255.255.255   1   FULL/DR         00:00:39    155.1.146.4     FastEthernet0/0

Rack1R2#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         1   FULL/DR         00:01:30    155.1.0.5       Serial0/0
192.10.1.254      1   FULL/DR         00:00:33    192.10.1.254    FastEthernet0/0

Rack1R3#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         1   FULL/DR         00:01:54    155.1.0.5       Serial1/0
150.1.7.7         1   FULL/BDR        00:00:30    155.1.37.7      FastEthernet0/0

Rack1R4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         1   FULL/DR         00:01:50    155.1.0.5       Serial0/0/0
150.1.1.1         1   FULL/DROTHER    00:00:32    155.1.146.1     FastEthernet0/1
150.1.6.6         1   FULL/BDR        00:00:32    155.1.146.6     FastEthernet0/1

Rack1R5#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.3.3         1   FULL/DROTHER    00:01:34    155.1.0.3       Serial0/0/0
150.1.1.1         1   FULL/DROTHER    00:01:48    155.1.0.1       Serial0/0/0
150.1.2.2         1   FULL/DROTHER    00:01:59    155.1.0.2       Serial0/0/0
223.255.255.255   1   FULL/BDR        00:01:45    155.1.0.4       Serial0/0/0
150.1.8.8         1   FULL/DR         00:00:33    155.1.58.8      FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

The DR state on the segment means that R5 is responsible for replicating LSA information between its adjacent neighbors. For example R3 sends R5 the LSA type-3 route 155.1.9.0/24 describing SW3's link to VLAN 9. When R2 learns this on the Frame Relay network it comes from the DR, R5, but the next-hop value of the route is 155.1.0.3, the originator of the LSA type-3 advertisement. However, when R2 learns the route to 155.1.10.0/24, the LSA type-3 originator is R5, so the next-hop value towards this route is also R5. This is seen in the below **`show ip route ospf`** output.

```
Rack1R2#show ip route ospf
     51.0.0.0/32 is subnetted, 1 subnets
O E2    51.51.51.51 [110/20] via 192.10.1.254, 00:04:23, FastEthernet0/0
     155.1.0.0/24 is subnetted, 13 subnets
O IA    155.1.146.0 [110/65] via 155.1.0.4, 00:04:23, Serial0/0
                    [110/65] via 155.1.0.1, 00:04:23, Serial0/0
O IA    155.1.10.0 [110/67] via 155.1.0.5, 00:04:23, Serial0/0
O IA    155.1.8.0 [110/66] via 155.1.0.5, 00:04:23, Serial0/0
O IA    155.1.9.0 [110/67] via 155.1.0.3, 00:04:23, Serial0/0
O IA    155.1.7.0 [110/66] via 155.1.0.3, 00:04:23, Serial0/0
O IA    155.1.5.0 [110/65] via 155.1.0.5, 00:04:23, Serial0/0
O IA    155.1.58.0 [110/65] via 155.1.0.5, 00:04:23, Serial0/0
O IA    155.1.37.0 [110/65] via 155.1.0.3, 00:04:23, Serial0/0
O IA    155.1.79.0 [110/66] via 155.1.0.3, 00:04:23, Serial0/0
O IA    155.1.67.0 [110/66] via 155.1.0.3, 00:04:23, Serial0/0
O IA    155.1.108.0 [110/66] via 155.1.0.5, 00:04:23, Serial0/0
     150.1.0.0/16 is variably subnetted, 2 subnets, 2 masks
O IA    150.1.7.7/32 [110/66] via 155.1.0.3, 00:04:24, Serial0/0
```

This is due to the fact that the DR passes the routes along, but it does not modify any of the routing lookup attributes. The result of this behavior is seen when route recursion is performed to the final destination. From the *IP Routing* section of this publication recall that when a routing lookup is done the router also needs to perform layer 3 to layer 2 mapping for the next-hop value on the link. Let's look at what happens when R2 tries to send traffic to 155.1.10.10.

First R2 finds the longest match to 155.1.10.10, which is 155.1.10.0/24 via the next-hop 155.1.0.5

```
Rack1R2#show ip route 155.1.10.10
Routing entry for 155.1.10.0/24
  Known via "ospf 1", distance 110, metric 67, type inter area
  Last update from 155.1.0.5 on Serial0/0, 00:05:20 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.5.5, 00:05:20 ago, via Serial0/0
      Route metric is 67, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

R2 then needs to do another recursive lookup to find out how to forward towards 155.1.0.5. This is seen via the match 155.1.0.0/24 out Serial0/0.

```
Rack1R2#show ip route 155.1.0.5
Routing entry for 155.1.0.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Routing Descriptor Blocks:
  * directly connected, via Serial0/0
      Route metric is 0, traffic share count is 1
```

Since Serial0/0 is a multipoint Frame Relay interface, R2 must now find out which Frame Relay DLCI is associated with the next-hop 155.1.0.5. From the Frame Relay mapping table R2 sees that DLCI 205 is used.

```
Rack1R2#show frame-relay map | include 155.1.0.5
Serial0/0 (up): ip 155.1.0.5 dlci 205(0xCD,0x30D0), static,
```

In this case the layer 3 routing lookup is successful, and the layer 2 resolution is successful. The result is a successful ICMP PING to the destination.

```
Rack1R2#ping 155.1.10.10

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.10.10, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/61 ms
```

Now let's look at what R2 does when it tries to send traffic towards 155.1.9.9. The routing lookup for this destination says the longest match is 155.1.9.0/24 via 155.1.0.3. Even though the route came from R5, the next-hop is still the originator, R3, because the DR does not update this.

```
Rack1R2#show ip route 155.1.9.9
Routing entry for 155.1.9.0/24
  Known via "ospf 1", distance 110, metric 67, type inter area
  Last update from 155.1.0.3 on Serial0/0, 00:05:42 ago
  Routing Descriptor Blocks:
  * 155.1.0.3, from 150.1.3.3, 00:05:42 ago, via Serial0/0
      Route metric is 67, traffic share count is 1
```

R2's recursive lookup for 155.1.0.3 says that it is directly connected via the multipoint Frame Relay interface, Serial0/0.

```
Rack1R2#show ip route 155.1.0.3
Routing entry for 155.1.0.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Routing Descriptor Blocks:
  * directly connected, via Serial0/0
      Route metric is 0, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

R2 now tries to find the Frame Relay PVC that is used to get towards 155.1.0.3. Since there is no mapping configured, and since R2 and R3 cannot use Inverse-ARP to resolve each other's non-directly connected interfaces, this lookup fails.

```
Rack1R2#show frame-relay map | include 155.1.0.3
```

The result of this layer 2 failure can be seen from both the **debug ip packet** and **debug frame-relay packet** output.

```
Rack1R2#debug ip packet
IP packet debugging is on

Rack1R2#debug frame-relay packet
Frame Relay packet debugging is on

Rack1R2#ping 155.1.9.9

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.9.9, timeout is 2 seconds:

IP: tableid=0, s=155.1.0.2 (local), d=155.1.9.9 (Serial0/0), routed via RIB
IP: s=155.1.0.2 (local), d=155.1.9.9 (Serial0/0), len 100, sending
Serial0/0:Encaps failed--no map entry link 7(IP)
IP: s=155.1.0.2 (local), d=155.1.9.9 (Serial0/0), len 100, encapsulation failed
<output omitted>
Success rate is 0 percent (0/5)
```

The first portion of this debug output is from the **debug ip packet**, and says that the destination 155.1.9.9 is out Serial0/0 and is *routed via RIB*. This means that the layer 3 routing lookup was successful. A failure in the routing lookup was seen in the previous task, and would show the *unroutable* output in this debug.

The next portion is the routing process sending the packet to the layer 2 process. The **debug frame-relay packet** output says that *encapsulation failed* and that there is *no map entry link*. **Debug ip packet** says that *encapsulation failed*. These two outputs mean that the router does not know which layer 2 PVC to use when building the frame towards the next-hop in the network. Although this technically is not an issue directly related to OSPF, it is part of the overall network design that must be solved.

There are basically two ways to solve this design problem, either change the next-hop to something the router *does* know how to resolve, or allow the router to properly resolve the current next-hop. In this solution, the current next-hop is resolved by adding additional **frame-relay map** statements. Changing the next-hop value is explored in the next few examples with the point-to-multipoint OSPF network type.

Copyright © 2010 Internetwork Expert

By adding additional mappings for R2 to reach R3 via R5, and for R3 to reach R2 via R5, layer 2 encapsulation is now successful, and end-to-end transport is achieved.

```
Rack1R2#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R2(config)#interface Serial0/0
Rack1R2(config-if)#frame-relay map ip 155.1.0.3 205
Rack1R2(config-if)#end
Rack1R2#

Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#interface Serial1/0
Rack1R3(config-if)#frame-relay map ip 155.1.0.2 305
Rack1R3(config-if)#end
Rack1R3#

Rack1R2#ping 155.1.9.9

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.9.9, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/117/125 ms
```

Based on this it is required that all spokes of the Frame Relay network, R1, R2, R3, and R4, must have **`frame-relay map`** statements for all other spokes via the DLCI to R5. Only once these additional statements are configured will there be end-to-end transport throughout the entire OSPF network.

Copyright © 2010 Internetwork Expert

6.3 OSPF DR/BDR Election Manipulation

- Configure the network so that R6 is elected the OSPF Designated Router for VLANs 67 and 146.
- If R6 goes down R1 should take over the DR status for VLAN 146.
- When R6 comes back up it should become the BDR.
- Modify the DR/BDR election on the Frame Relay network to ensure that if R5's connection to the Frame Relay network goes down and comes back up, full reachability is still maintained.

Configuration

```
R1:
interface Serial0/0
 ip ospf priority 0

R2:
interface Serial0/0
 ip ospf priority 0

R3:
interface Serial1/0
 ip ospf priority 0

R4:
interface Serial0/0/0
 ip ospf priority 0
!
interface FastEthernet0/1
 ip ospf priority 0

R6:
interface FastEthernet0/0.67
 ip ospf priority 255
!
interface FastEthernet0/0.146
 ip ospf priority 255
```

Copyright © 2010 Internetwork Expert

Verification

Note

The OSPF DR/BDR election is determined based on the interface level OSPF priority value along with the router-id. The device with the highest priority is elected the DR, and the device with the second highest priority is elected the BDR. If there is a tie in the priority value the device with the higher router-id is more likely to be elected the DR or BDR. However, in certain designs the election may become unpredictable.

This unpredictability is due to the fact that the OSPF DR/BDR election does not support preemption like IS-IS does for its DIS election. Preemption means that if a new device comes onto the segment with a higher priority or router-id, it can take the DR/BDR status away from the current device. Since OSPF does not support this, new devices must wait for a failure of the DR or BDR before the next election occurs. Additionally, in certain cases the order in which the routers load their OSPF process can influence the election.

In the below case on the LAN segment of VLAN 146, R1, R4, and R6 compete for the DR/BDR election. Since R4 has the Loopback interface 223.255.255.255/32 configured it is most likely to be elected the DR by default. This is based on the fact that the default interface priority is 1, and therefore if all three routers load the OSPF process at the same time R4 wins the election. R6 with the router-id 150.1.6.6 is next in line, with R1 having the router-id 150.1.1.1 being last.

By changing R6's interface level priority to 255 it is most likely to be elected the DR, and by changing R4's priority to 0 it means it can *never* be elected the DR or BDR. The step-by-step election can be tracked as follows.

R1's **`show ip ospf neighbor`** and **`show ip ospf interface`** output indicate that R6 is the DR with a priority of 255, R1 is the BDR with a priority of 1, and R4 is a DROTHER since it has a priority of 0.

```
Rack1R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         1   FULL/DR         00:01:58    155.1.0.5       Serial0/0
150.1.6.6       255   FULL/DR         00:00:39    155.1.146.6     FastEthernet0/0
223.255.255.255   0   FULL/DROTHER    00:00:39    155.1.146.4     FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip ospf interface FastEthernet0/0
FastEthernet0/0 is up, line protocol is up
  Internet Address 155.1.146.1/24, Area 1
  Process ID 1, Router ID 150.1.1.1, Network Type BROADCAST, Cost: 1
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State BDR, Priority 1
  Designated Router (ID) 150.1.6.6, Interface address 155.1.146.6
  Backup Designated router (ID) 150.1.1.1, Interface address 155.1.146.1
  Old designated Router (ID) 223.255.255.255, Interface address 155.1.146.4
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:01
  Supports Link-local Signaling (LLS)
  Index 1/1, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 0, maximum is 13
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 2, Adjacent neighbor count is 2
    Adjacent with neighbor 150.1.6.6  (Designated Router)
    Adjacent with neighbor 223.255.255.255
  Suppress hello for 0 neighbor(s)
```

R6's link to VLAN 146 goes down. After R1 and R4's dead timer expires a new DR/BDR election occurs.

```
Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface FastEthernet0/0.146
Rack1R6(config-subif)#shutdown
```

When R1 detects this event it asks all other neighbors on the segment to perform a new election. Since R4's priority is 0 it does not send a response back to R1. The result is that R1 is promoted from the BDR status to the DR, but there is no new BDR elected.

```
Rack1R1#debug ip ospf adj
OSPF adjacency events debugging is on

Rack1R1#
OSPF: Rcv LS UPD from 223.255.255.255 on FastEthernet0/0 length 64 LSA count 1
OSPF: Rcv LS UPD from 223.255.255.255 on FastEthernet0/0 length 64 LSA count 1
OSPF: 150.1.6.6 address 155.1.146.6 on FastEthernet0/0 is dead
OSPF: 150.1.6.6 address 155.1.146.6 on FastEthernet0/0 is dead, state DOWN
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.6.6 on FastEthernet0/0 from FULL to DOWN,
Neighbor Down: Dead timer expired
OSPF: Neighbor change Event on interface FastEthernet0/0
OSPF: DR/BDR election on FastEthernet0/0
OSPF: Elect BDR 150.1.1.1
OSPF: Elect DR 150.1.1.1
OSPF: Elect BDR 0.0.0.0
OSPF: Elect DR 150.1.1.1
        DR: 150.1.1.1 (Id)   BDR: none
OSPF: Remember old DR 150.1.6.6 (id)
OSPF: Build router LSA for area 1, router ID 150.1.1.1, seq 0x80000034
OSPF: Build network LSA for FastEthernet0/0, router ID 150.1.1.1
OSPF: Build network LSA for FastEthernet0/0, router ID 150.1.1.1
```

Copyright © 2010 Internetwork Expert

This can also be verified by the **show ip ospf neighbor** or **show ip ospf interface** output on R1 or R4. R1 sees R4 as a neighbor with a priority of 0, therefore it is a DROTHER.

```
Rack1R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         1   FULL/DR         00:01:57    155.1.0.5       Serial0/0
223.255.255.255   0   FULL/DROTHER    00:00:39    155.1.146.4     FastEthernet0/0
```

R4 indicates that there is a DR on the segment with the RID 150.1.1.1, but no BDR.

```
Rack1R4#show ip ospf interface FastEthernet0/1
FastEthernet0/1 is up, line protocol is up
  Internet Address 155.1.146.4/24, Area 1
  Process ID 1, Router ID 223.255.255.255, Network Type BROADCAST, Cost: 1
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State DROTHER, Priority 0
  Designated Router (ID) 150.1.1.1, Interface address 155.1.146.1
  No backup designated router on this network
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:06
  Supports Link-local Signaling (LLS)
  Index 1/1, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 14
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.1.1  (Designated Router)
```

Copyright © 2010 Internetwork Expert

When R6's connection to VLAN 146 comes back up R1 learns about the neighbor and a new election occurs. However, since there is no preemption for the DR election, R6 can only be elected the BDR, even though its priority is higher than R1.

```
Rack1R6(config-subif)#no shutdown
Rack1R6(config-subif)#

Rack1R1#
OSPF: Rcv DBD from 150.1.6.6 on FastEthernet0/0 seq 0x160C opt 0x52 flag 0x7
len 32  mtu 1500 state INIT
OSPF: 2 Way Communication to 150.1.6.6 on FastEthernet0/0, state 2WAY
OSPF: Neighbor change Event on interface FastEthernet0/0
OSPF: DR/BDR election on FastEthernet0/0
OSPF: Elect BDR 150.1.6.6
OSPF: Elect DR 150.1.1.1
       DR: 150.1.1.1 (Id)   BDR: 150.1.6.6 (Id)
<output omitted>

Rack1R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         1   FULL/DR         00:01:40    155.1.0.5       Serial0/0
150.1.6.6       255   FULL/BDR        00:00:30    155.1.146.6     FastEthernet0/0
223.255.255.255   0   FULL/DROTHER    00:00:32    155.1.146.4     FastEthernet0/0

Rack1R1#show ip ospf interface FastEthernet0/0
FastEthernet0/0 is up, line protocol is up
  Internet Address 155.1.146.1/24, Area 1
  Process ID 1, Router ID 150.1.1.1, Network Type BROADCAST, Cost: 1
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 150.1.1.1, Interface address 155.1.146.1
  Backup Designated router (ID) 150.1.6.6, Interface address 155.1.146.6
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:00
  Supports Link-local Signaling (LLS)
  Index 1/1, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 0, maximum is 13
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 2, Adjacent neighbor count is 2
    Adjacent with neighbor 150.1.6.6  (Backup Designated Router)
    Adjacent with neighbor 223.255.255.255
  Suppress hello for 0 neighbor(s)
```

Copyright © 2010 Internetwork Expert

For LAN segments such as VLAN 146 it technically does not matter which device is the DR or the BDR, because everyone has direct layer 2 connectivity with each other. The only design issue of the DR/BDR placement in this case is a function of the memory and CPU resources of the DR/BDR and how many neighbors are on the segment. For partially meshed NBMA networks however, a larger design issue becomes evident.

On the Frame Relay network between R1, R2, R3, R4, and R5, R5 was elected as the DR due to the order of operations in which OSPF was configured. First OSPF was enabled on R5's link, then R1, R2, R3, and R4 in that order. Since R5's OSPF process loaded first it elected itself as the DR. The neighbor with the next highest router-id, R4 with 223.255.255.255/32, was then elected the BDR. This can be seen from the **`show ip ospf interface`** output on R5. All routers still have the default interface priority of 1.

```
Rack1R5#show ip ospf interface Serial0/0/0
Serial0/0/0 is up, line protocol is up
  Internet Address 155.1.0.5/24, Area 0
  Process ID 1, Router ID 150.1.5.5, Network Type NON_BROADCAST, Cost: 64
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 150.1.5.5, Interface address 155.1.0.5
  Backup Designated router (ID) 223.255.255.255, Interface address 155.1.0.4
  Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit 5
    oob-resync timeout 120
    Hello due in 00:00:02
  Supports Link-local Signaling (LLS)
  Index 1/2, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 0, maximum is 4
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 4, Adjacent neighbor count is 4
    Adjacent with neighbor 150.1.3.3
    Adjacent with neighbor 150.1.1.1
    Adjacent with neighbor 150.1.2.2
    Adjacent with neighbor 223.255.255.255  (Backup Designated Router)
  Suppress hello for 0 neighbor(s)
```

Copyright © 2010 Internetwork Expert

When R5's link to the Frame Relay network goes down all spokes discover this after the dead timer of 120 seconds expires. A new election then occurs, and each of these routers elect themselves the DR, and no one as the BDR.

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#shutdown
Rack1R5(config-if)#

Rack1R1#debug ip ospf adj
OSPF adjacency events debugging is on
OSPF: 150.1.5.5 address 155.1.0.5 on Serial0/0 is dead
OSPF: 150.1.5.5 address 155.1.0.5 on Serial0/0 is dead, state DOWN
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.5.5 on Serial0/0 from FULL to DOWN,
Neighbor Down: Dead timer expired
OSPF: Neighbor change Event on interface Serial0/0
OSPF: DR/BDR election on Serial0/0
OSPF: Elect BDR 150.1.1.1
OSPF: Elect DR 150.1.1.1
OSPF: Elect BDR 0.0.0.0
OSPF: Elect DR 150.1.1.1
       DR: 150.1.1.1 (Id)   BDR: none

Rack1R4#debug ip ospf adj
OSPF adjacency events debugging is on
OSPF: 150.1.5.5 address 155.1.0.5 on Serial0/0/0 is dead
OSPF: 150.1.5.5 address 155.1.0.5 on Serial0/0/0 is dead, state DOWN
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.5.5 on Serial0/0/0 from FULL to DOWN,
Neighbor Down: Dead timer expired
Rack1R4#
OSPF: Neighbor change Event on interface Serial0/0/0
OSPF: DR/BDR election on Serial0/0/0
OSPF: Elect BDR 223.255.255.255
OSPF: Elect DR 223.255.255.255
OSPF: Elect BDR 0.0.0.0
OSPF: Elect DR 223.255.255.255
       DR: 223.255.255.255 (Id)   BDR: none
```

Copyright © 2010 Internetwork Expert

The problem in this design occurs when R5 tries to come back onto the segment.

```
Rack1R5#debug ip ospf adj
OSPF adjacency events debugging is on
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#no shutdown
Rack1R5(config-if)#
```

R5 sends unicast updates to the neighbors configured under the OSPF process.

```
OSPF: Starting 0.0.0.0 address 155.1.0.3 on Serial0/0/0
OSPF: Starting 0.0.0.0 address 155.1.0.1 on Serial0/0/0
OSPF: Starting 0.0.0.0 address 155.1.0.2 on Serial0/0/0
OSPF: Starting 0.0.0.0 address 155.1.0.4 on Serial0/0/0
<output omitted>
OSPF: 2 Way Communication to 150.1.3.3 on Serial0/0/0, state 2WAY
OSPF: 2 Way Communication to 150.1.1.1 on Serial0/0/0, state 2WAY
OSPF: 2 Way Communication to 150.1.2.2 on Serial0/0/0, state 2WAY
OSPF: 2 Way Communication to 223.255.255.255 on Serial0/0/0, state 2WAY
```

R5 finds four neighbors, and sees that R4 has the highest router-id of 223.255.255.255, and is the DR. The next highest router-id is 150.1.5.5, R5, and it elects itself as the BDR. R1, R2, and R3 revert to DROTHERS.

```
OSPF: Neighbor change Event on interface Serial0/0/0
OSPF: DR/BDR election on Serial0/0/0
OSPF: Elect BDR 0.0.0.0
OSPF: Elect DR 223.255.255.255
OSPF: Elect BDR 150.1.5.5
OSPF: Elect DR 223.255.255.255
       DR: 223.255.255.255 (Id)   BDR: 150.1.5.5 (Id)
<output omitted>
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.3.3 on Serial0/0/0 from LOADING to FULL,
Loading Done
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.1.1 on Serial0/0/0 from LOADING to FULL,
Loading Done
%OSPF-5-ADJCHG: Process 1, Nbr 223.255.255.255 on Serial0/0/0 from LOADING to
FULL, Loading Done
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.2.2 on Serial0/0/0 from LOADING to FULL,
Loading Done
<output omitted>

Rack1R5#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.3.3         1   FULL/DROTHER    00:01:39    155.1.0.3       Serial0/0/0
150.1.1.1         1   FULL/DROTHER    00:01:44    155.1.0.1       Serial0/0/0
150.1.2.2         1   FULL/DROTHER    00:01:55    155.1.0.2       Serial0/0/0
223.255.255.255   1   FULL/DR         00:01:41    155.1.0.4       Serial0/0/0
150.1.8.8         1   FULL/DR         00:00:36    155.1.58.8      FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

Due to the desired function of the DR, the network design is now broken. For example in order for R2 to advertise the network 192.10.1.0/24, this update must be sent to the DR before being sent to the rest of the neighbors on the segment. However R2 does not have a direct PVC on the Frame Relay network connecting to R4 who is the DR. The result is that LSA replication inside area 0 is now incomplete, and different routers end up with different views of the topology. Based on this inconsistency of information SPF is not uniform, and some routers have reachability to some segments, while others do not. The result of this can be seen in either the OSPF database or the routing tables of the devices in question.

R1 thinks that R5 is the DR, but R5 thinks that R4 is the DR. LSAs that R1 directs to R5 are dropped, and LSAs from R4 that should be replicated back to R1 cannot be. The result is that R1 has no routes installed in the routing table.

```
Rack1R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         1   FULL/DR         00:01:47    155.1.0.5       Serial0/0
150.1.6.6       255   FULL/BDR        00:00:37    155.1.146.6     FastEthernet0/0
223.255.255.255   0   FULL/DROTHER    00:00:39    155.1.146.4     FastEthernet0/0

Rack1R1#show ip route ospf
```

R2 has the same result as R1.

```
Rack1R2#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         1   FULL/DR         00:01:40    155.1.0.5       Serial0/0
192.10.1.254      1   FULL/DR         00:00:37    192.10.1.254    FastEthernet0/0

Rack1R2#show ip route ospf
     51.0.0.0/32 is subnetted, 1 subnets
O E2    51.51.51.51 [110/20] via 192.10.1.254, 00:01:13, FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

R3 knows about area 2 routes, but cannot install Inter-Area routes that should come from other area 0 routers.

```
Rack1R3#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         1   FULL/DR         00:01:31    155.1.0.5       Serial1/0
150.1.7.7         1   FULL/BDR        00:00:33    155.1.37.7      FastEthernet0/0

Rack1R3#show ip route ospf
     155.1.0.0/24 is subnetted, 8 subnets
O       155.1.9.0 [110/3] via 155.1.37.7, 04:48:25, FastEthernet0/0
O       155.1.7.0 [110/2] via 155.1.37.7, 04:48:25, FastEthernet0/0
O       155.1.79.0 [110/2] via 155.1.37.7, 04:48:25, FastEthernet0/0
O       155.1.67.0 [110/2] via 155.1.37.7, 04:48:25, FastEthernet0/0
     150.1.0.0/16 is variably subnetted, 2 subnets, 2 masks
O       150.1.7.7/32 [110/2] via 155.1.37.7, 04:48:25, FastEthernet0/0
```

The process continues as such depending on where the OSPF topology is viewed from. Therefore, at this point in the configuration the network design works in some cases, but not in all cases. In order to ensure that the network works in *all* cases, we must guarantee that R5 is always elected the DR. This is due to the fact that R5 is the only neighbor who can form a direct adjacency with all other devices on the Frame Relay segment, as a result of the way the layer 2 circuits are provisioned.

The logic of the solution, in this particular case, is somewhat backwards due to how OSPF deals with preemption. Even if R5 is configured with a priority value of 255, it cannot preempt whichever router elected itself as the DR after it went down. The only thing R5 can do is re-elect itself as the BDR. Therefore, instead of ensuring that R5 *is* elected the DR, we need to ensure that R1, R2, R3, and R4 *are not* elected the DR. This is accomplished by configuring the OSPF priority value as 0 at the interface level of each of these devices. Since priority 0 means they will not participate in the election, R5 is the only "candidate" able to be elected the DR and additionally no one is eligible to be elected the BDR. If R5 goes down Inter-Area routing is lost, however when R5 comes back it is fully restored.

Verification of this can be seen from the **`show ip ospf neighbor`** output on R5. With the remote devices configured with priority 0 R5 is elected the DR, while all other devices revert back to DROTHERs.

```
Rack1R5#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.3.3         0   FULL/DROTHER    00:01:56    155.1.0.3       Serial0/0/0
150.1.1.1         0   FULL/DROTHER    00:01:41    155.1.0.1       Serial0/0/0
150.1.2.2         0   FULL/DROTHER    00:01:51    155.1.0.2       Serial0/0/0
223.255.255.255   0   FULL/DROTHER    00:01:38    155.1.0.4       Serial0/0/0
150.1.8.8         1   FULL/DR         00:00:33    155.1.58.8      FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

Once the design configuration is complete it can also be verified by the **`show ip ospf database`** output. Only the DR on an OSPF segment originates the Network LSA (LSA 2). The Network LSA is used to describe all the neighbors that the DR on the segment is adjacent with to the BDR and DROTHERs on that link.

From SW3's below output the *Link ID* under *Net Link States (Area 2)* is the interface IP address of the DR, and the *ADV Router* is the router-id of the DR. Based on this we can see that R3 is the DR for the link from R3 to SW1, R6 is the DR for VLAN 67, and SW3 is the DR for VLAN 79. SW3 does not see Network LSAs for other areas, as they are only needed for Intra-Area SPF calculations.

```
Rack1SW3#show ip ospf database

            OSPF Router with ID (150.1.9.9) (Process ID 1)

                Router Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum Link count
150.1.3.3       150.1.3.3       660         0x80000051 0x0061BB 1
150.1.6.6       150.1.6.6       1019        0x80000050 0x00D8F7 1
150.1.7.7       150.1.7.7       894         0x80000055 0x0005C0 5
150.1.9.9       150.1.9.9       670         0x8000004F 0x00B13F 2

                Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.37.3      150.1.3.3       660         0x8000004E 0x0082C4
155.1.67.6      150.1.6.6       1019        0x8000004E 0x002BEE
155.1.79.9      150.1.9.9       671         0x8000004E 0x009A64
<output omitted>
```

Copyright © 2010 Internetwork Expert

R1 participates in both area 0 and area 1 SPF, so it sees the Network LSAs coming from R5 for the Frame Relay segment, and R6 for VLAN 146. This means that R5 and R6 are the DRs for those segments respectively.

```
Rack1R1#show ip ospf database

            OSPF Router with ID (150.1.1.1) (Process ID 1)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
150.1.1.1       150.1.1.1       212         0x80000030 0x002130 1
150.1.2.2       150.1.2.2       219         0x80000032 0x000941 1
150.1.3.3       150.1.3.3       213         0x80000032 0x00DBEA 2
150.1.5.5       150.1.5.5       218         0x80000040 0x009F76 3
223.255.255.255 223.255.255.255 219         0x80000037 0x00568C 3

                Net Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.0.5       150.1.5.5       213         0x80000002 0x00DEB6

<output omitted>

                Net Link States (Area 1)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.146.6     150.1.6.6       294         0x80000001 0x00E05F

<output omitted>
```

SW4 only sees Network LSAs for area 3 because it does not participate in other areas. The below output indicates that SW2 is the DR for the VLAN 58 segment and SW4 is the DR for the layer 3 EtherChannel connecting to SW2.

```
Rack1SW4#show ip ospf database

            OSPF Router with ID (223.255.255.255) (Process ID 1)

                Router Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum Link count
150.1.5.5       150.1.5.5       1856        0x80000052 0x00F13E 2
150.1.8.8       150.1.8.8       1660        0x80000056 0x0033B3 3
223.255.255.255 223.255.255.255 1506        0x80000051 0x0056EE 2

                Net Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.58.8      150.1.8.8       1660        0x8000004E 0x0050CC
155.1.108.10    223.255.255.255 1506        0x8000004E 0x00A2CF

<output omitted>
```

Copyright © 2010 Internetwork Expert

6.4 OSPF Network Point-to-Point

- Configure OSPF area 0 on the point-to-point Serial link between R4 & R5.
- Configure OSPF area 4 on the point-to-point Serial link between R1 & R3.
- Configure OSPF area 5 on the point-to-point Serial link between R2 & R3.

Configuration

```
R1:
interface Serial0/1
 ip ospf 1 area 4

R2:
interface Serial0/1
 ip ospf 1 area 5

R3:
interface Serial1/2
 ip ospf 1 area 4
!
interface Serial1/3
 ip ospf 1 area 5

R4:
interface Serial0/1/0
 ip ospf 1 area 0

R5:
interface Serial0/1/0
 ip ospf 1 area 0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> OSPF network point-to-point is the default option for point-to-point interfaces such as HDLC, PPP, or point-to-point NBMA subinterfaces. It uses multicast hellos, does not use the DR/BDR election, and only supports the adjacency of exactly two neighbors on a segment. No special design considerations need to be taken into account for point-to-point OSPF interfaces.

```
Rack1R4#show ip ospf interface Serial0/1/0
Serial0/1/0 is up, line protocol is up
  Internet Address 155.1.45.4/24, Area 0
 Process ID 1, Router ID 223.255.255.255, Network Type POINT_TO_POINT, Cost: 64
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:02
  Supports Link-local Signaling (LLS)
  Index 2/3, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.5.5
  Suppress hello for 0 neighbor(s)

Rack1R5#show ip ospf interface Serial0/1/0
Serial0/1/0 is up, line protocol is up
  Internet Address 155.1.45.5/24, Area 0
  Process ID 1, Router ID 150.1.5.5, Network Type POINT_TO_POINT, Cost: 64
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:09
  Supports Link-local Signaling (LLS)
  Index 2/4, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 4
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 223.255.255.255
  Suppress hello for 0 neighbor(s)
```

Copyright © 2010 Internetwork Expert

The null output after the slash in the *State* column in the **`show ip ospf neighbor`** output indicates that no DR or BDR is elected for network type point-to-point.

```
Rack1R4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         0   FULL/  -        00:00:37    155.1.45.5      Serial0/1/0
150.1.5.5         1   FULL/DR         00:01:50    155.1.0.5       Serial0/0/0
150.1.1.1         1   FULL/DR         00:00:32    155.1.146.1     FastEthernet0/1
150.1.6.6       255   FULL/BDR        00:00:30    155.1.146.6     FastEthernet0/1

Rack1R5#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
223.255.255.255   0   FULL/  -        00:00:38    155.1.45.4      Serial0/1/0
150.1.3.3         0   FULL/DROTHER    00:01:35    155.1.0.3       Serial0/0/0
150.1.1.1         0   FULL/DROTHER    00:01:50    155.1.0.1       Serial0/0/0
150.1.2.2         0   FULL/DROTHER    00:01:30    155.1.0.2       Serial0/0/0
223.255.255.255   0   FULL/DROTHER    00:01:47    155.1.0.4       Serial0/0/0
150.1.8.8         1   FULL/DR         00:00:31    155.1.58.8      FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

6.5 OSPF Network Point-to-Multipoint

- Remove all **`frame-relay map`** statements on R1, R2, R3, and R4 with the exception of the mapping to R5, and note any changes in reachability throughout the OSPF domain.
- Modify the Frame Relay network's OSPF configuration to use network type point-to-multipoint, and note any changes in the OSPF database, the routing table, and in reachability throughout the network.

Configuration

```
R1:
interface Serial0/0
 ip ospf network point-to-multipoint
 no frame-relay map ip 155.1.0.2 105
 no frame-relay map ip 155.1.0.3 105
 no frame-relay map ip 155.1.0.4 105
 frame-relay map ip 155.1.0.5 105 broadcast

R2:
interface Serial0/0
 ip ospf network point-to-multipoint
 no frame-relay map ip 155.1.0.1 205
 no frame-relay map ip 155.1.0.3 205
 no frame-relay map ip 155.1.0.4 205
 frame-relay map ip 155.1.0.5 205 broadcast

R3:
interface Serial1/0
 ip ospf network point-to-multipoint
 no frame-relay map ip 155.1.0.1 305
 no frame-relay map ip 155.1.0.2 305
 no frame-relay map ip 155.1.0.4 305
 frame-relay map ip 155.1.0.5 305 broadcast

R4:
interface Serial0/0/0
 ip ospf network point-to-multipoint
 no frame-relay map ip 155.1.0.1 405
 no frame-relay map ip 155.1.0.2 405
 no frame-relay map ip 155.1.0.3 405
 frame-relay map ip 155.1.0.5 405 broadcast

R5:
interface Serial0/0/0
 ip ospf network point-to-multipoint
 frame-relay map ip 155.1.0.1 501 broadcast
 frame-relay map ip 155.1.0.2 502 broadcast
 frame-relay map ip 155.1.0.3 503 broadcast
 frame-relay map ip 155.1.0.4 504 broadcast
```

Copyright © 2010 Internetwork Expert

Verification

Note

OSPF network type point-to-multipoint is specifically designed to solve reachability problems in partially meshed NBMA network designs. Like network type point-to-point, it sends hellos as multicasts and does not support the DR/BDR election. Unlike point-to-point however, multiple adjacencies on a single interface are supported.

In this particular scenario the first design issue that must be solved is allowing the transport of multicast OSPF packets over the Frame Relay circuits. Since the initial configurations exclude the broadcast keyword from the **`frame-relay map`** statements configured, the **`debug ip packet`** and **`debug frame-relay packet`** outputs indicate that the encapsulation is failing when the OSPF multicast 224.0.0.5 is sent out the Frame Relay circuit. Note that this was not an issue when using network type non-broadcast, because non-broadcast uses unicast hellos, and therefore does not require multicast transport.

```
Rack1R5#debug ip packet
IP packet debugging is on

Rack1R5#debug frame-relay packet
Frame Relay packet debugging is on

IP: s=155.1.45.5 (local), d=224.0.0.5 (Serial0/1/0), len 80, sending
broad/multicast
IP: s=155.1.5.5 (local), d=224.0.0.5 (FastEthernet0/1), len 76, sending
broad/multicast
IP: s=155.1.58.5 (local), d=224.0.0.5 (FastEthernet0/0), len 80, sending
broad/multicast
IP: s=155.1.0.5 (local), d=224.0.0.5 (Serial0/0/0), len 76, sending
broad/multicast
Serial0/0/0: broadcast search
Serial0/0/0:encaps failed on broadcast for link 7(IP)
IP: s=155.1.0.5 (local), d=224.0.0.5 (Serial0/0/0), len 76, encapsulation
failed
```

Copyright © 2010 Internetwork Expert

To resolve this, the **broadcast** keyword must be added on the Frame Relay mappings from R5 to R1, R2, R3, & R4, and from R1, R2, R3, & R4 back to R5.

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0
Rack1R5(config-if)# frame-relay map ip 155.1.0.1 501 broadcast
Rack1R5(config-if)# frame-relay map ip 155.1.0.2 502 broadcast
Rack1R5(config-if)# frame-relay map ip 155.1.0.3 503 broadcast
Rack1R5(config-if)# frame-relay map ip 155.1.0.4 504 broadcast
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.3.3 on Serial0/0 from LOADING to FULL,
Loading Done
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.2.2 on Serial0/0 from LOADING to FULL,
Loading Done
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.4.4 on Serial0/0 from LOADING to FULL,
Loading Done
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.1.1 on Serial0/0 from LOADING to FULL,
Loading Done
```

Once adjacency is established the **show ip ospf neighbor** output indicates that there is no DR or BDR for the segment with the null output in the *State* field.

```
Rack1R5#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.1.1         0   FULL/  -        00:01:59    155.1.0.1       Serial0/0/0
150.1.3.3         0   FULL/  -        00:01:59    155.1.0.3       Serial0/0/0
150.1.2.2         0   FULL/  -        00:01:59    155.1.0.2       Serial0/0/0
223.255.255.255   0   FULL/  -        00:01:59    155.1.0.4       Serial0/0/0
223.255.255.255   0   FULL/  -        00:00:36    155.1.45.4      Serial0/1/0
150.1.8.8         1   FULL/DR         00:00:35    155.1.58.8      FastEthernet0/0
```

The spokes of the network only form adjacency with R5, and not each other, because layer 2 PVCs only exist between the hub and the spokes.

```
Rack1R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         0   FULL/  -        00:01:54    155.1.0.5       Serial0/0
150.1.6.6       255   FULL/BDR        00:00:30    155.1.146.6     FastEthernet0/0
223.255.255.255   0   FULL/DROTHER    00:00:32    155.1.146.4     FastEthernet0/0
150.1.3.3         0   FULL/  -        00:00:31    155.1.13.3      Serial0/1

Rack1R2#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         0   FULL/  -        00:01:45    155.1.0.5       Serial0/0
150.1.3.3         0   FULL/  -        00:00:36    155.1.23.3      Serial0/1
192.10.1.254      1   FULL/DR         00:00:36    192.10.1.254    FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         0   FULL/  -        00:01:33    155.1.0.5       Serial1/0
150.1.7.7         1   FULL/BDR        00:00:32    155.1.37.7      FastEthernet0/0
150.1.1.1         0   FULL/  -        00:00:34    155.1.13.1      Serial1/2
150.1.2.2         0   FULL/  -        00:00:32    155.1.23.2      Serial1/3

Rack1R4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         0   FULL/  -        00:01:39    155.1.0.5       Serial0/0/0
150.1.5.5         0   FULL/  -        00:00:33    155.1.45.5      Serial0/1/0
150.1.1.1         1   FULL/DR         00:00:37    155.1.146.1     FastEthernet0/1
150.1.6.6       255   FULL/BDR        00:00:35    155.1.146.6     FastEthernet0/1
```

The output from the **`show ip ospf database`** in area 0 also reinforces the fact that there is no DR or BDR for the segment. Recall from the DR/BDR election section that the Network LSA (LSA 2) is used to describe the DR and its attached neighbors. Since there are no DRs for any transit links in area 0, the *Net Link States (Area 0)* section does not appear in the below output.

```
Rack1R5#show ip ospf database

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
150.1.1.1       150.1.1.1       989         0x8000002E 0x004660 2
150.1.2.2       150.1.2.2       1386        0x80000030 0x00445A 2
150.1.3.3       150.1.3.3       1065        0x80000030 0x00DFE8 2
150.1.5.5       150.1.5.5       1119        0x8000003D 0x00EB9B 7
223.255.255.255 223.255.255.255 1522        0x80000033 0x001F18 4

                Summary Net Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum
150.1.7.7       150.1.3.3       322         0x80000028 0x008F41
155.1.5.0       150.1.5.5       1119        0x80000028 0x00864B
155.1.7.0       150.1.3.3       322         0x80000028 0x00943E
155.1.8.0       150.1.5.5       1119        0x80000028 0x006F5E
155.1.9.0       150.1.3.3       322         0x80000028 0x008847
155.1.10.0      150.1.5.5       1119        0x80000028 0x006367
155.1.13.0      150.1.1.1       1243        0x8000001D 0x00F0AC
155.1.13.0      150.1.3.3       1823        0x8000001D 0x00F1D7
155.1.23.0      150.1.2.2       1876        0x8000001D 0x00751C
155.1.23.0      150.1.3.3       1823        0x8000001D 0x00833C
155.1.37.0      150.1.3.3       323         0x80000028 0x003F76
155.1.58.0      150.1.5.5       1120        0x80000028 0x003D5F
155.1.67.0      150.1.3.3       323         0x80000028 0x00FD98
155.1.79.0      150.1.3.3       323         0x80000028 0x007911
155.1.108.0     150.1.5.5       1120        0x80000028 0x001F4A
155.1.146.0     150.1.1.1       2008        0x80000027 0x00A7A5
155.1.146.0     223.255.255.255 1523        0x80000028 0x0027DE
192.10.1.0      150.1.2.2       130         0x80000028 0x008A23

<output omitted>
```

Copyright © 2010 Internetwork Expert

Once LSA replication is complete the next change that should be evident is how the routing table is processed on the Frame Relay segment between the neighbors. For example take the following view of the network from R2's perspective.

```
Rack1R2#show ip route ospf
     51.0.0.0/32 is subnetted, 1 subnets
O E2    51.51.51.51 [110/20] via 192.10.1.254, 00:02:33, FastEthernet0/0
     155.1.0.0/16 is variably subnetted, 19 subnets, 2 masks
O IA    155.1.146.0/24 [110/129] via 155.1.0.5, 00:02:33, Serial0/0
O IA    155.1.10.0/24 [110/67] via 155.1.0.5, 00:02:33, Serial0/0
O IA    155.1.8.0/24 [110/66] via 155.1.0.5, 00:02:33, Serial0/0
O IA    155.1.9.0/24 [110/131] via 155.1.0.5, 00:02:33, Serial0/0
O IA    155.1.13.0/24 [110/192] via 155.1.0.5, 00:02:33, Serial0/0
O       155.1.0.3/32 [110/128] via 155.1.0.5, 00:02:33, Serial0/0
O       155.1.0.1/32 [110/128] via 155.1.0.5, 00:02:33, Serial0/0
O IA    155.1.7.0/24 [110/130] via 155.1.0.5, 00:02:33, Serial0/0
O       155.1.0.4/32 [110/128] via 155.1.0.5, 00:02:33, Serial0/0
O IA    155.1.5.0/24 [110/65] via 155.1.0.5, 00:02:33, Serial0/0
O       155.1.0.5/32 [110/64] via 155.1.0.5, 00:02:33, Serial0/0
O IA    155.1.58.0/24 [110/65] via 155.1.0.5, 00:02:33, Serial0/0
O       155.1.45.0/24 [110/128] via 155.1.0.5, 00:02:34, Serial0/0
O IA    155.1.37.0/24 [110/129] via 155.1.0.5, 00:02:34, Serial0/0
O IA    155.1.79.0/24 [110/130] via 155.1.0.5, 00:02:34, Serial0/0
O IA    155.1.67.0/24 [110/130] via 155.1.0.5, 00:02:34, Serial0/0
O IA    155.1.108.0/24 [110/66] via 155.1.0.5, 00:02:34, Serial0/0
     150.1.0.0/16 is variably subnetted, 2 subnets, 2 masks
O IA    150.1.7.7/32 [110/130] via 155.1.0.5, 00:02:34, Serial0/0
```

For all routes that are learned from the hub of the Frame Relay network the next-hop value has been updated to the interface IP address of the hub. Recall from the previous design with the non-broadcast network type that the DR did not update the next-hop. The result of this is that when route recursion is performed for traffic that needs to be routed between the spokes, *layer 3 to layer 2 resolution now occurs for the hub, not the spoke*.

Specifically, we can see this on R2 for its lookup to reach the VLAN 9 segment. R2's longest match for 155.1.9.9 is 155.1.9.0/24. The evident change here is that the next-hop value is not the LSA 3 originator, who is R3, but the hub, R5. Note the field *from 150.1.3.3* still tells R2 that R3 is the LSA 3 originator.

```
Rack1R2#show ip route 155.1.9.9
Routing entry for 155.1.9.0/24
  Known via "ospf 1", distance 110, metric 131, type inter area
  Last update from 155.1.0.5 on Serial0/0, 00:02:48 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.3.3, 00:02:48 ago, via Serial0/0
      Route metric is 131, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

R2 now does a recursive lookup on 155.1.0.5, and it points to an OSPF host route for 155.1.0.5/32 that is learned from R5 via Serial0/0.

```
Rack1R2#show ip route 155.1.0.5
Routing entry for 155.1.0.5/32
  Known via "ospf 1", distance 110, metric 64, type intra area
  Last update from 155.1.0.5 on Serial0/0, 00:02:59 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.5.5, 00:02:59 ago, via Serial0/0
      Route metric is 64, traffic share count is 1
```

This implies that R2 must now resolve 155.1.0.5 at the Frame Relay layer. Since there is a static **`frame-relay map`** statement pointing towards 155.1.0.5, R2 knows to use PVC 205 when the layer 2 frame is built.

```
Rack1R2#show frame-relay map | include 155.1.0.5
Serial0/0 (up): ip 155.1.0.5 dlci 205(0xCD,0x30D0), static,
```

The end result is end-to-end reachability.

```
Rack1R2#ping 155.1.9.9

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.9.9, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/116/124 ms
```

The design implication of this change is that the spokes of the NBMA network no longer need to maintain detailed mapping tables for all the other spokes. For example take the Frame Relay interface configuration of R2 below.

```
Rack1R2#show run interface Serial0/0
Building configuration...

Current configuration : 237 bytes
!
interface Serial0/0
 ip address 155.1.0.2 255.255.255.0
 encapsulation frame-relay
 ip ospf network point-to-multipoint
 ip ospf priority 0
 ip ospf 1 area 0
 frame-relay map ip 155.1.0.5 205 broadcast
 no frame-relay inverse-arp
end
```

Copyright © 2010 Internetwork Expert

R2 has a **`frame-relay map`** statement pointing to R5, but not R1, R3, or R4. When R2 does a layer 3 routing lookup on R1's Frame Relay address, 155.1.0.1, it shows that it is learned from R5 as a host route. In the previous non-broadcast design R2 would have seen this as directly connected. In this case R2 now says that 155.1.0.1 is via 155.1.0.5, and 155.1.0.5 is via Serial0/0.

```
Rack1R2#show ip route 155.1.0.1
Routing entry for 155.1.0.1/32
  Known via "ospf 1", distance 110, metric 128, type intra area
  Last update from 155.1.0.5 on Serial0/0, 00:08:18 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.1.1, 00:08:18 ago, via Serial0/0
      Route metric is 128, traffic share count is 1

Rack1R2#show ip route 155.1.0.5
Routing entry for 155.1.0.5/32
  Known via "ospf 1", distance 110, metric 64, type intra area
  Last update from 155.1.0.5 on Serial0/0, 00:08:21 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.5.5, 00:08:21 ago, via Serial0/0
      Route metric is 64, traffic share count is 1

This implies that as long as R2 has a mapping for 155.1.0.5 in the Frame Relay
table, encapsulation is successful.

Rack1R2#ping 155.1.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/115/120 ms
```

Likewise R1 says that 155.1.0.2 is via 155.1.0.5, and 155.1.0.5 is via Serial0/0. Since R1 has a mapping for 155.1.0.5 the traffic flow is successful.

```
Rack1R1#show ip route 155.1.0.2
Routing entry for 155.1.0.2/32
  Known via "ospf 1", distance 110, metric 128, type intra area
  Last update from 155.1.0.5 on Serial0/0, 00:11:52 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.2.2, 00:11:52 ago, via Serial0/0
      Route metric is 128, traffic share count is 1

Rack1R1#show ip route 155.1.0.5
Routing entry for 155.1.0.5/32
  Known via "ospf 1", distance 110, metric 64, type intra area
  Last update from 155.1.0.5 on Serial0/0, 00:11:55 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.5.5, 00:11:55 ago, via Serial0/0
      Route metric is 64, traffic share count is 1

Rack1R1#ping 155.1.0.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/116/125 ms
```

Copyright © 2010 Internetwork Expert

From views outside of the connected segment the point-to-multipoint network is only seen as a collection of endpoints, not a transit segment itself. For example take the following output on SW4 who is in area 3.

```
Rack1SW4#show ip route 155.1.0.0 255.255.255.0
% Subnet not in table

Rack1SW4#show ip route | include 155.1.0.
     155.1.0.0/16 is variably subnetted, 19 subnets, 2 masks
O IA    155.1.0.2/32 [110/66] via 155.1.108.8, 00:15:21, Port-channel1
O IA    155.1.0.3/32 [110/66] via 155.1.108.8, 00:15:21, Port-channel1
O IA    155.1.0.1/32 [110/66] via 155.1.108.8, 00:15:21, Port-channel1
O IA    155.1.0.4/32 [110/66] via 155.1.108.8, 00:15:31, Port-channel1
O IA    155.1.0.5/32 [110/2] via 155.1.108.8, 00:15:31, Port-channel1
```

SW4 does not have an exact match for the prefix 155.1.0.0/24, which is the actual subnet assignment of the Frame Relay link. Instead it knows that there are five endpoints on the network, 155.1.0.1, 155.1.0.2, 155.1.0.3, 155.1.0.4, and 155.1.0.5. This is the normal and desirable behavior for OSPF network type point-to-multipoint per the RFC standard.

Copyright © 2010 Internetwork Expert

6.6 OSPF Network Point-to-Multipoint Non-Broadcast

- Remove the **broadcast** keyword from the **frame-relay map** statements of R1, R2, R3, R4, and R5, and note any changes in reachability throughout the OSPF domain.
- Modify the Frame Relay network's OSPF configuration to use network type point-to-multipoint non-broadcast, and note any changes in the OSPF database, the routing table, and in reachability throughout the network.

Configuration

```
R1:
interface Serial0/0
 ip ospf network point-to-multipoint non-broadcast
 no frame-relay map ip 155.1.0.5 105 broadcast
 frame-relay map ip 155.1.0.5 105

R2:
interface Serial0/0
 ip ospf network point-to-multipoint non-broadcast
 no frame-relay map ip 155.1.0.5 205 broadcast
 frame-relay map ip 155.1.0.5 205

R3:
interface Serial1/0
 ip ospf network point-to-multipoint non-broadcast
 no frame-relay map ip 155.1.0.5 305 broadcast
 frame-relay map ip 155.1.0.5 305

R4:
interface Serial0/0/0
 ip ospf network point-to-multipoint non-broadcast
 no frame-relay map ip 155.1.0.5 405 broadcast
 frame-relay map ip 155.1.0.5 405

R5:
interface Serial0/0/0
 ip ospf network point-to-multipoint non-broadcast
 no frame-relay map ip 155.1.0.1 501 broadcast
 no frame-relay map ip 155.1.0.2 502 broadcast
 no frame-relay map ip 155.1.0.3 503 broadcast
 no frame-relay map ip 155.1.0.4 504 broadcast
 frame-relay map ip 155.1.0.1 501
 frame-relay map ip 155.1.0.2 502
 frame-relay map ip 155.1.0.3 503
 frame-relay map ip 155.1.0.4 504
!
router ospf 1
 neighbor 155.1.0.1
 neighbor 155.1.0.2
 neighbor 155.1.0.3
 neighbor 155.1.0.4
```

Copyright © 2010 Internetwork Expert

Verification

Note

OSPF network type point-to-multipoint non-broadcast is essentially the same as network type point-to-multipoint, with one exception. Point-to-multipoint network type uses multicast hellos, while point-to-multipoint non-broadcast uses unicast hellos. Both do not support the DR/BDR election, automatically update the next-hop value of routes learned on partially meshed networks to the directly connected neighbor, and advertise the network as a set of endpoints instead of a transit network.

The **`show ip ospf neighbor`** output is identical between the two. In this case we can see that the spokes are adjacent with R5, the hub, and the hub is adjacent with the spokes. The null field under State indicates that no DR/BDR election has occurred.

```
Rack1R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         0   FULL/  -        00:01:50    155.1.0.5       Serial0/0
150.1.6.6       255   FULL/DR         00:00:37    155.1.146.6     FastEthernet0/0
223.255.255.255   0   FULL/DROTHER    00:00:30    155.1.146.4     FastEthernet0/0
150.1.3.3         0   FULL/  -        00:00:37    155.1.13.3      Serial0/1

Rack1R2#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         0   FULL/  -        00:01:44    155.1.0.5       Serial0/0
150.1.3.3         0   FULL/  -        00:00:35    155.1.23.3      Serial0/1
192.10.1.254      1   FULL/DR         00:00:39    192.10.1.254    FastEthernet0/0

Rack1R3#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         0   FULL/  -        00:01:34    155.1.0.5       Serial1/0
150.1.7.7         1   FULL/BDR        00:00:39    155.1.37.7      FastEthernet0/0
150.1.1.1         0   FULL/  -        00:00:38    155.1.13.1      Serial1/2
150.1.2.2         0   FULL/  -        00:00:31    155.1.23.2      Serial1/3

Rack1R4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         0   FULL/  -        00:01:55    155.1.0.5       Serial0/0/0
150.1.5.5         0   FULL/  -        00:00:36    155.1.45.5      Serial0/1/0
150.1.1.1         1   FULL/BDR        00:00:33    155.1.146.1     FastEthernet0/1
150.1.6.6       255   FULL/DR         00:00:36    155.1.146.6     FastEthernet0/1
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.1.1         0   FULL/  -        00:01:48    155.1.0.1       Serial0/0/0
150.1.3.3         0   FULL/  -        00:01:48    155.1.0.3       Serial0/0/0
150.1.2.2         0   FULL/  -        00:01:48    155.1.0.2       Serial0/0/0
223.255.255.255   0   FULL/  -        00:01:48    155.1.0.4       Serial0/0/0
223.255.255.255   0   FULL/  -        00:00:32    155.1.45.4      Serial0/1/0
150.1.8.8         1   FULL/DR         00:00:30    155.1.58.8      FastEthernet0/0
```

The routing table processing is the same between both these types. For example R2 sees the network 155.1.7.0/24 as reachable through R5, not through the originator, R3. This means that R2 does not need a **`frame-relay map`** statement to R3, it just needs one towards R5.

```
Rack1R2#show ip route ospf
     51.0.0.0/32 is subnetted, 1 subnets
O E2    51.51.51.51 [110/20] via 192.10.1.254, 00:06:24, FastEthernet0/0
     155.1.0.0/16 is variably subnetted, 19 subnets, 2 masks
O IA    155.1.146.0/24 [110/129] via 155.1.0.5, 00:06:34, Serial0/0
O IA    155.1.10.0/24 [110/67] via 155.1.0.5, 00:06:34, Serial0/0
O IA    155.1.8.0/24 [110/66] via 155.1.0.5, 00:06:34, Serial0/0
O IA    155.1.9.0/24 [110/131] via 155.1.0.5, 00:06:34, Serial0/0
O IA    155.1.13.0/24 [110/192] via 155.1.0.5, 00:06:34, Serial0/0
O       155.1.0.3/32 [110/128] via 155.1.0.5, 00:06:34, Serial0/0
O       155.1.0.1/32 [110/128] via 155.1.0.5, 00:06:34, Serial0/0
O IA    155.1.7.0/24 [110/130] via 155.1.0.5, 00:06:34, Serial0/0
O       155.1.0.4/32 [110/128] via 155.1.0.5, 00:06:34, Serial0/0
O IA    155.1.5.0/24 [110/65] via 155.1.0.5, 00:06:35, Serial0/0
O       155.1.0.5/32 [110/64] via 155.1.0.5, 00:06:35, Serial0/0
O IA    155.1.58.0/24 [110/65] via 155.1.0.5, 00:06:35, Serial0/0
O       155.1.45.0/24 [110/128] via 155.1.0.5, 00:06:35, Serial0/0
O IA    155.1.37.0/24 [110/129] via 155.1.0.5, 00:06:35, Serial0/0
O IA    155.1.79.0/24 [110/130] via 155.1.0.5, 00:06:35, Serial0/0
O IA    155.1.67.0/24 [110/130] via 155.1.0.5, 00:06:35, Serial0/0
O IA    155.1.108.0/24 [110/66] via 155.1.0.5, 00:06:35, Serial0/0
     150.1.0.0/16 is variably subnetted, 2 subnets, 2 masks
O IA    150.1.7.7/32 [110/130] via 155.1.0.5, 00:06:36, Serial0/0
```

Like point-to-multipoint, when neighbors outside area 0 see this segment it appears as a collection of endpoints, not the transit subnet itself.

```
Rack1SW4#show ip route ospf | include 155.1.0.
     155.1.0.0/16 is variably subnetted, 19 subnets, 2 masks
O IA    155.1.0.2/32 [110/66] via 155.1.108.8, 00:07:09, Port-channel1
O IA    155.1.0.3/32 [110/66] via 155.1.108.8, 00:07:09, Port-channel1
O IA    155.1.0.1/32 [110/66] via 155.1.108.8, 00:07:09, Port-channel1
O IA    155.1.0.4/32 [110/66] via 155.1.108.8, 04:15:58, Port-channel1
O IA    155.1.0.5/32 [110/2] via 155.1.108.8, 04:15:58, Port-channel1
```

Copyright © 2010 Internetwork Expert

The difference between the two types can be seen in the **debug ip packet** output. R5 sends multicast hellos to 224.0.0.5 out the other LAN interfaces which use network type broadcast, along with the point-to-point Serial link to R4 that uses network type point-to-point. Out the Frame Relay link running point-to-multipoint non-broadcast R5 sends unicast hellos. This implies that the neighbor statement needs to be configured under the OSPF process, like the non-broadcast network type, to tell the router which devices to send hellos to.

```
Rack1R5#debug ip packet detail
IP packet debugging is on (detailed)
IP: s=155.1.45.5 (local), d=224.0.0.5 (Serial0/1/0), len 80, sending
broad/multicast, proto=89
IP: s=155.1.5.5 (local), d=224.0.0.5 (FastEthernet0/1), len 76, sending
broad/multicast, proto=89
IP: s=155.1.58.5 (local), d=224.0.0.5 (FastEthernet0/0), len 80, sending
broad/multicast, proto=89
IP: s=155.1.0.5 (local), d=155.1.0.1 (Serial0/0/0), len 76, sending, proto=89
IP: s=155.1.0.5 (local), d=155.1.0.3 (Serial0/0/0), len 76, sending, proto=89
IP: s=155.1.0.5 (local), d=155.1.0.2 (Serial0/0/0), len 76, sending, proto=89
IP: s=155.1.0.5 (local), d=155.1.0.4 (Serial0/0/0), len 76, sending, proto=89
IP: s=155.1.0.1 (Serial0/0/0), d=155.1.0.5, len 76, rcvd 0, proto=89
IP: s=155.1.0.4 (Serial0/0/0), d=155.1.0.5, len 76, rcvd 0, proto=89
IP: s=155.1.0.3 (Serial0/0/0), d=155.1.0.5, len 104, rcvd 0, proto=89
IP: s=155.1.0.2 (Serial0/0/0), d=155.1.0.5, len 104, rcvd 0, proto=89
```

Copyright © 2010 Internetwork Expert

6.7 OSPF Network Loopback

- Advertise the Loopback 0 interfaces of R1, R2, R3, and R4 into OSPF area 0.
- Advertise the Loopback 0 interface of R6 into OSPF area 1.
- Advertise the Loopback 0 interfaces of SW1, and SW3 into OSPF area 2.
- Advertise the Loopback 0 interfaces of R5, SW2, and SW4 into OSPF area 3.
- Modify the network type of these interfaces so that the links are not advertised as host routes.

Configuration

```
R1:
interface Loopback0
 ip ospf 1 area 0
 ip ospf network point-to-point

R2:
interface Loopback0
 ip ospf 1 area 0
 ip ospf network point-to-point

R3:
interface Loopback0
 ip ospf 1 area 0
 ip ospf network point-to-point

R4:
interface Loopback0
 ip ospf 1 area 0
 ip ospf network point-to-point

R5:
interface Loopback0
 ip ospf 1 area 3
 ip ospf network point-to-point

R6:
interface Loopback0
 ip ospf 1 area 1
 ip ospf network point-to-point

SW1:
interface Loopback0
 ip ospf network point-to-point
!
router ospf 1
 network 150.1.7.7 0.0.0.0 area 2
```

Copyright © 2010 Internetwork Expert

```
SW2:
interface Loopback0
 ip ospf network point-to-point
!
router ospf 1
 network 150.1.8.8 0.0.0.0 area 3
```

Copyright © 2010 Internetwork Expert

```
SW3:
interface Loopback0
 ip ospf network point-to-point
!
router ospf 1
 network 150.1.9.9 0.0.0.0 area 2

SW4:
interface Loopback0
 ip ospf network point-to-point
!
router ospf 1
 network 150.1.10.10 0.0.0.0 area 3
```

Verification

> **Note**
>
> OSPF network type Loopback is a special case that is used for Loopback and looped back interfaces. Similar to point-to-multipoint an interface running network type Loopback is advertised as a stub endpoint instead of a transit subnet. The result of this network type that we see in our configuration is that when the Loopback 0 interfaces of these devices are advertised into OSPF they appear in the routing table as /32 host routes, instead of the actual subnet mask of /24.

```
Rack1R1#show ip route 150.1.10.0 255.255.255.0
% Subnet not in table

Rack1R1#show ip route 150.1.10.10
Routing entry for 150.1.10.10/32
  Known via "ospf 1", distance 110, metric 67, type inter area
  Last update from 155.1.0.5 on Serial0/0, 00:00:07 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.5.5, 00:00:07 ago, via Serial0/0
      Route metric is 67, traffic share count is 1

Rack1SW4#show ip ospf interface Loopback0
Loopback0 is up, line protocol is up
  Internet Address 150.1.10.10/24, Area 3
  Process ID 1, Router ID 223.255.255.255, Network Type LOOPBACK, Cost: 1
  Loopback interface is treated as a stub Host
```

If the network type of the interface is changed to point-to-point the host route advertisement is withdrawn and the interface is advertised with its native mask.

```
Rack1SW4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW4(config)#interface Loopback0
Rack1SW4(config-if)#ip ospf network point-to-point
Rack1SW4(config-if)#end
Rack1SW4#

Rack1R1#show ip route 155.1.10.10
Routing entry for 155.1.10.0/24
  Known via "ospf 1", distance 110, metric 67, type inter area
  Last update from 155.1.0.5 on Serial0/0, 00:01:39 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.5.5, 00:01:39 ago, via Serial0/0
      Route metric is 67, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

6.8 OSPF Path Selection with Auto-Cost

- Modify the global OSPF cost calculation of all devices so that a Ten Gigabit Ethernet interface has a cost of 3, and an OC-3 link has a cost of 193.

Configuration

```
R1 - R6, SW1 - SW4:
router ospf 1
 auto-cost reference-bandwidth 30000
```

Verification

Note

OSPF automatic cost calculation is an inverse function of the bandwidth of an interface, meaning the higher the bandwidth value of an interface, the lower the cost value. Specifically the formula for this calculation is:

Interface Cost = Reference Bandwidth / Interface Bandwidth

The issue with this calculation in current practical network designs is that the default reference bandwidth value used is 100Mbps. This means that a 100Mbps Fast Ethernet interface will have a cost of 1, and furthermore all interfaces with higher bandwidth values are also 1. The result of this is that OSPF routers in the topology cannot make an accurate path calculation when comparing an OC-48 POS interface versus a Gigabit Ethernet interface, as both interfaces revert to a cost of 1.

In order to resolve this, the reference bandwidth value is typically modified to allow these higher bandwidth interfaces to have separate, more granular cost values. In this particular case a Ten Gigabit Ethernet interface, or an OC-192 POS interface, are asked to have a cost value of 3, while a 155Mbps OC-3 POS/ATM interface is asked to have a cost value of 193.

Copyright © 2010 Internetwork Expert

For tasks like this it is not necessary to memorize the formula for cost calculation, but instead just to know how to derive it using various show outputs on the IOS CLI. Since the OSPF cost is derived from the bandwidth, changing the interface **bandwidth** value should change the cost. Without auto-cost modification, as seen below, R1 sees a 10Gbps interface with a cost of 1.

```
Rack1R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#interface FastEthernet0/0
Rack1R1(config-if)#bandwidth ?
  <1-10000000>  Bandwidth in kilobits
  inherit       Specify that bandwidth is inherited
  receive       Specify receive-side bandwidth

Rack1R1(config-if)#bandwidth 10000000
Rack1R1(config-if)#do show ip ospf interface FastEthernet0/0 | include Cost
  Process ID 1, Router ID 150.1.1.1, Network Type BROADCAST, Cost: 1
```

Changing the reference bandwidth to 100,000 has the following result:

```
Rack1R1(config-if)#router ospf 1
Rack1R1(config-router)#auto-cost reference-bandwidth 100000
Rack1R1(config-router)#do show ip ospf interface FastEthernet0/0 | include Cost
  Process ID 1, Router ID 150.1.1.1, Network Type BROADCAST, Cost: 10
```

Since the desired result is a cost of 3 for 10Gbps, the reference bandwidth 100,000 is too high. By process of elimination the correct value can be inferred.

```
Rack1R1(config-router)#auto-cost reference-bandwidth 50000
Rack1R1(config-router)#do show ip ospf interface FastEthernet0/0 | include Cost
  Process ID 1, Router ID 150.1.1.1, Network Type BROADCAST, Cost: 5
Rack1R1(config-router)#auto-cost reference-bandwidth 40000
Rack1R1(config-router)#do show ip ospf interface FastEthernet0/0 | include Cost
  Process ID 1, Router ID 150.1.1.1, Network Type BROADCAST, Cost: 4
Rack1R1(config-router)#auto-cost reference-bandwidth 30000
Rack1R1(config-router)#do show ip ospf interface FastEthernet0/0 | include Cost
  Process ID 1, Router ID 150.1.1.1, Network Type BROADCAST, Cost: 3
```

Copyright © 2010 Internetwork Expert

A reference bandwidth of 30,000 results in a cost of 3 (30000 reference bw/ 10000 interface bw = 3). This calculation then stays true for 155Mbps OC-3.

```
Rack1R1(config-router)#interface FastEthernet0/0
Rack1R1(config-if)#bandwidth 10000000
Rack1R1(config-if)#do show ip ospf interface FastEthernet0/0 | include Cost
  Process ID 1, Router ID 150.1.1.1, Network Type BROADCAST, Cost: 3
Rack1R1(config-if)#bandwidth 155000
Rack1R1(config-if)#do show ip ospf interface FastEthernet0/0 | include Cost
  Process ID 1, Router ID 150.1.1.1, Network Type BROADCAST, Cost: 193
```

Pitfall

Technically OSPF auto-cost reference-bandwidth does not need to match for neighbors to form an adjacency. However, if different neighbors throughout the topology calculate SPF based on conflicting cost calculations, loops can occur. For this reason, it is always recommended to set the auto-cost value consistently throughout the entire OSPF domain.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

6.9 OSPF Path Selection with Cost

- Using the interface level **ip ospf cost** command modify the OSPF domain so that traffic from SW4 going to the Loopback 0 network of R6 uses the point-to-point Serial link between R4 and R5.

Configuration

```
R5:
interface Serial0/1/0
 ip ospf cost 10000
```

Verification

Note

In order to understand how OSPF path selection can be modified for traffic engineering purposes, it is important to understand why the current path is being selected. OSPF is sometimes described as being partially link-state, and partially distance vector. This is due to the fact that SPF calculations are only performed for Intra-Area routing, while the Area Border Routers' (ABRs) advertised information is trusted for Inter-Area and External calculations.

In this particular scenario the goal is to modify the traffic flow so that packets coming from SW4 going to the network 150.1.6.0/24 use the point-to-point Serial link between R4 and R5. By visually mapping out the traffic flow we can see that SW4 has only one possible path to reach this network in the first place, the EtherChannel link in area 3 to SW2. SW2 likewise has only one path, the VLAN 58 link in area 3 to R5. R5 has multiple possible paths, the Frame Relay link to R1, the Frame Relay link to R4, and the point-to-point Serial link to R5. Since we are trying to affect the decision process of R5 this means that any cost changes in area 3 will not affect the traffic flow. Only calculation changes on or upstream from R5 can affect this path.

To reinforce this let's first find out how, and more importantly why, SW4 is choosing its current path. This can be determined by finding the longest match for R6's Loopback 0 interface via the **show ip route** output.

```
Rack1SW4#show ip route 150.1.6.6
Routing entry for 150.1.6.0/24
  Known via "ospf 1", distance 110, metric 19911, type inter area
  Last update from 155.1.108.8 on Port-channel1, 00:03:12 ago
  Routing Descriptor Blocks:
  * 155.1.108.8, from 150.1.5.5, 00:03:12 ago, via Port-channel1
      Route metric is 19911, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

SW4 says that the longest match to 150.1.6.6 is an OSPF Inter-Area route to 150.1.6.0/24 via the next-hop 155.1.108.8 (SW2). Note that after the next-hop value this output shows that the route is *from 150.1.5.5*. This means that R5 is the ABR originating the Network Summary LSA (LSA 3) from area 0 into area 3. By viewing the detailed information about this LSA 3 we can see what R5's cost to the destination is. Specifically, the syntax for this is **`show ip ospf database summary 150.1.6.0`**, where 150.1.6.0 is the link-state ID (the network without the mask) for the destination.

```
Rack1SW4#show ip ospf database summary 150.1.6.0

            OSPF Router with ID (223.255.255.255) (Process ID 1)

                Summary Net Link States (Area 3)

  Routing Bit Set on this LSA
  LS age: 212
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.6.0 (summary Network Number)
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000004
  Checksum: 0x7129
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 19731
```

The above output indicates that SW4 is using the ABR 150.1.5.5 as the exit point to reach 150.1.6.0/24. Note that the metric value of 19731 in this output is not SW4's metric to the destination, but the ABR's (R5's) metric to the destination. In order for SW4 to actually route to this destination it must now run SPF on the advertising router 150.1.5.5 (R5's RID).

This is why OSPF is considered partially distance vector and partially link-state. SW4 does not know the detailed information about the path selection that is occurring behind R5. SW4 trusts R5's calculation for the prefix, which is typically a distance vector behavior, and then calculates the shortest Intra-Area path to reach R5, which is a link-state behavior. The advantage of this design inherent to OSPF is that SW4 does not need to run SPF on the entire OSPF topology, which immensely increases scalability. Now let's find out how SW4 routes to get to R5.

Copyright © 2010 Internetwork Expert

OSPF's Intra-Area SPF determines this by finding out who SW4 is directly adjacent with, and what the cost values to these neighbors are. Once this is determined, SW4 asks these neighbors who they are adjacent with, and what their costs are to their adjacent neighbors. The process continues over and over until SW4 finds the shortest path to the ABR 150.1.5.5. This Intra-Area metric is then added to the LSA 3 metric that R5 is advertising, and results in the end-to-end metric value that is actually installed in the routing table. We can verify this by first finding out who SW4 is adjacent with by viewing its Router LSA (LSA 1). Specifically, this is accomplished with the **`show ip ospf database router 223.255.255.255`** syntax, where 223.255.255.255 is SW4's OSPF Router-ID.

```
Rack1SW4#show ip ospf database router 223.255.255.255

            OSPF Router with ID (223.255.255.255) (Process ID 1)

                Router Link States (Area 3)

  LS age: 699
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 223.255.255.255
  Advertising Router: 223.255.255.255
  LS Seq Number: 8000005E
  Checksum: 0xBA0E
  Length: 60
  Number of Links: 3

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 150.1.10.10
     (Link Data) Network Mask: 255.255.255.255
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.108.10
     (Link Data) Router Interface address: 155.1.108.10
      Number of TOS metrics: 0
       TOS 0 Metrics: 150

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 155.1.10.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 30
```

SW4 sees that it is adjacent with the *Designated Router address: 155.1.108.10*, and that the cost is 150. Although, technically the DR for this segment is SW4 itself, OSPF (like IS-IS) treats the DR as a separate "pseudonode" which it must consult first before computing SPF. This is because the EtherChannel link connecting to SW2 runs OSPF network type broadcast, which requires the DR/BDR election. Therefore, SW4 must ask the DR who it is adjacent with, and what its cost calculation to its adjacent neighbors is.

Copyright © 2010 Internetwork Expert

This information can be viewed by checking the DR's Network LSA (LSA 2) via the **show ip ospf database network 155.1.108.10**. From the below information we can see that SW4 finds out that the DR for the segment is adjacent with 150.1.8.8 (SW2's RID).

```
Rack1SW4#show ip ospf database network 155.1.108.10

            OSPF Router with ID (223.255.255.255) (Process ID 1)

                Net Link States (Area 3)

  Routing Bit Set on this LSA
  LS age: 641
  Options: (No TOS-capability, DC)
  LS Type: Network Links
  Link State ID: 155.1.108.10 (address of Designated Router)
  Advertising Router: 223.255.255.255
  LS Seq Number: 80000002
  Checksum: 0x3B83
  Length: 32
  Network Mask: /24
        Attached Router: 223.255.255.255
        Attached Router: 150.1.8.8
```

Based on this SW4 infers that it should now ask the router 150.1.8.8 who it is adjacent with. We can view this information on SW4 with the **show ip ospf database router 150.1.8.8** or on SW2 with the **show ip ospf database router 150.1.8.8 self-originate**.

```
Rack1SW4#show ip ospf database router 150.1.8.8

            OSPF Router with ID (223.255.255.255) (Process ID 1)

                Router Link States (Area 3)

  LS age: 428
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.8.8
  Advertising Router: 150.1.8.8
  LS Seq Number: 80000005
  Checksum: 0x7446
  Length: 72
  Number of Links: 4

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 150.1.8.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.108.10
     (Link Data) Router Interface address: 155.1.108.8
      Number of TOS metrics: 0
       TOS 0 Metrics: 150
```

Copyright © 2010 Internetwork Expert

```
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.58.8
     (Link Data) Router Interface address: 155.1.58.8
      Number of TOS metrics: 0
       TOS 0 Metrics: 30

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 155.1.8.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 30
```

The above output means that SW2 is adjacent with the DR 155.1.58.8 (SW2), and that the cost to the DR is 30. SW2 must now ask the DR who it is adjacent with.

```
Rack1SW4#show ip ospf database network 155.1.58.8

            OSPF Router with ID (223.255.255.255) (Process ID 1)

                Net Link States (Area 3)

  Routing Bit Set on this LSA
  LS age: 111
  Options: (No TOS-capability, DC)
  LS Type: Network Links
  Link State ID: 155.1.58.8 (address of Designated Router)
  Advertising Router: 150.1.8.8
  LS Seq Number: 80000001
  Checksum: 0xEA7F
  Length: 32
  Network Mask: /24
        Attached Router: 150.1.8.8
        Attached Router: 150.1.5.5
```

Since the DR is directly adjacent with 150.1.5.5, SW4 infers the total cost to 150.1.5.5 as 150 + 30. Since R5 is advertising the metric 19731 in its LSA 3 for 150.1.6.0, and SW4's metric to R5 is 180, we now know why the total metric in the routing table of SW4 is 19911. However, we do not yet know how R5's calculation to the final destination occurs. Since area 3 has no visibility on complete Inter-Area calculations, this can only be verified starting on R5.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip route 150.1.6.6
Routing entry for 150.1.6.0/24
  Known via "ospf 1", distance 110, metric 19731, type inter area
  Last update from 155.1.0.1 on Serial0/0/0, 00:06:48 ago
  Routing Descriptor Blocks:
    155.1.45.4, from 223.255.255.255, 00:06:48 ago, via Serial0/1/0
      Route metric is 19731, traffic share count is 1
  * 155.1.0.4, from 223.255.255.255, 00:06:48 ago, via Serial0/0/0
      Route metric is 19731, traffic share count is 1
    155.1.0.1, from 150.1.1.1, 00:06:48 ago, via Serial0/0/0
      Route metric is 19731, traffic share count is 1
```

R5 says that it has three possible paths to the destination 150.1.6.6 via the longest match 150.1.6.0/24. One of these comes from the area 0 router 150.1.1.1 via the next-hop 155.1.0.1 (R1 via the Frame Relay segment), and two of these come from the area 0 router 223.255.255.255 via the next-hops 155.1.45.4 and 155.1.0.4 (R4 via the point-to-point Serial and Frame Relay segments respectively). Since R5 sees this prefix as Inter-Area, it must ask the area 0 ABRs how they are calculating the path, and then run SPF on the ABRs themselves to find the ultimate shortest path. This can be verified on R5 by first viewing the possible paths to the Network Summary LSAs (LSA 3) that the ABRs are advertising, and then to the ABRs that are originating it via their Router LSAs (LSA 1).

```
Rack1R5#show ip ospf database summary 150.1.6.0

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Summary Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 524
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.6.0 (summary Network Number)
  Advertising Router: 150.1.1.1
  LS Seq Number: 80000001
  Checksum: 0x2D5
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 301

  Routing Bit Set on this LSA
  LS age: 524
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.6.0 (summary Network Number)
  Advertising Router: 223.255.255.255
  LS Seq Number: 80000001
  Checksum: 0x830E
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 301
```

Copyright © 2010 Internetwork Expert

```
                Summary Net Link States (Area 3)

  LS age: 523
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.6.0 (summary Network Number)
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000004
  Checksum: 0x7129
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 19731
```

R5 sees three possible paths to the destination in the OSPF database. The first is in area 0 via the ABR 150.1.1.1 (R1); the second is in area 0 via the ABR 223.255.255.255 (R4), and the third in area 3 via the ABR 150.1.5.5 (R5). Since the area 3 LSA 3 is originated by R5 itself, this one is immediately eliminated. Now R5 must find the shortest path to the two ABRs 150.1.1.1 and 223.255.255.255 in area 0. This is accomplished the same way it was in area 3, by finding out the cost to R5's adjacent neighbors, and then recursing over and over to find their cost and their adjacent neighbors' cost towards the ABRs. In this case the topology is fairly simple because R5 is directly adjacent with both of these area 0 routers.

```
Rack1R5#show ip ospf database router 150.1.5.5 self-originate

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Router Link States (Area 0)

  LS age: 769
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.5.5
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000052
  Checksum: 0xDCE9
  Length: 108
  Area Border Router
  Number of Links: 7

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.1.1
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 19430

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.3.3
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 19430

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.2.2
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 19430
```

Copyright © 2010 Internetwork Expert

```
    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 223.255.255.255
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 19430

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 155.1.0.5
     (Link Data) Network Mask: 255.255.255.255
      Number of TOS metrics: 0
       TOS 0 Metrics: 0

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 223.255.255.255
     (Link Data) Router Interface address: 155.1.45.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 19430

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 155.1.45.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 19430
<output omitted>
```

Two points should be noted about the above output. First, the area 3 output that R5 would normally have originated is omitted, because this does not affect its path selection out towards area 1. Secondly, the output of the three area 0 learned routes indicates that the information is from a *Link connected to: another Router (point-to-point)*. This is due to the fact that the OSPF network type point-to-multipoint non-broadcast, which is running over the Frame Relay link, and the network type point-to-point, which is running over the direct link to R4, does not support the DR/BDR election. Recall that this output in area 3 was different because SW4 and SW2 could only learn LSA 1 information via the DR's pseudonode from LSA 2. Network type's point-to-multipoint, point-to-multipoint non-broadcast, and point-to-point do not support the DR/BDR election, and therefore do not require the Network LSA (LSA 2) to recurse towards the adjacent neighbor.

At this point we can now see why R5 is installing three paths to the same destination. R1 is originating the LSA 3 with a metric of 301, as is R4. R5 has a metric of 19430 to R1 and R4 via the Frame Relay, and to R4 via the point-to-point link. Since the end-to-end metric is the same all three routes are installed.

Copyright © 2010 Internetwork Expert

Since the goal of the section is to route only via the point-to-point link to R4, we can now see where changes can and cannot be made. R1's metric to R6 could be raised, which would raise its LSA 3 advertisement to R5. This in turn would cause R5 to choose only R4 and not R1. However, this still results in R5 routing towards both links to R4. A lowering of R4's metric to R6 will make it lower the LSA 3 advertisement to R5, and in turn make it more preferred over R1. Again however this still results in R5 routing towards both links to R4, because it affects the LSA 3 sent out both of R4's links. Therefore, the only way to force the path to only the point-to-point link is either to lower the cost of the point-to-point link on R5, or to raise the cost of the Frame Relay. In this solution the point-to-point link's cost is lowered from 19430 to 10000. The result of this can be seen in R5's LSA 1 advertisement.

```
Rack1R5#show ip ospf database router 150.1.5.5 self-originate

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Router Link States (Area 0)

  LS age: 75
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.5.5
  Advertising Router: 150.1.5.5
  LS Seq Number: 8000000D
  Checksum: 0xCA37
  Length: 108
  Area Border Router
  Number of Links: 7

<output omitted>

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 223.255.255.255
     (Link Data) Router Interface address: 155.1.45.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 10000

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 155.1.45.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 10000
```

Copyright © 2010 Internetwork Expert

Final verification of the change in traffic flow can be seen in R5's routing table and in a traceroute from SW4.

```
Rack1R5#show ip route 150.1.6.6
Routing entry for 150.1.6.0/24
  Known via "ospf 1", distance 110, metric 10301, type inter area
  Last update from 155.1.45.4 on Serial0/1/0, 00:03:22 ago
  Routing Descriptor Blocks:
  * 155.1.45.4, from 223.255.255.255, 00:03:22 ago, via Serial0/1/0
      Route metric is 10301, traffic share count is 1

Rack1SW4#traceroute 150.1.6.6

Type escape sequence to abort.
Tracing the route to 150.1.6.6

  1 155.1.108.8 0 msec 4 msec 0 msec
  2 155.1.58.5 0 msec 4 msec 0 msec
  3 155.1.45.4 16 msec 16 msec 12 msec
  4 155.1.146.6 24 msec *  24 msec
```

Copyright © 2010 Internetwork Expert

6.10 OSPF Path Selection with Bandwidth

- Using the interface level **bandwidth** keyword modify the OSPF domain so that traffic from R6 to the Loopback 0 network of SW2 is first sent to R1.

Configuration

```
R1:
interface Serial0/0
 bandwidth 10000
```

Verification

> **Note**
>
> Like the previous section it is first important to understand why the current path selection occurs before modifying it. In this case R6 has two routes to the destination 150.1.8.8 via the longest match 150.1.8.0/24 from R1 and R4.

```
Rack1R6#show ip route 150.1.8.8
Routing entry for 150.1.8.0/24
  Known via "ospf 1", distance 110, metric 20031, type inter area
  Last update from 155.1.146.4 on FastEthernet0/0.146, 00:27:54 ago
  Routing Descriptor Blocks:
    155.1.146.4, from 223.255.255.255, 00:27:54 ago, via FastEthernet0/0.146
      Route metric is 20031, traffic share count is 1
  * 155.1.146.1, from 150.1.1.1, 00:27:54 ago, via FastEthernet0/0.146
      Route metric is 20031, traffic share count is 1
```

The metric 20031 through both R1 and R4 implies that the addition of R1 and R4's LSA 3 advertisement for 150.1.8.0 plus R6's metric to the DR for VLAN 146 is the same. This can be verified by viewing the ABRs' LSA 3 advertisements, and R6's LSA 1 advertisement.

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip ospf database summary 150.1.8.0

            OSPF Router with ID (150.1.6.6) (Process ID 1)

                Summary Net Link States (Area 1)

  Routing Bit Set on this LSA
  LS age: 395
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.8.0 (summary Network Number)
  Advertising Router: 150.1.1.1
  LS Seq Number: 80000002
  Checksum: 0x930F
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 19731

  Routing Bit Set on this LSA
  LS age: 379
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.8.0 (summary Network Number)
  Advertising Router: 223.255.255.255
  LS Seq Number: 80000002
  Checksum: 0x1547
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 19731


                Summary Net Link States (Area 2)

  Routing Bit Set on this LSA
  LS age: 380
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.8.0 (summary Network Number)
  Advertising Router: 150.1.3.3
  LS Seq Number: 80000002
  Checksum: 0xCD03
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 65836
```

R6 sees the Inter-Area LSA 3 route via three ABRs, to R1 with a metric of 19731, to R4 with a metric of 19731, and to R3 with the metric 65836. R6 must then recurse to LSA 1 to find the shortest Intra-Area paths to these ABRs.

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip ospf database router 150.1.6.6 self-originate

            OSPF Router with ID (150.1.6.6) (Process ID 1)

                Router Link States (Area 1)

  LS age: 627
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.6.6
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000008
  Checksum: 0x415E
  Length: 48
  Number of Links: 2

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 150.1.6.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.146.6
     (Link Data) Router Interface address: 155.1.146.6
      Number of TOS metrics: 0
       TOS 0 Metrics: 300



                Router Link States (Area 2)

  LS age: 1147
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.6.6
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000005
  Checksum: 0x935B
  Length: 36
  Number of Links: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.67.7
     (Link Data) Router Interface address: 155.1.67.6
      Number of TOS metrics: 0
       TOS 0 Metrics: 300
```

Copyright © 2010 Internetwork Expert

R6 says that the metric to reach the DR 155.1.146.6 for VLAN 146 is 300. Since we know from the topology diagram that the only neighbors on VLAN 146 are R1, R4, and R6, this implies the metric to reach the ABRs R1 and R4 is 300. Since R1 and R4 are advertising the LSA 3 with a metric of 19731, and R3 is advertising it with a metric of 65836, we can assume at this point that traffic from R6 to the destination will transit area 1 and not area 2, and we can exclude R3 from the problem at hand. Next we come to the decision of where cost manipulations can and cannot occur.

By changing R6's metric to the DR of VLAN 146, it would affect the metric to both ABRs R1 and R4. While this will change the overall end-to-end cost, it will not change R6's path selection. By changing R5's area 3 path to the destination this would affect the LSA 3 advertisement R5 sends into area 0, but would still result in the same LSA 3 being received by R6 via R1 and R4, just with a different metric than the default. By raising R4's metric of the LSA 3 sent into area 1, or by lowering R1's metric of the LSA 3 sent into area 1, this would cause R6 to route to R1. In this particular solution R1's metric is lowered in the LSA 3.

Since the OSPF cost value is derived from the bandwidth, changing the **`bandwidth`** keyword at the interface level changes the cost.

```
Rack1R1#show interface Serial0/0 | include BW
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,

Rack1R1#show ip ospf interface Serial0/0 | include Cost
  Process ID 1, Router ID 150.1.1.1, Network Type POINT_TO_MULTIPOINT, Cost:
19430

Rack1R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#interface Serial0/0
Rack1R1(config-if)# bandwidth 10000
Rack1R1(config-if)#end

Rack1R1#show interface Serial0/0 | include BW
  MTU 1500 bytes, BW 10000 Kbit, DLY 20000 usec,

Rack1R1#show ip ospf interface Serial0/0 | include Cost
  Process ID 1, Router ID 150.1.1.1, Network Type POINT_TO_MULTIPOINT, Cost:
3000
```

Copyright © 2010 Internetwork Expert

Based on this change R6 now sees different values in the LSA 3 summary reported by R1 and R4.

```
Rack1R6#show ip ospf database summary 150.1.8.0

            OSPF Router with ID (150.1.6.6) (Process ID 1)

                Summary Net Link States (Area 1)

  Routing Bit Set on this LSA
  LS age: 95
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.8.0 (summary Network Number)
  Advertising Router: 150.1.1.1
  LS Seq Number: 80000005
  Checksum: 0x7E8F
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 3301

  Routing Bit Set on this LSA
  LS age: 772
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.8.0 (summary Network Number)
  Advertising Router: 223.255.255.255
  LS Seq Number: 80000004
  Checksum: 0x1149
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 19731


                Summary Net Link States (Area 2)

  LS age: 762
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.8.0 (summary Network Number)
  Advertising Router: 150.1.3.3
  LS Seq Number: 80000004
  Checksum: 0xC905
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 65836
```

Copyright © 2010 Internetwork Expert

Since R6's cost of 300 to the VLAN 146 DR did not change R6 now has a metric of 300 + 3301 if it routes to R1, or 300 + 19731 if it routes to R4. The end result is that the path selection is changed to route only to R1 with a metric of 3601.

```
Rack1R6#show ip route 150.1.8.8
Routing entry for 150.1.8.0/24
  Known via "ospf 1", distance 110, metric 3601, type inter area
  Last update from 155.1.146.1 on FastEthernet0/0.146, 00:05:18 ago
  Routing Descriptor Blocks:
  * 155.1.146.1, from 150.1.1.1, 00:05:18 ago, via FastEthernet0/0.146
      Route metric is 3601, traffic share count is 1
```

Final verification of this can be seen via a traceroute from R6.

```
Rack1R6#traceroute 150.1.8.8

Type escape sequence to abort.
Tracing the route to 150.1.8.8

  1 155.1.146.1 4 msec 4 msec 0 msec
  2 155.1.0.5 24 msec 20 msec 20 msec
  3 155.1.58.8 20 msec *  20 msec
```

Copyright © 2010 Internetwork Expert

6.11 OSPF Path Selection with Per-Neighbor Cost

- The Frame Relay PVC between R1 & R5 has been provisioned with 512Kbps of bandwidth by the service provider, while the PVC between R4 & R5 has been provisioned at 768Kbps. Configure per-neighbor cost values on R5 to reflect the underlying bandwidth of the Frame Relay circuits connecting it to R1 and R4.

Configuration

```
R5:
router ospf 1
 neighbor 155.1.0.1 cost 58593
 neighbor 155.1.0.4 cost 39062
```

Verification

Note

OSPF cost calculation for a segment is based on the bandwidth value of the outgoing interface. In certain network topologies this can be an issue if the underlying layer 2 network does not map directly to the bandwidth value of the interface connecting to it.

For example in this topology R5's single Serial interface connecting to the Frame Relay cloud has multiple underlying layer 2 PVCs. Although the physical Serial interface has a bandwidth value of T1, which based on the WIC card being used, the PVCs themselves are not provisioned at this rate. Normally, the easiest way to fix this is to just change the bandwidth value to match the underlying circuit speed. However, the interface does not support setting the bandwidth on a per circuit basis, only on a per interface basis. Therefore, even though the circuit to R1 is slower than the circuit to R4, there is no way for OSPF to tell the difference based on the interface itself. In order to resolve this situation the network type's point-to-multipoint and point-to-multipoint non-broadcast support the setting of the OSPF cost value on a per-neighbor basis.

Copyright © 2010 Internetwork Expert

To solve this task we must first determine what the cost values associated with bandwidths of 512Kbps and 768Kbps are. One way is to manually compute this based on the auto-cost formula. The other way is to use the command line output to determine it.

```
Rack1R5#show interface Serial0/0/0 | include BW
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
Rack1R5#show ip ospf interface Serial0/0/0 | include Cost
  Process ID 1, Router ID 150.1.5.5, Network Type POINT_TO_MULTIPOINT, Cost:
19430
```

From the above output we can see that R5's T1 Serial interface has a cost of 19430.

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#bandwidth 512
Rack1R5(config-if)#do show ip ospf interface Serial0/0/0 | include Cost
  Process ID 1, Router ID 150.1.5.5, Network Type POINT_TO_MULTIPOINT, Cost:
58593
```

When the bandwidth is changed to 512Kbps, the cost is calculated as 58593.

```
Rack1R5(config-if)#bandwidth 768
Rack1R5(config-if)#do show ip ospf interface Serial0/0/0 | include Cost
  Process ID 1, Router ID 150.1.5.5, Network Type POINT_TO_MULTIPOINT, Cost:
39062
```

When the bandwidth is changed to 768Kbps, the cost is calculated as 39062. These values can now be configured under the OSPF process as the per-neighbor costs.

```
Rack1R5(config-if)#no bandwidth
Rack1R5(config-if)#router ospf 1
Rack1R5(config-router)#neighbor 155.1.0.1 cost 58593
Rack1R5(config-router)#neighbor 155.1.0.4 cost 39062
Rack1R5(config-router)#end
Rack1R5#
```

Copyright © 2010 Internetwork Expert

Once complete we can see that the interface cost value stays the same as the original default, however R5's router LSA (LSA 1) is changed to reflect the cost values defined on a per-neighbor basis.

```
Rack1R5#show ip ospf interface Serial0/0/0 | include Cost
  Process ID 1, Router ID 150.1.5.5, Network Type POINT_TO_MULTIPOINT, Cost:
19430

Rack1R5#show ip ospf database router 150.1.5.5 self-originate

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Router Link States (Area 0)

  LS age: 19
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.5.5
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000011
  Checksum: 0xA2C9
  Length: 108
  Area Border Router
  Number of Links: 7

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.3.3
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 19430

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.1.1
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 58593

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.2.2
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 19430

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 223.255.255.255
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 39062

<output omitted>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

6.12 Repairing Discontiguous OSPF Areas with Virtual-Links

- Configure the network so that full reachability is maintained if R3's connection to SW1 goes down.
- Do not apply any configuration onto R4 to solve this task.

Configuration

```
R1:
router ospf 1
 area 1 virtual-link 150.1.6.6

R6:
router ospf 1
 area 1 virtual-link 150.1.1.1
```

Verification

> **Note**
>
> Devices in OSPF area 2 have two exit points out the rest of the network, R3 via area 0, and R6 via area 1. However, although R6 has interfaces in multiple areas, it is technically not an ABR. This can be verified in the OSPF database of devices area 2, as only R3 can originate Summary Network LSAs (LSA 3) to describe Inter-Area destinations.

```
Rack1SW3#show ip ospf database

            OSPF Router with ID (150.1.9.9) (Process ID 1)

                Router Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum Link count
150.1.3.3       150.1.3.3       1894        0x80000010 0x00FD34 1
150.1.6.6       150.1.6.6       1960        0x8000000F 0x007F65 1
150.1.7.7       150.1.7.7       1038        0x80000012 0x000C7F 5
150.1.9.9       150.1.9.9       1878        0x8000000F 0x0046FE 3

                Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.37.3      150.1.3.3       109         0x8000000E 0x000384
155.1.67.7      150.1.7.7       29          0x8000000E 0x008CCA
155.1.79.9      150.1.9.9       136         0x8000000E 0x001B24

                Summary Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
150.1.1.0       150.1.3.3       1118        0x8000000D 0x002112
150.1.2.0       150.1.3.3       357         0x80000011 0x001E24
150.1.3.0       150.1.3.3       1378        0x8000000D 0x002EC9
150.1.4.0       150.1.3.3       1898        0x8000000C 0x00249C
150.1.5.0       150.1.3.3       1121        0x8000000C 0x0018DE
150.1.6.0       150.1.3.3       1898        0x8000000C 0x00D0C0
```

Copyright © 2010 Internetwork Expert

```
150.1.8.0       150.1.3.3       361         0x80000013 0x001BCC
150.1.10.0      150.1.3.3       361         0x80000013 0x00E668
155.1.0.1       150.1.3.3       1121        0x8000000D 0x00D658
155.1.0.2       150.1.3.3       361         0x80000011 0x00D469
155.1.0.3       150.1.3.3       1382        0x8000000D 0x00E50E
155.1.0.4       150.1.3.3       1898        0x8000000C 0x00DCDF
155.1.0.5       150.1.3.3       1122        0x8000000C 0x00D31F
155.1.5.0       150.1.3.3       361         0x80000011 0x00B803
155.1.8.0       150.1.3.3       361         0x80000013 0x00FCC8
155.1.10.0      150.1.3.3       361         0x80000013 0x00C864
155.1.13.0      150.1.3.3       1643        0x8000000E 0x007276
155.1.23.0      150.1.3.3       1643        0x8000000E 0x0004DA
155.1.45.0      150.1.3.3       1899        0x8000000C 0x00147F
155.1.58.0      150.1.3.3       362         0x80000013 0x00A70A
155.1.108.0     150.1.3.3       362         0x80000013 0x006187
155.1.146.0     150.1.3.3       1899        0x8000000C 0x007B85
192.10.1.0      150.1.3.3       362         0x80000011 0x005192

                Summary ASB Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
192.10.1.254    150.1.3.3       362         0x80000011 0x004D96

                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
51.51.51.51     192.10.1.254    1302        0x80000039 0x002CA4 0
```

Based on this output we can infer that if area 2 loses its connection to R3, all Inter-Area connectivity to area 2 will be lost. This can be demonstrated by shutting down R3's connection to SW1.

```
Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#interface FastEthernet0/0
Rack1R3(config-if)#shutdown
Rack1R3(config-if)#end
Rack1R3#
```

Once the link to R3 is down SW1 issues a withdraw message, but the devices in area 2 cache the LSA information from R3 in the OSPF database until the LSA age expires.

Copyright © 2010 Internetwork Expert

```
Rack1SW3#show ip ospf database

            OSPF Router with ID (150.1.9.9) (Process ID 1)

                Router Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum Link count
150.1.3.3       150.1.3.3       351         0x80000011 0x00FB35 1
150.1.6.6       150.1.6.6       75          0x80000011 0x007172 1
150.1.7.7       150.1.7.7       54          0x80000016 0x00FC57 4
150.1.9.9       150.1.9.9       54          0x80000002 0x003420 3

                Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.37.3      150.1.3.3       592         0x8000000E 0x000384
155.1.67.6      150.1.6.6       76          0x80000001 0x00C5A1
155.1.79.7      150.1.7.7       56          0x80000001 0x0073DE

                Summary Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
150.1.1.0       150.1.3.3       1600        0x8000000D 0x002112
150.1.2.0       150.1.3.3       840         0x80000011 0x001E24
150.1.3.0       150.1.3.3       1861        0x8000000D 0x002EC9
150.1.4.0       150.1.3.3       352         0x8000000D 0x00229D
150.1.5.0       150.1.3.3       1601        0x8000000C 0x0018DE
150.1.6.0       150.1.3.3       352         0x8000000D 0x00CEC1
150.1.8.0       150.1.3.3       841         0x80000013 0x001BCC
150.1.10.0      150.1.3.3       841         0x80000013 0x00E668
155.1.0.1       150.1.3.3       1602        0x8000000D 0x00D658
155.1.0.2       150.1.3.3       841         0x80000011 0x00D469
155.1.0.3       150.1.3.3       1862        0x8000000D 0x00E50E
155.1.0.4       150.1.3.3       353         0x8000000D 0x00DAE0
155.1.0.5       150.1.3.3       1603        0x8000000C 0x00D31F
155.1.5.0       150.1.3.3       842         0x80000011 0x00B803
155.1.8.0       150.1.3.3       842         0x80000013 0x00FCC8
155.1.10.0      150.1.3.3       842         0x80000013 0x00C864
155.1.13.0      150.1.3.3       2124        0x8000000E 0x007276
155.1.23.0      150.1.3.3       2124        0x8000000E 0x0004DA
155.1.45.0      150.1.3.3       353         0x8000000D 0x001280
155.1.58.0      150.1.3.3       842         0x80000013 0x00A70A
155.1.108.0     150.1.3.3       842         0x80000013 0x006187
155.1.146.0     150.1.3.3       353         0x8000000D 0x007986
192.10.1.0      150.1.3.3       842         0x80000011 0x005192

                Summary ASB Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
192.10.1.254    150.1.3.3       842         0x80000011 0x004D96

                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
51.51.51.51     192.10.1.254    1781        0x80000039 0x002CA4 0
```

Copyright © 2010 Internetwork Expert

Although the LSA 3's still show up as reachable via R3, the recursion process towards R3 fails because the LSA 1 describing R3 is unreachable.

```
Rack1SW3#show ip ospf database router 150.1.3.3

            OSPF Router with ID (150.1.9.9) (Process ID 1)

                Router Link States (Area 2)

  Adv Router is not-reachable
  LS age: 363
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.3.3
  Advertising Router: 150.1.3.3
  LS Seq Number: 80000011
  Checksum: 0xFB35
  Length: 36
  Area Border Router
  Number of Links: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.37.3
     (Link Data) Router Interface address: 155.1.37.3
      Number of TOS metrics: 0
       TOS 0 Metrics: 300
```

The end result of this is that devices in area 2 cannot reach devices in other areas.

```
Rack1SW3#show ip route ospf
     155.1.0.0/24 is subnetted, 4 subnets
O       155.1.7.0 [110/60] via 155.1.79.7, 00:00:55, Vlan79
O       155.1.67.0 [110/60] via 155.1.79.7, 00:00:55, Vlan79
     150.1.0.0/24 is subnetted, 2 subnets
O       150.1.7.0 [110/31] via 155.1.79.7, 00:00:55, Vlan79
```

To resolve this problem R6 must offer devices in area 2 an alternate path to area 0. This can be done with a new link being added to R6 that is in area 0, such as another Ethernet interface or a Tunnel interface, or by configuring an OSPF virtual-link.

An OSPF virtual-link allows the creation of an indirect area 0 adjacency. This adjacency can be used to repair breaks in the OSPF domain or to solve traffic engineering requirements. In this particular case R6 can virtual-link to the neighbors R1 or R4 who are in area 1, since they both have connections to area 0. R1 is chosen in this solution since the question states not to configure R4.

Copyright © 2010 Internetwork Expert

The first important point to note about the virtual-link is that the neighbor value specified in the virtual-link syntax is the router-id of the neighbor in the transit area. This means that if for some reason the router-id changes, i.e. a new higher Loopback interface is added, or the **`router-id`** command is changed, the virtual-link will fail.

Additionally, the neighbors forming adjacency over the virtual-link do not have to be directly connected; they simply need to know how to recurse towards each others' LSA 1 advertisements. This means that the traffic flow via the virtual-link should naturally follow the Intra-Area SPF calculation between the routers' LSA 1 advertisements. Verification of this can be seen as follows.

```
Rack1R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#router ospf 1
Rack1R1(config-router)#area 1 virtual-link 150.1.6.6
Rack1R1(config-router)#end
Rack1R1#

Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#router ospf 1
Rack1R6(config-router)#area 1 virtual-link 150.1.1.1
Rack1R6(config-router)#end
Rack1R6#
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.1.1 on OSPF_VL0 from LOADING to FULL,
Loading Done

Rack1R6#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address     Interface
150.1.1.1         0   FULL/  -          -         155.1.146.1 OSPF_VL0
150.1.1.1         1   FULL/BDR        00:00:39    155.1.146.1 FastEthernet0/0.146
223.255.255.255   0   FULL/DROTHER    00:00:37    155.1.146.4 FastEthernet0/0.146
150.1.7.7         1   FULL/BDR        00:00:39    155.1.67.7  FastEthernet0/0.67
```

Copyright © 2010 Internetwork Expert

R1 and R6 form an adjacency over the virtual-link. **Show ip ospf interface** indicates that the virtual-link is an area 0 interface.

```
Rack1R6#show ip ospf interface
OSPF_VL0 is up, line protocol is up
  Internet Address 155.1.146.6/24, Area 0
  Process ID 1, Router ID 150.1.6.6, Network Type VIRTUAL_LINK, Cost: 300
  Configured as demand circuit.
  Run as demand circuit.
  DoNotAge LSA allowed.
  Transmit Delay is 1 sec, State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:01
  Supports Link-local Signaling (LLS)
  Index 1/4, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.1.1  (Hello suppressed)
  Suppress hello for 1 neighbor(s)

The cost value of the virtual-link is based on R6's cost to reach the router
150.1.1.1 in area 1.  This can be calculated via LSA 1 recursion.

Rack1R6#show ip ospf database router 150.1.6.6 self-originate

            OSPF Router with ID (150.1.6.6) (Process ID 1)

                Router Link States (Area 0)

  LS age: 3
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.6.6
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000002
  Checksum: 0xEDE
  Length: 36
  Area Border Router
  Number of Links: 1

    Link connected to: a Virtual Link
     (Link ID) Neighboring Router ID: 150.1.1.1
     (Link Data) Router Interface address: 155.1.146.6
      Number of TOS metrics: 0
       TOS 0 Metrics: 300

                Router Link States (Area 1)

  LS age: 3
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.6.6
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000017
  Checksum: 0x3259
  Length: 48
  Area Border Router
  Virtual Link Endpoint
```

Copyright © 2010 Internetwork Expert

```
  Number of Links: 2

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 150.1.6.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.146.6
     (Link Data) Router Interface address: 155.1.146.6
      Number of TOS metrics: 0
       TOS 0 Metrics: 300


                Router Link States (Area 2)

  LS age: 13
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.6.6
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000012
  Checksum: 0x726F
  Length: 36
  Area Border Router
  Number of Links: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.67.6
     (Link Data) Router Interface address: 155.1.67.6
      Number of TOS metrics: 0
       TOS 0 Metrics: 300
```

R6 now generates an LSA 1 for area 0, along with its previous LSA 1's in areas 1 and 2. Inside the area 0 LSA 1 it says that it is adjacent to R1 over a virtual-link. Inside the area 1 LSA 1 it says that it is the endpoint for the virtual-link, and is adjacent with the DR 155.1.146.6 (itself) with a cost of 300. Asking the DR who it is adjacent with via LSA 2 it implies that R6 is on the same segment as R1, with a cost of 300. This metric of 300 is what the virtual-link inherits. Now since R6 is a true ABR, it should be generating LSA 3 into area 2 about area 0 (and the other areas), and should be generating LSA 3 into area 0 about area 2.

```
Rack1SW3#show ip ospf database

            OSPF Router with ID (150.1.9.9) (Process ID 1)

                Router Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum Link count
150.1.3.3       150.1.3.3       2257        0x80000011 0x00FB35 1
150.1.6.6       150.1.6.6       481         0x80000012 0x00726F 1
150.1.7.7       150.1.7.7       131         0x80000017 0x00FA58 4
150.1.9.9       150.1.9.9       26          0x80000003 0x003221 3
```

```
                Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.37.3      150.1.3.3       2498        0x8000000E 0x000384
155.1.67.6      150.1.6.6       34          0x80000002 0x00C3A2
155.1.79.7      150.1.7.7       132         0x80000002 0x0071DF

                Summary Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
150.1.1.0       150.1.3.3       3507        0x8000000D 0x002112
150.1.1.0       150.1.6.6       467         0x80000001 0x00F7DA
150.1.2.0       150.1.3.3       2747        0x80000011 0x001E24
150.1.2.0       150.1.6.6       468         0x80000001 0x0075E5
150.1.3.0       150.1.6.6       468         0x80000001 0x006AEF
150.1.4.0       150.1.3.3       2258        0x8000000D 0x00229D
150.1.4.0       150.1.6.6       463         0x80000002 0x00D4F9
150.1.5.0       150.1.3.3       3507        0x8000000C 0x0018DE
150.1.5.0       150.1.6.6       468         0x80000001 0x0066A4
150.1.6.0       150.1.3.3       2258        0x8000000D 0x00CEC1
150.1.6.0       150.1.6.6       483         0x80000001 0x00FDFC
150.1.8.0       150.1.3.3       2748        0x80000013 0x001BCC
150.1.8.0       150.1.6.6       468         0x80000001 0x00778B
150.1.10.0      150.1.3.3       2748        0x80000013 0x00E668
150.1.10.0      150.1.6.6       468         0x80000001 0x004228
155.1.0.1       150.1.3.3       3509        0x8000000D 0x00D658
155.1.0.1       150.1.6.6       468         0x80000001 0x00AD21
155.1.0.2       150.1.3.3       2748        0x80000011 0x00D469
155.1.0.2       150.1.6.6       468         0x80000001 0x002C2B
155.1.0.3       150.1.6.6       469         0x80000001 0x002234
155.1.0.4       150.1.3.3       2259        0x8000000D 0x00DAE0
155.1.0.4       150.1.6.6       464         0x80000002 0x008D3D
155.1.0.5       150.1.3.3       3509        0x8000000C 0x00D31F
155.1.0.5       150.1.6.6       469         0x80000001 0x0020E6
155.1.5.0       150.1.3.3       2748        0x80000011 0x00B803
155.1.5.0       150.1.6.6       469         0x80000001 0x0010C4
155.1.8.0       150.1.3.3       2748        0x80000013 0x00FCC8
155.1.8.0       150.1.6.6       470         0x80000001 0x005888
155.1.10.0      150.1.3.3       2749        0x80000013 0x00C864
155.1.10.0      150.1.6.6       470         0x80000001 0x002424
155.1.13.0      150.1.6.6       470         0x80000001 0x00D1BE
155.1.23.0      150.1.6.6       470         0x80000001 0x00EB24
155.1.45.0      150.1.3.3       2260        0x8000000D 0x001280
155.1.45.0      150.1.6.6       470         0x80000001 0x006245
155.1.58.0      150.1.3.3       2749        0x80000013 0x00A70A
155.1.58.0      150.1.6.6       470         0x80000001 0x0004C8
155.1.108.0     150.1.3.3       2750        0x80000013 0x006187
155.1.108.0     150.1.6.6       470         0x80000001 0x00BC47
155.1.146.0     150.1.3.3       2261        0x8000000D 0x007986
155.1.146.0     150.1.6.6       485         0x80000001 0x006BD1
192.10.1.0      150.1.3.3       2750        0x80000011 0x005192
192.10.1.0      150.1.6.6       470         0x80000001 0x00A854

                Summary ASB Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
192.10.1.254    150.1.3.3       2750        0x80000011 0x004D96
192.10.1.254    150.1.6.6       471         0x80000001 0x00A458
```

Copyright © 2010 Internetwork Expert

```
                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
51.51.51.51     192.10.1.254    1685        0x8000003A 0x002AA5 0
```

Note that in the above output LSA 3's appear in area 2 from both R3 and R6, even though R3 is still unreachable. This is due to the fact that R3's LSAs are still aging out, even though the router R3 itself is still unreachable. The end result of the virtual-link is that traffic from area 2 to area 0, along with the other areas, now transits area 1 via the virtual-link.

```
Rack1SW3#show ip route ospf
     51.0.0.0/32 is subnetted, 1 subnets
O E2    51.51.51.51 [110/20] via 155.1.79.7, 00:07:51, Vlan79
     155.1.0.0/16 is variably subnetted, 18 subnets, 2 masks
O IA    155.1.146.0/24 [110/360] via 155.1.79.7, 00:08:01, Vlan79
O IA    155.1.23.0/24 [110/26028] via 155.1.79.7, 00:07:52, Vlan79
O IA    155.1.10.0/24 [110/3545] via 155.1.79.7, 00:07:52, Vlan79
O IA    155.1.8.0/24 [110/3395] via 155.1.79.7, 00:07:52, Vlan79
O IA    155.1.13.0/24 [110/19790] via 155.1.79.7, 00:07:52, Vlan79
O IA    155.1.0.2/32 [110/6598] via 155.1.79.7, 00:07:52, Vlan79
O IA    155.1.0.3/32 [110/6598] via 155.1.79.7, 00:07:52, Vlan79
O IA    155.1.0.1/32 [110/360] via 155.1.79.7, 00:07:52, Vlan79
O       155.1.7.0/24 [110/60] via 155.1.79.7, 00:08:02, Vlan79
O IA    155.1.0.4/32 [110/360] via 155.1.79.7, 00:07:47, Vlan79
O IA    155.1.5.0/24 [110/3410] via 155.1.79.7, 00:07:52, Vlan79
O IA    155.1.0.5/32 [110/3360] via 155.1.79.7, 00:07:52, Vlan79
O IA    155.1.58.0/24 [110/3365] via 155.1.79.7, 00:07:52, Vlan79
O IA    155.1.45.0/24 [110/13360] via 155.1.79.7, 00:07:53, Vlan79
O       155.1.67.0/24 [110/60] via 155.1.79.7, 00:08:02, Vlan79
O IA    155.1.108.0/24 [110/3515] via 155.1.79.7, 00:07:53, Vlan79
O IA 192.10.1.0/24 [110/6898] via 155.1.79.7, 00:07:53, Vlan79
     150.1.0.0/24 is subnetted, 10 subnets
O       150.1.7.0 [110/31] via 155.1.79.7, 00:08:02, Vlan79
O IA    150.1.6.0 [110/61] via 155.1.79.7, 00:08:02, Vlan79
O IA    150.1.5.0 [110/3361] via 155.1.79.7, 00:07:53, Vlan79
O IA    150.1.4.0 [110/361] via 155.1.79.7, 00:07:48, Vlan79
O IA    150.1.3.0 [110/6599] via 155.1.79.7, 00:07:53, Vlan79
O IA    150.1.2.0 [110/6599] via 155.1.79.7, 00:07:53, Vlan79
O IA    150.1.1.0 [110/361] via 155.1.79.7, 00:07:53, Vlan79
O IA    150.1.10.0 [110/3516] via 155.1.79.7, 00:07:53, Vlan79
O IA    150.1.8.0 [110/3366] via 155.1.79.7, 00:07:53, Vlan79

Rack1SW3#traceroute 155.1.10.10

Type escape sequence to abort.
Tracing the route to 155.1.10.10

  1 155.1.79.7 0 msec 0 msec 0 msec
  2 155.1.67.6 4 msec 0 msec 4 msec
  3 155.1.146.1 0 msec 4 msec 0 msec
  4 155.1.0.5 32 msec 28 msec 28 msec
  5 155.1.58.8 32 msec 28 msec 28 msec
  6 155.1.108.10 32 msec *  28 msec
```

Copyright © 2010 Internetwork Expert

Once R3's connection to SW1 comes back up, traffic from area 2 still uses R6 as the exit point due to the shorter SPF. For example take SW3's path to the VLAN 10 network 155.1.10.0/24.

```
Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#
Rack1R3(config)#interface FastEthernet0/0
Rack1R3(config-if)#no shutdown
Rack1R3(config-if)#end
Rack1R3#
<output omitted>
Rack1R3#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         0   FULL/  -        00:01:33    155.1.0.5       Serial1/0
150.1.7.7         1   FULL/DR         00:00:37    155.1.37.7      FastEthernet0/0
150.1.1.1         0   FULL/  -        00:00:30    155.1.13.1      Serial1/2
150.1.2.2         0   FULL/  -        00:00:35    155.1.23.2      Serial1/3

Rack1SW3#show ip route 150.1.10.0
Routing entry for 150.1.10.0/24
  Known via "ospf 1", distance 110, metric 3516, type inter area
  Last update from 155.1.79.7 on Vlan79, 00:02:30 ago
  Routing Descriptor Blocks:
  * 155.1.79.7, from 150.1.6.6, 00:02:30 ago, via Vlan79
      Route metric is 3516, traffic share count is 1
```

SW3 says the metric to reach 150.1.10.0/24 is 3516 via the ABR 150.1.6.6. Why this path is being chosen can be verified by checking the metric that R3 and R6 are originating as LSA 3, and then find how the LSA 1 Intra-Area SPF calculation occurs to the ABR.

```
Rack1SW3#show ip ospf database summary 155.1.10.0

            OSPF Router with ID (150.1.9.9) (Process ID 1)

                Summary Net Link States (Area 2)

  LS age: 1805
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 155.1.10.0 (summary Network Number)
  Advertising Router: 150.1.3.3
  LS Seq Number: 80000014
  Checksum: 0xC665
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 65720
```

```
  Routing Bit Set on this LSA
  LS age: 1504
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 155.1.10.0 (summary Network Number)
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000001
  Checksum: 0x2424
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 3485
```

R3 says that the prefix 155.1.10.0/24 is reachable with the metric 65720, while R6 is advertising it with the metric 3485. Now we must determine the shortest path to the ABRs.

```
Rack1SW3#show ip ospf database router 150.1.9.9 self-originate

            OSPF Router with ID (150.1.9.9) (Process ID 1)

                Router Link States (Area 2)

  LS age: 700
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.9.9
  Advertising Router: 150.1.9.9
  LS Seq Number: 80000011
  Checksum: 0x162F
  Length: 60
  Number of Links: 3

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 150.1.9.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.79.7
     (Link Data) Router Interface address: 155.1.79.9
      Number of TOS metrics: 0
       TOS 0 Metrics: 30

<output omitted>
```

SW3's metric to the VLAN 79 DR (and hence SW1) is 30. The total metric to R3's advertisement is now 65750, while R6's advertisement is 3515.

```
Rack1SW1#show ip ospf database router 150.1.7.7 self-originate

            OSPF Router with ID (150.1.7.7) (Process ID 1)

                Router Link States (Area 2)

  LS age: 767
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.7.7
  Advertising Router: 150.1.7.7
  LS Seq Number: 80000019
  Checksum: 0xE39F
  Length: 84
  Number of Links: 5

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 150.1.7.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.37.7
     (Link Data) Router Interface address: 155.1.37.7
      Number of TOS metrics: 0
       TOS 0 Metrics: 300

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.79.7
     (Link Data) Router Interface address: 155.1.79.7
      Number of TOS metrics: 0
       TOS 0 Metrics: 30

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.67.6
     (Link Data) Router Interface address: 155.1.67.7
      Number of TOS metrics: 0
       TOS 0 Metrics: 30

<output omitted>
```

SW1's metric to the DR on the link to R3 (and hence R3) is 300, while the metric to the VLAN 67 DR (and hence R6) is 30. The total metric through R3 is now 66050, while the total metric to R6 is 3545, resulting in the path through R6 being installed in the routing table.

```
Rack1SW3#show ip route 155.1.10.0
Routing entry for 155.1.10.0/24
  Known via "ospf 1", distance 110, metric 3545, type inter area
  Last update from 155.1.79.7 on Vlan79, 01:00:04 ago
  Routing Descriptor Blocks:
  * 155.1.79.7, from 150.1.6.6, 01:00:04 ago, via Vlan79
      Route metric is 3545, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

6.13 OSPF Path Selection with Non-Backbone Transit Areas

- Configure R4's Frame Relay and point-to-point Serial links connecting to R5 to have an OSPF cost of 1.
- Modify the SPF calculation in the OSPF domain so that R6 cannot use R4 to reach area 0 by transiting area 1.
- Verify this by ensuring that traffic from R6 going to the Loopback 0 network of SW2 is still first sent to R1, and then directly to R5.

Configuration

```
R1:
router ospf 1
 no capability transit

R4:
interface Serial0/0/0
 ip ospf cost 1
!
interface Serial0/1/0
 ip ospf cost 1

R6:
router ospf 1
 no capability transit
```

Verification

Note

Per RFC 2328 (OSPF Version 2) section 16.3, *Examining transit areas' summary-LSAs*, non-backbone (not area 0) areas can be used for inter-area transit if a shorter path can be found through them, and if the "*TransitCapability parameter has been set to TRUE in Step 2 of the Dijkstra algorithm*". In Cisco's IOS implementation this flag is controlled with the **`capability transit`** routing process level command, and is on by default.

The design case when the feature is used is very specific, and has to do with a shorter Inter-Area path being found via a non-area 0 router as compared to the target router of a virtual-link. To understand this in detail let's first see R6's path selection to 150.1.8.0/24 when R4's links to R5 both have a cost of 1.

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip route 150.1.8.8
Routing entry for 150.1.8.0/24
  Known via "ospf 1", distance 110, metric 307, type inter area
  Last update from 155.1.146.4 on FastEthernet0/0.146, 00:12:31 ago
  Routing Descriptor Blocks:
  * 155.1.146.4, from 150.1.5.5, 00:12:31 ago, via FastEthernet0/0.146
      Route metric is 307, traffic share count is 1
```

R6's virtual-link to area 0 is via R1. Normally, we would assume that R6 must route via the virtual-link path to R1 to reach Inter-Area destinations advertised by area 0. However, in this case R6 says that the Inter-Area route 150.1.8.0/24 is via the next-hop 155.1.146.4 (R4), but originated by the router 150.1.5.5 (R5). The below traceroute output indicates that R1 is not used in the transit path for this traffic.

```
Rack1R6#traceroute 150.1.8.8

Type escape sequence to abort.
Tracing the route to 150.1.8.8

  1 155.1.146.4 4 msec 0 msec 0 msec
  2 155.1.0.5 20 msec 20 msec 20 msec
  3 155.1.58.8 20 msec *  20 msec
```

To determine why R6 chooses this path we must first find out how R6 is learning the Inter-Area LSA 3 route.

```
  Rack1R6#show ip ospf database summary 150.1.8.0

            OSPF Router with ID (150.1.6.6) (Process ID 1)

                Summary Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 342 (DoNotAge)
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.8.0 (summary Network Number)
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000014
  Checksum: 0x1E1
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 6
<output omitted>
```

This output indicates that 150.1.5.5 (R5) is the originating ABR, and has a metric of 6. We must now figure out how to route towards the ABR R5.

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip ospf database router 150.1.5.5

            OSPF Router with ID (150.1.6.6) (Process ID 1)

                Router Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 342 (DoNotAge)
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.5.5
  Advertising Router: 150.1.5.5
  LS Seq Number: 8000001E
  Checksum: 0x2539
  Length: 108
  Area Border Router
  Number of Links: 7

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.3.3
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 3238

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.1.1
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 58593

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.2.2
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 3238

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 223.255.255.255
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 39062

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 155.1.0.5
     (Link Data) Network Mask: 255.255.255.255
      Number of TOS metrics: 0
       TOS 0 Metrics: 0

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 223.255.255.255
     (Link Data) Router Interface address: 155.1.45.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 10000

<output omitted>
```

Copyright © 2010 Internetwork Expert

R5's area 0 Router LSA (LSA 1) states that it is adjacent with R1 over the Frame Relay network and R4 over the Frame Relay and point-to-point Serial links. Now we must determine what R1 and R4's costs are to R5.

```
Rack1R6#show ip ospf database router 150.1.1.1

            OSPF Router with ID (150.1.6.6) (Process ID 1)

                Router Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 1 (DoNotAge)
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.1.1
  Advertising Router: 150.1.1.1
  LS Seq Number: 8000001B
  Checksum: 0x4E2C
  Length: 72
  Area Border Router
  Number of Links: 4

    Link connected to: a Virtual Link
     (Link ID) Neighboring Router ID: 150.1.6.6
     (Link Data) Router Interface address: 155.1.146.1
      Number of TOS metrics: 0
       TOS 0 Metrics: 300

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 150.1.1.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.5.5
     (Link Data) Router Interface address: 155.1.0.1
      Number of TOS metrics: 0
       TOS 0 Metrics: 3000

<output omitted>
```

R1 says it's metric to R5 is 3000 via the Frame Relay link.

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip ospf database router 223.255.255.255

            OSPF Router with ID (150.1.6.6) (Process ID 1)

                Router Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 3 (DoNotAge)
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 223.255.255.255
  Advertising Router: 223.255.255.255
  LS Seq Number: 8000001D
  Checksum: 0x9EBF
  Length: 84
  Area Border Router
  Number of Links: 5

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 150.1.4.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.5.5
     (Link Data) Router Interface address: 155.1.0.4
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 155.1.0.4
     (Link Data) Network Mask: 255.255.255.255
      Number of TOS metrics: 0
       TOS 0 Metrics: 0

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.5.5
     (Link Data) Router Interface address: 155.1.45.4
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 155.1.45.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 1
```

R4 says its metrics are both 1 to R5 over the Frame Relay and point-to-point links. Now we must find what R6's metrics to R1 and R4 are.

```
Rack1R6#show ip ospf database router 150.1.6.6 self-originate

            OSPF Router with ID (150.1.6.6) (Process ID 1)

                Router Link States (Area 0)

  LS age: 50
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.6.6
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000023
  Checksum: 0xCBFF
  Length: 36
  Area Border Router
  Number of Links: 1

    Link connected to: a Virtual Link
     (Link ID) Neighboring Router ID: 150.1.1.1
     (Link Data) Router Interface address: 155.1.146.6
      Number of TOS metrics: 0
       TOS 0 Metrics: 300

                Router Link States (Area 1)

  LS age: 50
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.6.6
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000038
  Checksum: 0xEF7A
  Length: 48
  Area Border Router
  Virtual Link Endpoint
  Number of Links: 2

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 150.1.6.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.146.6
     (Link Data) Router Interface address: 155.1.146.6
      Number of TOS metrics: 0
       TOS 0 Metrics: 300
<output omitted>
```

Copyright © 2010 Internetwork Expert

R6 has reachability to R1 via the virtual-link with a metric of 300. R6 has reachability to R4 via the DR of VLAN 146 in area 1 with a metric of 300. Now R6 must make its final determination on how to route.

If R6 routes through R1 via the virtual-link the metric is 300 to R1, 3000 from R1 to R5, and R5's metric of 6 to the final destination, for a total of 3306. If R6 routes through R4 via area 1 the metric is 300 to R4, 1 from R4 to R5, and R5's metric of 6 to the final destination, for a total of 307.

Under normal conditions R6 should choose to route through R1 since this is the route through area 0. However, if the **capability transit** feature is enabled, which it is by default in IOS 12.4, R6 can choose the shorter path through area 1 to R4, resulting in the final metric of 307 via R4 being installed in the routing table.

```
Rack1R6#show ip route 150.1.8.8
Routing entry for 150.1.8.0/24
  Known via "ospf 1", distance 110, metric 307, type inter area
  Last update from 155.1.146.4 on FastEthernet0/0.146, 17:01:50 ago
  Routing Descriptor Blocks:
  * 155.1.146.4, from 150.1.5.5, 17:01:50 ago, via FastEthernet0/0.146
      Route metric is 307, traffic share count is 1
```

If area 1 is not allowed to be used as transit, by issuing the **no capability transit**, R6 cannot choose this route.

```
Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#router ospf 1
Rack1R6(config-router)#no capability transit
Rack1R6(config-router)#end
Rack1R6#

Rack1R6#show ip route 150.1.8.8
Routing entry for 150.1.8.0/24
  Known via "ospf 1", distance 110, metric 3306, type inter area
  Last update from 155.1.146.1 on FastEthernet0/0.146, 00:00:06 ago
  Routing Descriptor Blocks:
  * 155.1.146.1, from 150.1.5.5, 00:00:06 ago, via FastEthernet0/0.146
      Route metric is 3306, traffic share count is 1
```

R6 now says the route is through R1 with a metric of 3306, which matches what we calculated from the OSPF database. The traceroute output however, is still not exactly the desired result.

Copyright © 2010 Internetwork Expert

```
Rack1R6#traceroute 150.1.8.8

Type escape sequence to abort.
Tracing the route to 150.1.8.8

  1 155.1.146.1 0 msec 4 msec 4 msec
  2 155.1.146.4 4 msec 0 msec 4 msec
  3 155.1.0.5 20 msec 20 msec 20 msec
  4 155.1.58.8 20 msec *  20 msec
```

R6 sends its packet to R1, but R1 takes it and redirects it out the same interface back to R4. This is due to the fact that R1 now says its shorter path to the router R5 is through area 1 via R4.

```
Rack1R1#show ip ospf database router 150.1.1.1 self-originate

            OSPF Router with ID (150.1.1.1) (Process ID 1)

                Router Link States (Area 0)

  LS age: 740
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.1.1
  Advertising Router: 150.1.1.1
  LS Seq Number: 8000003C
  Checksum: 0xC4D
  Length: 72
  Area Border Router
  Number of Links: 4

    Link connected to: a Virtual Link
     (Link ID) Neighboring Router ID: 150.1.6.6
     (Link Data) Router Interface address: 155.1.146.1
      Number of TOS metrics: 0
       TOS 0 Metrics: 300

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 150.1.1.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.5.5
     (Link Data) Router Interface address: 155.1.0.1
      Number of TOS metrics: 0
       TOS 0 Metrics: 3000

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 155.1.0.1
     (Link Data) Network Mask: 255.255.255.255
      Number of TOS metrics: 0
       TOS 0 Metrics: 0
```

Copyright © 2010 Internetwork Expert

```
                Router Link States (Area 1)

  LS age: 744
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.1.1
  Advertising Router: 150.1.1.1
  LS Seq Number: 80000035
  Checksum: 0x6DC7
  Length: 36
  Area Border Router
  Virtual Link Endpoint
  Number of Links: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.146.6
     (Link Data) Router Interface address: 155.1.146.1
      Number of TOS metrics: 0
       TOS 0 Metrics: 300
```

R1 says that its direct adjacency with R5 over the Frame Relay link is via a metric of 3000 in area 0. R1's adjacency with the VLAN 146 DR (and hence R4) is 300 via area 1. Since R4's metric to R5 is 1, R1 says that the metric of 301 via area 1 is closer than the metric of 3000 via area 0. Since capability transit is on R1, it can use R4 to reach R5 instead of using area 0. With R5's metric of 6 to the destination R1 then inserts the total metric of 307 via R4, but from R5.

```
Rack1R1#show ip route 150.1.8.8
Routing entry for 150.1.8.0/24
  Known via "ospf 1", distance 110, metric 307, type inter area
  Last update from 155.1.146.4 on FastEthernet0/0, 00:11:39 ago
  Routing Descriptor Blocks:
  * 155.1.146.4, from 150.1.5.5, 00:11:39 ago, via FastEthernet0/0
      Route metric is 307, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

If we disable R1's ability to use area 1 as transit to reach area 0 it will force R1 to route directly to R5, instead of to R4. This will force traffic to flow from R6 to SW2 as required by the task's parameters.

```
Rack1R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#router ospf 1
Rack1R1(config-router)#no capability transit
Rack1R1(config-router)#end
Rack1R1#

Rack1R1#show ip route 150.1.8.8
Routing entry for 150.1.8.0/24
  Known via "ospf 1", distance 110, metric 3006, type inter area
  Last update from 155.1.0.5 on Serial0/0, 00:00:06 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.5.5, 00:00:06 ago, via Serial0/0
      Route metric is 3006, traffic share count is 1

Rack1R6#traceroute 150.1.8.8

Type escape sequence to abort.
Tracing the route to 150.1.8.8

  1 155.1.146.1 4 msec 4 msec 0 msec
  2 155.1.0.5 24 msec 20 msec 24 msec
  3 155.1.58.8 20 msec *  20 msec
```

Copyright © 2010 Internetwork Expert

6.14 OSPF Path Selection with Virtual-Links

- Configure the OSPF domain so that traffic from SW3 going to OSPF area 51 transits the point-to-point Serial link between R3 and R2.

Configuration

```
R2:
router ospf 1
 area 5 virtual-link 150.1.3.3

R3:
interface Serial1/3
 ip ospf cost 1
!
router ospf 1
 area 5 virtual-link 150.1.2.2
```

Verification

Note

As we saw in the previous section where the virtual-link was added between R1 and R6, traffic from area 2 into area 0 transited R6. This was due to the fact that R3's slow serial links have a much higher cost value than R6 has via R1. This can be verified from the below traceroute from SW3 to area 51.

```
Rack1SW3#traceroute 51.51.51.51

Type escape sequence to abort.
Tracing the route to 51.51.51.51

  1 155.1.79.7 0 msec 0 msec 4 msec
  2 155.1.67.6 0 msec 4 msec 0 msec
  3 155.1.146.1 4 msec 0 msec 4 msec
  4 155.1.0.5 28 msec 32 msec 28 msec
  5 155.1.0.2 28 msec 32 msec 28 msec
  6 192.10.1.254 32 msec *  32 msec
```

Copyright © 2010 Internetwork Expert

The goal of this section is to modify the path selection process so that area 2 exits via R3, and uses the link through area 5. Recall though that inter-area routing can only occur through area 0, or in the previous specific design, by way of a shorter path to area 0 via a non-area 0 transit area. Based on this design however, the area 5 link can never be used to reach area 51 regardless of the cost. This can be illustrated as follows.

```
Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#interface Serial1/3
Rack1R3(config-if)#ip ospf cost 1
Rack1R3(config-if)#end
Rack1R3#

Rack1R3#show ip route 51.51.51.51
Routing entry for 51.51.51.51/32
  Known via "ospf 1", distance 110, metric 20, type extern 2, forward metric
69073
  Last update from 155.1.0.5 on Serial1/0, 00:00:14 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 192.10.1.254, 00:00:14 ago, via Serial1/0
      Route metric is 20, traffic share count is 1

Rack1R3#traceroute 51.51.51.51

Type escape sequence to abort.
Tracing the route to 51.51.51.51

  1 155.1.0.5 28 msec 32 msec 28 msec
  2 155.1.0.2 64 msec 56 msec 56 msec
  3 192.10.1.254 60 msec *  60 msec
```

R3 sets the cost of the link to R2 via area 5 to 1. Even though the cost is lower through this link than through area 0 to R5, the direct route to R2 cannot be used because R3 does not know how to reach the ASBR for area 51 via area 5. This can be seen by recursing towards the external route 51.51.51.51/32 that comes from area 51.

Copyright © 2010 Internetwork Expert

```
Rack1R3#show ip ospf database external 51.51.51.51

            OSPF Router with ID (150.1.3.3) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 342
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 51.51.51.51 (External Network Number )
  Advertising Router: 192.10.1.254
  LS Seq Number: 8000005D
  Checksum: 0xE3C8
  Length: 36
  Network Mask: /32
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 0.0.0.0
        External Route Tag: 0
```

R3 says the prefix 51.51.51.51 has a forward address of 0.0.0.0. This means that R3 needs to lookup the advertising router for the external LSA, who in this case is 192.10.1.254.

Copyright © 2010 Internetwork Expert

```
Rack1R3#show ip ospf database asbr-summary 192.10.1.254

            OSPF Router with ID (150.1.3.3) (Process ID 1)

                Summary ASB Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 519
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(AS Boundary Router)
  Link State ID: 192.10.1.254 (AS Boundary Router address)
  Advertising Router: 150.1.2.2
  LS Seq Number: 80000032
  Checksum: 0x2B4C
  Length: 28
  Network Mask: /0
        TOS: 0  Metric: 300

<output omitted>

                Summary ASB Link States (Area 5)

  Adv Router is not-reachable
  LS age: 283
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(AS Boundary Router)
  Link State ID: 192.10.1.254 (AS Boundary Router address)
  Advertising Router: 150.1.2.2
  LS Seq Number: 80000032
  Checksum: 0x2B4C
  Length: 28
  Network Mask: /0
        TOS: 0  Metric: 300

<output omitted>
```

The above output indicates that R3 has multiple paths to the ASBR of area 51, however most are invalid. R3 can use the area 0 route towards R2 to reach 192.10.1.254, but it cannot use the area 5 route towards R2, as the *Adv Router is not-reachable* through this path. This output essentially means that area 5 cannot be used as transit to reach area 51, only area 0 can. Therefore, the only way to force the traffic to use the point-to-point Serial link between R2 and R3 is to give R3 an area 0 route towards R2 via this link. This can be accomplished by configuring a tunnel interface over the link in area 0, or more simply, through a virtual-link.

Copyright © 2010 Internetwork Expert

```
Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#router ospf 1
Rack1R3(config-router)#area 5 virtual-link 150.1.2.2
Rack1R3(config-router)#end
Rack1R3#

Rack1R2#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R2(config)#router ospf 1
Rack1R2(config-router)#area 5 virtual-link 150.1.3.3
Rack1R2(config-router)#end
Rack1R2#
```

Now that R3 has an area 0 route to R2 via the virtual-link through area 5, and the cost of the area 5 link is 1, R3 chooses this as the shortest path towards the ASBR. The final result is that R3 advertises a better cost towards area 51 into area 2 than R6 does.

```
Rack1SW3#show ip ospf database external 51.51.51.51

            OSPF Router with ID (150.1.9.9) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 1592
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 51.51.51.51 (External Network Number )
  Advertising Router: 192.10.1.254
  LS Seq Number: 8000005D
  Checksum: 0xE3C8
  Length: 36
  Network Mask: /32
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 0.0.0.0
        External Route Tag: 0
```

SW3 now says that the external prefix 51.51.51.51/32 is reachable through the ASBR 192.10.1.254, which is the same output it would have shown before the changes on R3. What is different now however, is how area 2 routes towards the ASBR.

Copyright © 2010 Internetwork Expert

```
Rack1SW3#show ip ospf database asbr-summary 192.10.1.254

            OSPF Router with ID (150.1.9.9) (Process ID 1)

                Summary ASB Link States (Area 2)

  Routing Bit Set on this LSA
  LS age: 140
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(AS Boundary Router)
  Link State ID: 192.10.1.254 (AS Boundary Router address)
  Advertising Router: 150.1.3.3
  LS Seq Number: 80000039
  Checksum: 0x1A53
  Length: 28
  Network Mask: /0
        TOS: 0  Metric: 301

  LS age: 57
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(AS Boundary Router)
  Link State ID: 192.10.1.254 (AS Boundary Router address)
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000025
  Checksum: 0x5C7C
  Length: 28
  Network Mask: /0
        TOS: 0  Metric: 6838
```

SW3 says the ASBR is reachable via the area 2 ABRs 150.1.3.3 (R3) and 150.1.6.6 (R6). R3's reported metric is 301, while R6's metric is 6838. Based on this SW3 chooses to use R3 as the exit point, and the traffic is routed as desired.

```
Rack1SW3#traceroute 51.51.51.51

Type escape sequence to abort.
Tracing the route to 51.51.51.51

  1 155.1.79.7 0 msec 0 msec 0 msec
  2 155.1.37.3 0 msec 0 msec 4 msec
  3 155.1.23.2 16 msec 12 msec 16 msec
  4 192.10.1.254 20 msec *  16 msec
```

Copyright © 2010 Internetwork Expert

6.15 OSPF Demand Circuit

- The point-to-point connection between R4 and R5 is a low-speed Serial link. Configure the OSPF demand circuit feature on this link to reduce periodic OSPF hello transmission and paranoid update flooding.

Configuration

```
R4:
interface Serial0/1/0
 ip ospf demand-circuit
```

Verification

> **Note**
>
> Per RFC 1793, *Extending OSPF to Support Demand Circuits*, "*OSPF Hellos and the refresh of OSPF routing information are suppressed on demand circuits, allowing the underlying data-link connections to be closed when not carrying application traffic.*" This feature allows low-speed and pay-per-use links, such as analog dial, ISDN, etc. to run OSPF without the need for periodic hellos and LSA flooding.
>
> This feature is enabled with the interface level command **`ip ospf demand-circuit`**, and is negotiated as part of the neighbor adjacency establishment. The show command below output outlines the change in the adjacency between R4 and R5 once the demand circuit feature is enabled.

```
Rack1R4#show ip ospf interface Serial0/1/0
Serial0/1/0 is up, line protocol is up
  Internet Address 155.1.45.4/24, Area 0
  Process ID 1, Router ID 223.255.255.255, Network Type POINT_TO_POINT, Cost: 1
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:00
  Supports Link-local Signaling (LLS)
  Index 2/3, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 5 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.5.5
  Suppress hello for 0 neighbor(s)
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface Serial0/1/0
Rack1R4(config-if)#ip ospf demand-circuit
Rack1R4(config-if)#end
Rack1R4#
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.5.5 on Serial0/1/0 from FULL to DOWN,
Neighbor Down: Interface down or detached
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.5.5 on Serial0/1/0 from LOADING to FULL,
Loading Done
%SYS-5-CONFIG_I: Configured from console by console

Rack1R4#show ip ospf interface Serial0/1/0
Serial0/1/0 is up, line protocol is up
  Internet Address 155.1.45.4/24, Area 0
  Process ID 1, Router ID 223.255.255.255, Network Type POINT_TO_POINT, Cost: 1
  Enabled by interface config, including secondary ip addresses
  Configured as demand circuit.
  Run as demand circuit.
  DoNotAge LSA allowed.
  Transmit Delay is 1 sec, State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:06
  Supports Link-local Signaling (LLS)
  Index 2/3, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 5 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.5.5  (Hello suppressed)
  Suppress hello for 1 neighbor(s)

Rack1R4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         0   FULL/  -            -        155.1.45.5      Serial0/1/0
150.1.5.5         0   FULL/  -        00:01:53     155.1.0.5       Serial0/0/0
150.1.1.1         1   FULL/BDR       00:00:35     155.1.146.1     FastEthernet0/1
150.1.6.6       255   FULL/DR        00:00:39     155.1.146.6     FastEthernet0/1
```

The above output indicates that the link between R4 and R5 is set to *Run as demand circuit*, that *DoNotAge LSA* is allowed, and that R4 is going to *Suppress hello for 1 neighbor*. This indicates that periodic hellos are not sent, and the "paranoid" flooding of LSAs is disabled. Normally once an LSA reaches an age of 30 minutes it must be re-flooded, regardless if the network is stable and has not changed. The lack of periodic hello exchange can also be seen in the **`show ip ospf neighbor`** output, notice that the Dead Time field for the adjacency to R5 is null "—".

Copyright © 2010 Internetwork Expert

6.16 OSPF Flooding Reduction

- Configure R5, SW2, and SW4 so that links in OSPF area 3 do not participate in periodic paranoid update LSA flooding.

Configuration

```
R5:
interface FastEthernet0/0
 ip ospf flood-reduction
!
interface FastEthernet0/1
 ip ospf flood-reduction

SW2:
interface Vlan8
 ip ospf flood-reduction
!
interface Vlan58
 ip ospf flood-reduction
!
interface Port-channel1
 ip ospf flood-reduction

SW4:
interface Vlan10
 ip ospf flood-reduction
!
interface Port-channel1
 ip ospf flood-reduction
```

Verification

Note

Per RFC 2328, *OSPF Version 2*, "*an LSA's LS age is never incremented past the value MaxAge.*" When the Link State Age reaches MaxAge "*the router must attempt to flush the LSA... by reflooding the MaxAge LSA just as if it was a newly originated LSA*".

In Cisco's IOS implementation of OSPF the MaxAge value is 3600 seconds, or 30 minutes. This means that each LSA is reflooded after 30 minutes, regardless if the topology is stable. This periodic flooding behavior is commonly referred to as the "paranoid update". The **`ip ospf flood-reduction`** feature stops unnecessary LSA flooding by setting the DoNotAge (DNA) bit in the LSA, removing the requirement for the periodic refresh.

Copyright © 2010 Internetwork Expert

In the below output SW4 has the (DNA) field next to all LSAs learned from area 3, which indicates that they do not need to be periodically flooded.

```
Rack1SW4#show ip ospf database

            OSPF Router with ID (223.255.255.255) (Process ID 1)

                Router Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum Link count
150.1.5.5       150.1.5.5       2     (DNA) 0x80000048 0x0071E8 3
150.1.8.8       150.1.8.8       1     (DNA) 0x80000042 0x0088F6 4
223.255.255.255 223.255.255.255 32          0x8000003C 0x00BF37 3

                Net Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.58.5      150.1.5.5       2     (DNA) 0x80000001 0x00482B
155.1.108.8     150.1.8.8       1     (DNA) 0x80000001 0x0055A4

                Summary Net Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
150.1.1.0       150.1.5.5       13    (DNA) 0x80000037 0x00B450
150.1.2.0       150.1.5.5       1546  (DNA) 0x8000003A 0x00B361
150.1.3.0       150.1.5.5       1546  (DNA) 0x8000003A 0x00A86B
150.1.4.0       150.1.5.5       767   (DNA) 0x80000039 0x00B1DD
150.1.6.0       150.1.5.5       767   (DNA) 0x8000003A 0x005C03
150.1.7.0       150.1.5.5       13    (DNA) 0x8000002A 0x005F93
150.1.9.0       150.1.5.5       13    (DNA) 0x8000002A 0x00765C
155.1.0.1       150.1.5.5       13    (DNA) 0x80000037 0x006A96
155.1.0.2       150.1.5.5       1546  (DNA) 0x8000003A 0x006AA6
155.1.0.3       150.1.5.5       1546  (DNA) 0x8000003A 0x0060AF
155.1.0.4       150.1.5.5       767   (DNA) 0x80000039 0x006A21
155.1.0.5       150.1.5.5       13    (DNA) 0x80000038 0x006161
155.1.7.0       150.1.5.5       13    (DNA) 0x8000002A 0x00418F
155.1.9.0       150.1.5.5       13    (DNA) 0x8000002A 0x005759
155.1.13.0      150.1.5.5       1546  (DNA) 0x8000003B 0x00EA1A
155.1.23.0      150.1.5.5       1029  (DNA) 0x8000003B 0x008871
155.1.37.0      150.1.5.5       13    (DNA) 0x80000029 0x00CA07
155.1.45.0      150.1.5.5       767   (DNA) 0x8000003A 0x009FC1
155.1.67.0      150.1.5.5       13    (DNA) 0x8000002A 0x00AAE9
155.1.79.0      150.1.5.5       13    (DNA) 0x8000002A 0x002662
155.1.146.0     150.1.5.5       767   (DNA) 0x8000003A 0x0007C7
192.10.1.0      150.1.5.5       1546  (DNA) 0x8000003A 0x00E6CF

                Summary ASB Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
192.10.1.254    150.1.5.5       1546  (DNA) 0x8000003A 0x00E2D3

                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
51.51.51.51     192.10.1.254    587   (DNA) 0x80000063 0x00D7CE 0
```

Copyright © 2010 Internetwork Expert

6.17 OSPF Clear Text Authentication

- Configure clear-text OSPF authentication for all adjacencies in area 2 using the password CLEARKEY.
- SW1 should enable authentication on all interfaces in area 2 with one single command.
- R3, R6, and SW3 should only enable authentication on their links connecting to SW1.

Configuration

```
R3:
interface FastEthernet0/0
 ip ospf authentication
 ip ospf authentication-key CLEARKEY

R6:
interface FastEthernet0/0.67
 ip ospf authentication
 ip ospf authentication-key CLEARKEY

SW1:
interface Vlan67
 ip ospf authentication-key CLEARKEY
!
interface Vlan79
 ip ospf authentication-key CLEARKEY
!
interface FastEthernet0/3
 ip ospf authentication-key CLEARKEY
!
router ospf 1
 area 2 authentication

SW3:
interface Vlan79
 ip ospf authentication
 ip ospf authentication-key CLEARKEY
```

Copyright © 2010 Internetwork Expert

Verification

Note

OSPF supports three types of authentication, type 0, or null authentication (no authentication), type 1, or clear text authentication, and type 2, or MD5 authentication. The type of authentication can be configured at the interface level with the **ip ospf authentication** command, or at the process level with the **area [id] authentication** command. The only difference between these commands is whether authentication is enabled on all interfaces in the area at the same time, or on a per link basis. In either case the password must still be configured at the interface level with the **ip ospf authentication-key** or **ip ospf message-digest-key** commands.

A successfully authenticated adjacency can be verified as follows.

```
Rack1SW3#show ip ospf interface Vlan79 | include authentication
  Simple password authentication enabled

Rack1SW3#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.7.7         1   FULL/DR         00:00:30    155.1.79.7      Vlan79
```

Pitfall

A failure in authentication can occur for two reasons, a mismatch in authentication type, or a mismatch involving the authentication key. An authentication type mismatch occurs when one neighbor is configured with clear text, while the other is running MD5, or one is running MD5, and the other is running null, etc. A key mismatch is simply when the routers are using different passwords. Failure in authentication type can be verified as follows.

Copyright © 2010 Internetwork Expert

```
Rack1SW3#debug ip ospf adj
OSPF adjacency events debugging is on

Rack1SW3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW3(config)#interface Vlan79
Rack1SW3(config-if)#ip ospf authentication message-digest
Rack1SW3(config-if)#
OSPF: Send with youngest Key 0
OSPF: Rcv pkt from 155.1.79.7, Vlan79 : Mismatch Authentication type. Input
packet specified type 1, we use type 2
```

The above output from **`debug ip ospf adj`** indicates a *Mismatch Authentication type*, where the inbound OSPF packet is using type 1 and the local router is using type 2. This indicates that the local router is trying to do MD5 authentication and the remote router is using clear text.

```
Rack1SW3#debug ip ospf adj
OSPF adjacency events debugging is on

Rack1SW3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW3(config)#interface Vlan79
Rack1SW3(config-if)#ip ospf authentication null
OSPF: Rcv pkt from 155.1.79.7, Vlan79 : Mismatch Authentication type. Input
packet specified type 1, we use type 0
```

The above output indicates that SW3 is not doing authentication, i.e. type 0, and the remote end is doing clear text authentication, i.e. type 1. A mismatch in the password itself can be seen as follows.

```
Rack1SW3#debug ip ospf adj
OSPF adjacency events debugging is on

Rack1SW3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW3(config)#interface Vlan79
Rack1SW3(config-if)#ip ospf authentication-key WRONGKEY
Rack1SW3(config-if)#
OSPF: Rcv pkt from 155.1.79.7, Vlan79 : Mismatch Authentication Key - Clear
Text
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

6.18 OSPF MD5 Authentication

- Configure MD5 based OSPF authentication for all adjacencies in area 0, including the Virtual-Links, using the password MD5KEY.
- R1 should enable MD5 authentication on all interfaces in area 0 with a single command.
- All other devices in area 0 should enable MD5 authentication on a per interface basis.

Configuration

```
R1:
interface Serial0/0
 ip ospf message-digest-key 1 md5 MD5KEY
!
router ospf 1
 area 0 authentication message-digest
 area 1 virtual-link 150.1.6.6 message-digest-key 1 md5 MD5KEY

R2:
interface Serial0/0
 ip ospf authentication message-digest
 ip ospf message-digest-key 1 md5 MD5KEY
!
router ospf 1
 area 5 virtual-link 150.1.3.3 authentication message-digest
 area 5 virtual-link 150.1.3.3 message-digest-key 1 md5 MD5KEY

R3:
interface Serial1/0
 ip ospf authentication message-digest
 ip ospf message-digest-key 1 md5 MD5KEY
!
router ospf 1
 area 5 virtual-link 150.1.2.2 authentication message-digest
 area 5 virtual-link 150.1.2.2 message-digest-key 1 md5 MD5KEY

R4:
interface Serial0/0/0
 ip ospf authentication message-digest
 ip ospf message-digest-key 1 md5 MD5KEY
!
interface Serial0/1/0
 ip ospf authentication message-digest
 ip ospf message-digest-key 1 md5 MD5KEY

R5:
interface Serial0/0/0
 ip ospf authentication message-digest
 ip ospf message-digest-key 1 md5 MD5KEY
!
interface Serial0/1/0
 ip ospf authentication message-digest
 ip ospf message-digest-key 1 md5 MD5KEY
```

Copyright © 2010 Internetwork Expert

```
R6:
router ospf 1
 area 1 virtual-link 150.1.1.1 authentication message-digest
 area 1 virtual-link 150.1.1.1 message-digest-key 1 md5 MD5KEY
```

Verification

> **Note**
>
> Like clear text authentication, MD5 authentication can be enabled on a per link basis with the interface level command **`ip ospf authentication message-digest`**, or for all links in the area with the process level command **`area [id] authentication message-digest`**. For successful MD5 authentication the authentication type, the password, and the key ID must match. Correctly configured authentication can be verified as follows.

```
Rack1R5#show ip ospf interface | include protocol|authentication|key
Serial0/0/0 is up, line protocol is up
  Message digest authentication enabled
    Youngest key id is 1
Serial0/1/0 is up, line protocol is up
  Message digest authentication enabled
    Youngest key id is 1
FastEthernet0/1 is up, line protocol is up
FastEthernet0/0 is up, line protocol is up
Loopback0 is up, line protocol is up

Rack1R5#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.3.3         0   FULL/  -        00:01:31    155.1.0.3       Serial0/0/0
150.1.1.1         0   FULL/  -        00:01:31    155.1.0.1       Serial0/0/0
150.1.2.2         0   FULL/  -        00:01:31    155.1.0.2       Serial0/0/0
223.255.255.255   0   FULL/  -        00:01:31    155.1.0.4       Serial0/0/0
223.255.255.255   0   FULL/  -           -        155.1.45.4      Serial0/1/0
150.1.8.8         1   FULL/BDR        00:00:33    155.1.58.8      FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

Pitfall

Remember that a virtual-link is an interface in area 0. This means that if the **area 0 authentication [message-digest]** command is enabled, authentication is also enabled on the virtual-link. In this example MD5 authentication is enabled on the virtual-link on R1 with the **area 0 authentication message-digest** command, but the password must still be assigned on the virtual-link interface itself with the **area 1 virtual-link 150.1.6.6 message-digest-key 1 md5 MD5KEY** command. On R6 authentication is enabled at the virtual-link interface level with the **area 1 virtual-link 150.1.1.1 authentication message-digest** command, and the key is applied with the **area 1 virtual-link 150.1.1.1 message-digest-key 1 md5 MD5KEY** command. In some versions these two commands are combined automatically in the running config to the single statement **area 1 virtual-link 150.1.1.1 authentication message-digest message-digest-key 1 md5 MD5KEY**, however the result of either syntax is the same.

Once authentication is enabled on the virtual-link ensure to issue the **clear ip ospf process** command, because the virtual-link does not support periodic hellos. This means that if the authentication is wrong, the virtual-link interface will not immediately go down, but if there is a change in the topology it won't actually be propagated across the virtual-link.

Failures in MD5 authentication can occur because of a type mismatch, a password mismatch, or a key ID mismatch, as verified below.

```
Rack1R5#debug ip ospf adj
OSPF adjacency events debugging is on

Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#no ip ospf message-digest-key 1
Rack1R5(config-if)#ip ospf message-digest-key 1 md5 WRONGKEY
Rack1R5(config-if)#
OSPF: Send with youngest Key 1
OSPF: Rcv pkt from 155.1.0.3, Serial0/0/0 : Mismatch Authentication Key -
Message Digest Key 1
```

This output indicates a password mismatch.

Copyright © 2010 Internetwork Expert

```
Rack1R5(config-if)#no ip ospf message-digest-key 1
Rack1R5(config-if)#ip ospf message-digest-key 2 md5 MD5KEY
Rack1R5(config-if)#
OSPF: Rcv pkt from 155.1.0.3, Serial0/0/0 : Mismatch Authentication Key - No
message digest key 1 on interface
Rack1R5(config-if)#
OSPF: Send with youngest Key 2
```

This output indicates a mismatch in the key ID. R5 is sending key 2, but R3 is sending key 1.

```
Rack1R5(config-if)#ip ospf authentication null
Rack1R5(config-if)#
OSPF: Rcv pkt from 155.1.0.3, Serial0/0/0 : Mismatch Authentication type. Input
packet specified type 2, we use type 0
```

This output indicates a mismatch in authentication type. R5 is using null authentication (type 0), and R3 is using MD5 authentication (type 2).

Copyright © 2010 Internetwork Expert

6.19 OSPF Null Authentication

- Modify SW1 so that if a new router is added to area 2 on VLAN 7 it does not require authentication.

Configuration

```
SW1:
interface Vlan7
 ip ospf authentication null
```

Verification

> **Note**
>
> The authentication type configured at the interface level overrides the authentication type configured at the process level. In this example, SW1 disables authentication on Vlan7 by configuring type 0 (null) authentication at the interface level. Note that the other links in area 2 are not affected by this change.

```
Rack1SW1#show ip ospf interface | include authentication|protocol
Loopback0 is up, line protocol is up
  Simple password authentication enabled
FastEthernet0/3 is up, line protocol is up (connected)
  Simple password authentication enabled
Vlan79 is up, line protocol is up
  Simple password authentication enabled
Vlan67 is up, line protocol is up
  Simple password authentication enabled
Vlan7 is up, line protocol is up
  Simple password authentication enabled

Rack1SW1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface Vlan7
Rack1SW1(config-if)#ip ospf authentication null
Rack1SW1(config-if)#end
Rack1SW1#show ip ospf interface | include authentication|protocol
Loopback0 is up, line protocol is up
  Simple password authentication enabled
FastEthernet0/3 is up, line protocol is up (connected)
  Simple password authentication enabled
Vlan79 is up, line protocol is up
  Simple password authentication enabled
Vlan67 is up, line protocol is up
  Simple password authentication enabled
Vlan7 is up, line protocol is up
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

6.20 OSPF MD5 Authentication with Multiple Keys

- Enable MD5 authentication in OSPF area 1.
- R1 should authenticate R6 using the password R1R6KEY.
- R4 should authenticate R6 using the password R4R6KEY.

Configuration

```
R1:
interface FastEthernet0/0
 ip ospf authentication message-digest
 ip ospf message-digest-key 16 md5 R1R6KEY

R4:
interface FastEthernet0/1
 ip ospf authentication message-digest
 ip ospf message-digest-key 46 md5 R4R6KEY

R6:
interface FastEthernet0/0.146
 ip ospf authentication message-digest
 ip ospf message-digest-key 16 md5 R1R6KEY
 ip ospf message-digest-key 46 md5 R4R6KEY
```

Verification

Note

OSPF supports multiple MD5 keys applied to a single interface in order to allow for key rotation. The normal application of this would be that all neighbors are configured with key 1. Key 2 is then added on all neighbors, and key 1 is removed. Since both keys are temporarily sent, there is no loss of adjacency while the old key is being removed. In this design, multiple keys are used to authenticate different neighbors on the same interface.

```
Rack1R6#debug ip ospf adj
OSPF adjacency events debugging is on
Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface FastEthernet0/0.146
Rack1R6(config-subif)#ip ospf authentication message-digest
OSPF: Send with youngest Key 0
OSPF: Rcv pkt from 155.1.146.4, FastEthernet0/0.146 : Mismatch Authentication
Key - No message digest key 46 on interface
OSPF: Rcv pkt from 155.1.146.1, FastEthernet0/0.146 : Mismatch Authentication
Key - No message digest key 16 on interface
```

R6 has MD5 authentication enabled, but no keys configured. We can see that it receives key 46 from R4 and key 16 from R1.

Copyright © 2010 Internetwork Expert

```
Rack1R6(config-subif)#do undebug all
All possible debugging has been turned off
Rack1R6(config-subif)# ip ospf message-digest-key 16 md5 R1R6KEY
Rack1R6(config-subif)# ip ospf message-digest-key 46 md5 R4R6KEY
Rack1R6(config-subif)#end
Rack1R6#
%OSPF-5-ADJCHG: Process 1, Nbr 223.255.255.255 on FastEthernet0/0.146 from
LOADING to FULL, Loading Done
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.1.1 on FastEthernet0/0.146 from LOADING to
FULL, Loading Done
```

Once both keys are configured adjacency is established.

```
Rack1R6#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address     Interface
150.1.1.1         0   FULL/  -          -         155.1.146.1 OSPF_VL0
150.1.1.1         1   FULL/BDR       00:00:34    155.1.146.1 FastEthernet0/0.146
223.255.255.255   0   FULL/DROTHER   00:00:34    155.1.146.4 FastEthernet0/0.146
150.1.7.7         1   FULL/BDR       00:00:32    155.1.67.7  FastEthernet0/0.67
```

R6 sends both key 16 and 46.

```
Rack1R6#debug ip ospf adj
OSPF adjacency events debugging is on
OSPF: Send with key 16
OSPF: Send with key 46
```

Copyright © 2010 Internetwork Expert

6.21 OSPF Internal Summarization

- Configure R5 to advertise a summary route for the VLAN 8 and 10 prefixes as they are sent into area 0.
- This summary should be as specific as possible while still encompassing all addresses in both subnets.

Configuration

```
R5:
router ospf 1
 area 3 range 155.1.8.0 255.255.252.0
```

Verification

Note

Since devices in an OSPF area require the same copy of the database in order to compute correct SPF, filtering or summarization of routes in the database can only occur between areas or domains, not within an area. The below configuration illustrates how an Intra-Area Summary Network LSA (LSA 3) can be summarized as it is originated by an ABR.

Without any modification R1 sees two separate routes and LSAs to reach the networks 155.1.8.0/24 and 155.1.10.0/24. Since they are both originated by R5 (150.1.5.5), it implies that only R5 can modify R1's view of these. By configuring the **`area 3 range 155.1.8.0 255.255.252.0`** command R5 stops sending the specific LSAs from area 3 into area 0, and groups them into the single route 155.1.8.0/22.

```
Rack1R1#show ip route 155.1.8.8
Routing entry for 155.1.8.0/24
  Known via "ospf 1", distance 110, metric 3035, type inter area
  Last update from 155.1.0.5 on Serial0/0, 21:42:03 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.5.5, 21:42:03 ago, via Serial0/0
      Route metric is 3035, traffic share count is 1

Rack1R1#show ip route 155.1.10.10
Routing entry for 155.1.10.0/24
  Known via "ospf 1", distance 110, metric 3185, type inter area
  Last update from 155.1.0.5 on Serial0/0, 21:42:06 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.5.5, 21:42:06 ago, via Serial0/0
      Route metric is 3185, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#router ospf 1
Rack1R5(config-router)#area 3 range 155.1.8.0 255.255.252.0
Rack1R5(config-router)#

Rack1R1#show ip route 155.1.8.8
Routing entry for 155.1.8.0/22
  Known via "ospf 1", distance 110, metric 3035, type inter area
  Last update from 155.1.0.5 on Serial0/0, 00:00:05 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.5.5, 00:00:05 ago, via Serial0/0
      Route metric is 3035, traffic share count is 1

Rack1R1#show ip route 155.1.10.10
Routing entry for 155.1.8.0/22
  Known via "ospf 1", distance 110, metric 3035, type inter area
  Last update from 155.1.0.5 on Serial0/0, 00:00:07 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.5.5, 00:00:07 ago, via Serial0/0
      Route metric is 3035, traffic share count is 1
```

R5 now originates a single LSA 3 for 155.1.8.0/22.

```
Rack1R1#show ip ospf database summary 155.1.8.0

            OSPF Router with ID (150.1.1.1) (Process ID 1)

                Summary Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 1636
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 155.1.8.0 (summary Network Number)
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000029
  Checksum: 0xA905
  Length: 28
  Network Mask: /22
        TOS: 0  Metric: 35

<output omitted>
```

6.22 OSPF Path Selection with Summarization

- Configure R3 to originate the summary route 155.X.6.0/23 in order to force traffic destined to VLAN 7 to transit R6.
- If R6's connection to VLAN 146 is down, traffic for VLAN 7 should transit R3.

Configuration

```
R3:
router ospf 1
 area 2 range 155.1.6.0 255.255.254.0
```

Verification

Note

When a router does a routing lookup on a destination it always chooses the longest match route for the path. This means that if the router is trying to reach the destination 1.2.3.4 it will choose the route 1.2.3.0/24 over 1.2.0.0/18, or 1.2.0.0/18 over 0.0.0.0/0. This principle can be used for traffic engineering purposes by selectively summarizing prefixes into the IGP domain as seen below.

```
Rack1R5#show ip ospf database summary 155.1.7.0

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Summary Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 141
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 155.1.7.0 (summary Network Number)
  Advertising Router: 150.1.3.3
  LS Seq Number: 80000001
  Checksum: 0xBEF1
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 330
```

```
 Routing Bit Set on this LSA
  LS age: 12 (DoNotAge)
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 155.1.7.0 (summary Network Number)
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000001
  Checksum: 0x9713
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 330

<output omitted>
```

R5 learns the Summary Network LSA (LSA 3) route for 155.1.7.0/24 both from R3 and R6 with a metric of 330. Since R5's cost to R6 is higher than the cost to R3, R5 uses R3 as the exit point to this area 2 route.

```
Rack1R5#show ip route 155.1.7.7
Routing entry for 155.1.7.0/24
  Known via "ospf 1", distance 110, metric 3568, type inter area
  Last update from 155.1.0.3 on Serial0/0/0, 00:02:09 ago
  Routing Descriptor Blocks:
  * 155.1.0.3, from 150.1.3.3, 00:02:09 ago, via Serial0/0/0
      Route metric is 3568, traffic share count is 1
```

The path selection can be verified with a traceroute from area 3.

```
Rack1SW4#traceroute 155.1.7.7

Type escape sequence to abort.
Tracing the route to 155.1.7.7

  1 155.1.108.8 4 msec 4 msec 4 msec
  2 155.1.58.5 0 msec 4 msec 0 msec
  3 155.1.0.3 36 msec 36 msec 36 msec
  4 155.1.37.7 28 msec *  28 msec
```

Copyright © 2010 Internetwork Expert

By changing R3's origination of this route from 155.1.7.0/24 to 155.1.6.0/23, the longest match for all devices outside of area 2 is now 155.1.7.0/24 through R6.

```
Rack1R5#show ip ospf database summary 155.1.7.0

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Summary Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 12 (DoNotAge)
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 155.1.7.0 (summary Network Number)
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000001
  Checksum: 0x9713
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 330

<output omitted>

Rack1R5#show ip route 155.1.7.7
Routing entry for 155.1.7.0/24
  Known via "ospf 1", distance 110, metric 59223, type inter area
  Last update from 155.1.0.1 on Serial0/0/0, 00:00:17 ago
  Routing Descriptor Blocks:
  * 155.1.0.1, from 150.1.6.6, 00:00:17 ago, via Serial0/0/0
      Route metric is 59223, traffic share count is 1

Rack1SW4#traceroute 155.1.7.7

Type escape sequence to abort.
Tracing the route to 155.1.7.7

  1 155.1.108.8 4 msec 0 msec 4 msec
  2 155.1.58.5 0 msec 0 msec 4 msec
  3 155.1.0.1 28 msec 28 msec 28 msec
  4 155.1.146.6 28 msec 32 msec 28 msec
  5 155.1.67.7 32 msec *  28 msec
```

Copyright © 2010 Internetwork Expert

Due to the longest match routing principle, 155.1.6.0/23 can only be used to reach 155.1.7.7 if R6 withdraws 155.1.7.0/24.

```
Rack1R5#show ip ospf database summary 155.1.6.0

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Summary Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 81
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 155.1.6.0 (summary Network Number)
  Advertising Router: 150.1.3.3
  LS Seq Number: 80000001
  Checksum: 0xC4ED
  Length: 28
  Network Mask: /23
        TOS: 0  Metric: 330

Rack1R5#show ip route 155.1.7.7
Routing entry for 155.1.7.0/24
  Known via "ospf 1", distance 110, metric 59223, type inter area
  Last update from 155.1.0.1 on Serial0/0/0, 00:00:17 ago
  Routing Descriptor Blocks:
  * 155.1.0.1, from 150.1.6.6, 00:00:17 ago, via Serial0/0/0
      Route metric is 59223, traffic share count is 1

Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface FastEthernet0/0
Rack1R6(config-if)#shutdown
Rack1R6(config-if)#

Rack1R5#show ip route 155.1.7.7
Routing entry for 155.1.6.0/23
  Known via "ospf 1", distance 110, metric 3568, type inter area
  Last update from 155.1.0.3 on Serial0/0/0, 00:00:00 ago
  Routing Descriptor Blocks:
  * 155.1.0.3, from 150.1.3.3, 00:00:00 ago, via Serial0/0/0
      Route metric is 3568, traffic share count is 1

Rack1SW4#traceroute 155.1.7.7

Type escape sequence to abort.
Tracing the route to 155.1.7.7

  1 155.1.108.8 0 msec 4 msec 4 msec
  2 155.1.58.5 4 msec 4 msec 4 msec
  3 155.1.0.3 36 msec 40 msec 36 msec
  4 155.1.37.7 40 msec *  36 msec
```

Copyright © 2010 Internetwork Expert

6.23 OSPF External Summarization

- Redistribute between RIPv2 and OSPF on R4.
- Redistribute between EIGRP and OSPF on R6.
- R4 should advertise two summary routes that have a cost of 50 into OSPF for the prefixes learned from BB3.
- R6 should advertise a single summary route into OSPF for the prefixes learned from BB1 that has a cost of 100, and includes the cost needed to reach R6.

Configuration

```
R4:
router ospf 1
 router-id 150.1.4.4
 summary-address 30.0.0.0 255.252.0.0
 summary-address 31.0.0.0 255.252.0.0
 redistribute rip metric 50 subnets
!
router rip
 redistribute ospf 1 metric 1

R6:
router eigrp 10
 redistribute ospf 1 metric 10000 1000 255 1 1500
!
router ospf 1
 summary-address 200.0.0.0 255.255.252.0
 redistribute eigrp 10 metric 100 metric-type 1 subnets

SW4:
router ospf 1
 router-id 150.1.10.10
```

Copyright © 2010 Internetwork Expert

Verification

Pitfall

The first issue that should be evident with this section is that R4 and SW4 have the same OSPF router-id, due to a Loopback interface preconfigured with the IP address 223.255.255.255 from the initial configurations. Previously, we saw a problem where two routers in the same area had the same router-id, and the result was that SPF throughout that area failed. In this case the routers are in different areas, but one of them is originating external LSA type 5 advertisements through redistribution.

Since SW4 sees 223.255.255.255 as the router-id of the originator, which matches its own router-id, it thinks it is receiving its own advertisement back. Since SW4 is not originating these routes it sends a withdraw message back. The result can be seen from the following log messages on R4. The specifics behind this problem can be seen in section 13.4, "*Receiving self-originated LSAs*", of RFC 2328, *OSPF Version 2*.

```
R4#
%OSPF-4-FLOOD_WAR: Process 1 re-originates LSA ID 204.12.1.0 type-5
adv-rtr 223.255.255.255 in area 0
%OSPF-4-FLOOD_WAR: Process 1 re-originates LSA ID 31.2.0.0 type-5 adv-
rtr 223.255.255.255 in area 0
%OSPF-4-FLOOD_WAR: Process 1 re-originates LSA ID 31.3.0.0 type-5 adv-
rtr 223.255.255.255 in area 0
%OSPF-4-FLOOD_WAR: Process 1 re-originates LSA ID 204.12.1.0 type-5
adv-rtr 223.255.255.255 in area 0
%OSPF-4-FLOOD_WAR: Process 1 re-originates LSA ID 31.2.0.0 type-5 adv-
rtr 223.255.255.255 in area 0
%OSPF-4-FLOOD_WAR: Process 1 re-originates LSA ID 31.3.0.0 type-5 adv-
rtr 223.255.255.255 in area 0
%OSPF-4-FLOOD_WAR: Process 1 re-originates LSA ID 31.1.0.0 type-5 adv-
rtr
<output omitted>
```

Additionally, SW4 cannot install the external routes from R4 in the routing table or the OSPF database.

```
Rack1SW4#show ip route ospf | include E1|E2
O E2    51.51.51.51 [110/20] via 155.1.108.8, 01:41:14, Port-channel1
O E2 200.0.0.0/24 [110/20] via 155.1.108.8, 01:41:14, Port-channel1
O E2    54.1.1.0 [110/20] via 155.1.108.8, 01:41:14, Port-channel1
O E2 200.0.1.0/24 [110/20] via 155.1.108.8, 01:41:14, Port-channel1
O E2 200.0.2.0/24 [110/20] via 155.1.108.8, 01:41:14, Port-channel1
O E2 200.0.3.0/24 [110/20] via 155.1.108.8, 01:41:14, Port-channel1
```

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show ip ospf database | begin External
                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
51.51.51.51     192.10.1.254    26    (DNA) 0x80000001 0x009C6C 0
54.1.1.0        150.1.6.6       4     (DNA) 0x80000001 0x006E56 0
200.0.0.0       150.1.6.6       4     (DNA) 0x80000001 0x001420 0
200.0.1.0       150.1.6.6       4     (DNA) 0x80000001 0x00092A 0
200.0.2.0       150.1.6.6       4     (DNA) 0x80000001 0x00FD34 0
200.0.3.0       150.1.6.6       4     (DNA) 0x80000001 0x00F23E 0
```

To solve this problem R4 and SW4's router-ids should be set to be something unique.

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#router ospf 1
Rack1R4(config-router)#router-id 150.1.4.4
Reload or use "clear ip ospf process" command, for this to take effect
Rack1R4(config-router)#end
Rack1R4#clear ip ospf 1 process
Reset OSPF process? [no]: yes
Rack1R4#

Rack1SW4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW4(config)#router ospf 1
Rack1SW4(config-router)#router-id 150.1.10.10
Reload or use "clear ip ospf process" command, for this to take effect
Rack1SW4(config-router)#end
Rack1SW4#clear ip ospf 1 process
Reset OSPF process? [no]: yes
Rack1SW4#
```

Now SW4 can install all redistributed routes.

```
Rack1SW4#show ip route ospf | include E1|E2
O E2    51.51.51.51 [110/20] via 155.1.108.8, 00:31:05, Port-channel1
O E2 204.12.1.0/24 [110/50] via 155.1.108.8, 00:31:05, Port-channel1
O E1 200.0.0.0/24 [110/59173] via 155.1.108.8, 00:00:14, Port-channel1
O E1    54.1.1.0 [110/59173] via 155.1.108.8, 00:17:37, Port-channel1
O E1 200.0.1.0/24 [110/59173] via 155.1.108.8, 00:00:19, Port-channel1
O E1 200.0.2.0/24 [110/59173] via 155.1.108.8, 00:00:19, Port-channel1
O E1 200.0.3.0/24 [110/59173] via 155.1.108.8, 00:00:19, Port-channel1
O E2    31.3.0.0 [110/50] via 155.1.108.8, 00:00:00, Port-channel1
O E2    31.2.0.0 [110/50] via 155.1.108.8, 00:00:00, Port-channel1
O E2    31.1.0.0 [110/50] via 155.1.108.8, 00:00:00, Port-channel1
O E2    30.2.0.0 [110/50] via 155.1.108.8, 00:00:10, Port-channel1
O E2    30.3.0.0 [110/50] via 155.1.108.8, 00:00:10, Port-channel1
O E2    30.0.0.0 [110/50] via 155.1.108.8, 00:00:05, Port-channel1
O E2    30.1.0.0 [110/50] via 155.1.108.8, 00:00:10, Port-channel1
```

Copyright © 2010 Internetwork Expert

Note

External OSPF summarization is configured at the redistribution point between routing domains with the **summary-address** command. These summaries inherit their attributes from the subnets that make them up. For example a summary comprised of External Type-1 routes will result in an External Type-1 summary. This means that on R6 in this configuration the **metric-type 1** command is set at the time of redistribution instead of on the summary itself.

External Type-2 OSPF routes, which are the default, do not install the end-to-end metric in the routing table. Instead, only the metric that was reported via the ASBR is installed. The actual routing path is determined by the addition of the reported metric and the metric towards the ASBR, which is called the forward metric.

```
Rack1SW4#show ip ospf database external 30.0.0.0

            OSPF Router with ID (150.1.10.10) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 3 (DoNotAge)
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 30.0.0.0 (External Network Number )
  Advertising Router: 150.1.4.4
  LS Seq Number: 80000005
  Checksum: 0xF1D1
  Length: 36
  Network Mask: /14
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 50
        Forward Address: 0.0.0.0
        External Route Tag: 0
```

In the above output SW4 sees the summary 30.0.0.0/14 as an External Type-2 route originated by the ASBR 150.1.4.4. The *Forward Address: 0.0.0.0* field means that SW4 must now compute the metric towards the advertising router, R4, and install this metric in the routing table as the forward metric. Specifically this is calculated as follows.

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show ip ospf database router 150.1.10.10 self-originate

            OSPF Router with ID (150.1.10.10) (Process ID 1)

                Router Link States (Area 3)

  LS age: 1928
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.10.10
  Advertising Router: 150.1.10.10
  LS Seq Number: 80000004
  Checksum: 0x1384
  Length: 60
  Number of Links: 3

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.108.8
     (Link Data) Router Interface address: 155.1.108.10
      Number of TOS metrics: 0
       TOS 0 Metrics: 150

<output omitted>
```

SW4's metric to the DR on the link to SW2 is 150.

```
Rack1SW4#show ip ospf database router 150.1.8.8

            OSPF Router with ID (150.1.10.10) (Process ID 1)

                Router Link States (Area 3)

  LS age: 1 (DoNotAge)
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.8.8
  Advertising Router: 150.1.8.8
  LS Seq Number: 8000002F
  Checksum: 0xF09E
  Length: 72
  Number of Links: 4

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.108.8
     (Link Data) Router Interface address: 155.1.108.8
      Number of TOS metrics: 0
       TOS 0 Metrics: 150

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.58.8
     (Link Data) Router Interface address: 155.1.58.8
      Number of TOS metrics: 0
       TOS 0 Metrics: 30

<output omitted>
```

SW2's metric to the DR on the link to R5 is 30.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip ospf database router 150.1.5.5 self-originate

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Router Link States (Area 0)

  LS age: 1743
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.5.5
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000041
  Checksum: 0x2596
  Length: 108
  Area Border Router
  Number of Links: 7

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.4.4
     (Link Data) Router Interface address: 155.1.45.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 10000

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 155.1.45.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 10000

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.3.3
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 3238

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.1.1
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 58593

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.2.2
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 3238

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.4.4
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 39062

<output omitted>
```

Copyright © 2010 Internetwork Expert

R5's metric to R4 via the point-to-point Serial link is 10000. Therefore, SW4's total metric towards R4, the forward metric, is 10180.

The routing table output for the specific routing lookup should now show a metric of 50 to the destination, but a forward metric of 10180 towards the ASBR. This forward metric is what is used to determine the path towards the exit point.

```
Rack1SW4#show ip route | include E1|E2
       E1 - OSPF external type 1, E2 - OSPF external type 2
O E2    51.51.51.51 [110/20] via 155.1.108.8, 00:27:44, Port-channel1
O E2 204.12.1.0/24 [110/50] via 155.1.108.8, 00:27:44, Port-channel1
O E1    54.1.1.0 [110/59173] via 155.1.108.8, 00:14:17, Port-channel1
O E2    31.0.0.0 [110/50] via 155.1.108.8, 00:27:44, Port-channel1
O E2    30.0.0.0 [110/50] via 155.1.108.8, 00:27:44, Port-channel1
O E1 200.0.0.0/22 [110/59173] via 155.1.108.8, 00:14:17, Port-channel1

Rack1SW4#show ip route 30.0.0.1
Routing entry for 30.0.0.0/14
  Known via "ospf 1", distance 110, metric 50, type extern 2, forward
metric 10180
  Last update from 155.1.108.8 on Port-channel1, 00:27:49 ago
  Routing Descriptor Blocks:
  * 155.1.108.8, from 150.1.4.4, 00:27:49 ago, via Port-channel1
      Route metric is 50, traffic share count is 1
```

From the calculation of the forward metric we can predict that the traffic flow from SW4 to the final destination, 30.0.0.1, will be first sent to SW2, then to R5, over the point-to-point Serial link to R4, and finally out to the RIP domain via BB3.

```
Rack1SW4#traceroute 30.0.0.1

Type escape sequence to abort.
Tracing the route to 30.0.0.1

  1 155.1.108.8 4 msec 4 msec 4 msec
  2 155.1.58.5 0 msec 4 msec 0 msec
  3 155.1.45.4 16 msec 16 msec 12 msec
  4 204.12.1.254 28 msec *  24 msec
```

Copyright © 2010 Internetwork Expert

The calculation for External Type-1 OSPF routes does not distinguish in the routing table between the metric reported by the ASBR, and the metric to the ASBR via the forward metric. Instead, External Type-1 routes represent the metric as one cumulative value of the reported metric and the metric to the ASBR.

```
Rack1SW4#show ip route ospf | include E1
O E1    54.1.1.0 [110/59173] via 155.1.108.8, 00:56:00, Port-channel1
O E1 200.0.0.0/22 [110/59173] via 155.1.108.8, 00:02:03, Port-channel1

Rack1SW4#show ip route 200.0.0.1
Routing entry for 200.0.0.0/22, supernet
  Known via "ospf 1", distance 110, metric 59173, type extern 1
  Last update from 155.1.108.8 on Port-channel1, 00:03:53 ago
  Routing Descriptor Blocks:
  * 155.1.108.8, from 150.1.6.6, 00:03:53 ago, via Port-channel1
      Route metric is 59173, traffic share count is 1
```

The metric for this route is calculated as follows.

```
Rack1SW4#show ip ospf database external 200.0.0.0

            OSPF Router with ID (150.1.10.10) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 4 (DoNotAge)
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 200.0.0.0 (External Network Number )
  Advertising Router: 150.1.6.6
  LS Seq Number: 8000000E
  Checksum: 0x8ACF
  Length: 36
  Network Mask: /22
        Metric Type: 1 (Comparable directly to link state metric)
        TOS: 0
        Metric: 100
        Forward Address: 0.0.0.0
        External Route Tag: 0
```

SW4 sees that the ASBR 150.1.6.6 (R6) originated the route with a metric of 100. Now SW4 must find the closest path to forward toward the ASBR.

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show ip ospf database router 150.1.6.6

            OSPF Router with ID (150.1.10.10) (Process ID 1)
```

SW4 tries to find an Intra-Area path to the ASBR 150.1.6.6. Since R6 is not in the same area as SW4, SW4 must find which ABR it should use to exit towards R6.

```
Rack1SW4#show ip ospf database asbr-summary 150.1.6.6

            OSPF Router with ID (150.1.10.10) (Process ID 1)

                Summary ASB Link States (Area 3)

  Routing Bit Set on this LSA
  LS age: 2 (DoNotAge)
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(AS Boundary Router)
  Link State ID: 150.1.6.6 (AS Boundary Router address)
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000003
  Checksum: 0x53AD
  Length: 28
  Network Mask: /0
        TOS: 0  Metric: 58893
```

SW4's Intra-Area exit point towards R6 is R5. R5 has a metric of 58893 to reach R6. SW4 now runs SPF to reach R5.

```
Rack1SW4#show ip ospf database router 150.1.10.10 self-originate

            OSPF Router with ID (150.1.10.10) (Process ID 1)

                Router Link States (Area 3)

  LS age: 1111
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.10.10
  Advertising Router: 150.1.10.10
  LS Seq Number: 80000006
  Checksum: 0xF86
  Length: 60
  Number of Links: 3

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.108.8
     (Link Data) Router Interface address: 155.1.108.10
      Number of TOS metrics: 0
       TOS 0 Metrics: 150

<output omitted>
```

SW4's metric to the DR for the link to SW2 is 150.

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show ip ospf database router 150.1.8.8 self-originate

            OSPF Router with ID (150.1.8.8) (Process ID 1)

                Router Link States (Area 3)

  LS age: 1170
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.8.8
  Advertising Router: 150.1.8.8
  LS Seq Number: 80000035
  Checksum: 0xE4A4
  Length: 72
  Number of Links: 4

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.108.8
     (Link Data) Router Interface address: 155.1.108.8
      Number of TOS metrics: 0
       TOS 0 Metrics: 150

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.58.8
     (Link Data) Router Interface address: 155.1.58.8
      Number of TOS metrics: 0
       TOS 0 Metrics: 30

<output omitted>
```

SW2's metric to the DR on the link to R5 is 30. This means that SW4's metric to R5 is 150 + 30, R5's metric to R6 is 58893, and R6's metric to the external domain is 100, for a total of 59173. This is the value seen in the routing table as the final metric.

```
Rack1SW4#show ip route 200.0.0.1
Routing entry for 200.0.0.0/22, supernet
  Known via "ospf 1", distance 110, metric 59173, type extern 1
  Last update from 155.1.108.8 on Port-channel1, 00:15:10 ago
  Routing Descriptor Blocks:
  * 155.1.108.8, from 150.1.6.6, 00:15:10 ago, via Port-channel1
      Route metric is 59173, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

We can predict the traffic flow from SW4 to this destination by examining the Intra-Area routing of area 0. Since R6 is the ASBR originating the route, and R6 is in area 0, R5 will compute Intra-Area SPF to reach R6. Since R6 connects to area 0 via the virtual-link to R1, and R1 connects to R5 via the Frame Relay link, traffic must flow from SW4 to SW2, to R5, to R1 over the Frame Relay link, to R6 over VLAN 146, and finally out to BB1. Even if R5 has a lower cost to R6 via the point-to-point link to R4, and from R4 to VLAN 146, this path cannot be considered because it is Inter-Area, versus the Intra-Area path through the virtual-link. The result is as follows.

```
Rack1SW4#traceroute 200.0.0.1

Type escape sequence to abort.
Tracing the route to 200.0.0.1

  1 155.1.108.8 4 msec 4 msec 0 msec
  2 155.1.58.5 0 msec 4 msec 0 msec
  3 155.1.0.1 28 msec 32 msec 28 msec
  4 155.1.146.6 28 msec 28 msec 28 msec
  5 54.1.1.254 52 msec *  48 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Note

Revert all devices to the *Full OSPF Routing* initial configurations before continuing.

6.24 OSPF Stub Areas

- Configure OSPF area 3 so that R5 filters External routes out as they are sent from area 0 to area 3.
- Devices in area 3 should still have reachability to routes external to the OSPF domain.

Configuration

```
R5:
router ospf 1
 area 3 stub

SW2:
router ospf 1
 area 3 stub

SW4:
router ospf 1
 area 3 stub
```

Verification

Note

OSPF stub area types are used to filter information out of the OSPF database based on the role of the LSA. The stub flag is part of the OSPF adjacency formation, which implies that all devices in an area must agree on that parameter in order for adjacencies to establish. The four stub types that IOS supports are stub areas, totally stubby areas, not-so-stubby areas (NSSA), and not-so-totally-stubby areas. The first option, the stub area, is used to remove Type-5 External link states from the database, and replace them with a default route. The logic behind this feature stems from how external lookups between areas occur in OSPF.

When an OSPF router redistributes a route into the domain, it originates a Type-5 External LSA representing the route and its attributes. Inside this LSA the originating router sets the *advertising router* field to its local router-id, and generally, the *forward address* field to 0.0.0.0.

Copyright © 2010 Internetwork Expert

When an OSPF router in the same area as the originator looks up the Type-5 LSA, it looks at the forward address. If the forward address is set to 0.0.0.0, it means that the traffic should be sent towards the advertising router to reach the destination. To find out how to reach the advertising router, the advertising router's Type-1 Router LSA is consulted, and intra-area SPF is performed. This is similar to inter-area routing logic, because the router doing the lookup does not compute SPF to the final destination, only the intermediary advertising router. For external routing between areas, the logic is modified slightly.

When an Area Border Router receives a Type-5 External LSA from a device in its own area, and passes it into a different area, a Type-4 ASBR Summary LSA is generated. The Type-4 LSA tells devices in the new area how to forward towards the ASBR, which in turn tells them how to forward towards the external route. For example take the following situation in this topology. R4 redistributes the RIP route 30.1.0.0/24 into OSPF, originating a Type-5 External LSA.

```
Rack1R4#show ip ospf database external 30.1.0.0

            OSPF Router with ID (150.1.4.4) (Process ID 1)

                Type-5 AS External Link States

  LS age: 1651
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 30.1.0.0 (External Network Number )
  Advertising Router: 150.1.4.4
  LS Seq Number: 80000005
  Checksum: 0xC419
  Length: 36
  Network Mask: /16
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 0.0.0.0
        External Route Tag: 0
```

When R5 wants to reach the destination 30.1.0.0, it sees that the forward address is 0.0.0.0, and the advertising router is 150.1.4.4. R5 now does a Type-1 Router LSA lookup on 150.1.4.4 (R4).

```
Rack1R5#show ip ospf database router 150.1.4.4

<snip>

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.5.5
     (Link Data) Router Interface address: 155.1.45.4
      Number of TOS metrics: 0
       TOS 0 Metrics: 64

<snip>

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.5.5
     (Link Data) Router Interface address: 155.1.0.4
      Number of TOS metrics: 0
       TOS 0 Metrics: 64
```

Copyright © 2010 Internetwork Expert

From this R5 knows that R4 is directly adjacent via two paths, both with a metric of 64.

```
Rack1R5#show ip route 30.1.0.0
Routing entry for 30.1.0.0/16
  Known via "ospf 1", distance 110, metric 20, type extern 2, forward metric 64
  Last update from 155.1.0.4 on Serial0/0/0, 00:24:16 ago
  Routing Descriptor Blocks:
  * 155.1.45.4, from 150.1.4.4, 00:24:16 ago, via Serial0/1/0
      Route metric is 20, traffic share count is 1
    155.1.0.4, from 150.1.4.4, 00:24:16 ago, via Serial0/0/0
      Route metric is 20, traffic share count is 1
```

R5 installs both paths in the routing table with a metric of 20 from the Type-5 External LSA, and a forward metric of 64 to reach R4. If this route were redistributed as Type-1 External, as opposed to Type-2 External, the total metric would be 84 (the advertised metric plus the forward metric). Now R5 sends the Type-5 LSA from area 0 into area 3 to SW2.

```
Rack1SW2#show ip ospf database external 30.1.0.0

            OSPF Router with ID (150.1.8.8) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 902
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 30.1.0.0 (External Network Number )
  Advertising Router: 150.1.4.4
  LS Seq Number: 80000006
  Checksum: 0xC21A
  Length: 36
  Network Mask: /16
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 0.0.0.0
        External Route Tag: 0
```

SW2, like R5, sees the Forward Address for the route as 0.0.0.0, meaning a lookup on 150.1.4.4 must be performed. The difference here however, is that when SW2 looks for a Type-1 Router LSA, none is found.

```
Rack1SW2#show ip ospf database router 150.1.4.4

            OSPF Router with ID (150.1.8.8) (Process ID 1)
```

Copyright © 2010 Internetwork Expert

This essentially means that the advertising router (R4) for the external route is not in the same area as SW2. SW2 now checks to see which ABRs are advertising reachability information about R4.

```
Rack1SW2#show ip ospf database asbr-summary 150.1.4.4

            OSPF Router with ID (150.1.8.8) (Process ID 1)

                Summary ASB Link States (Area 3)

  Routing Bit Set on this LSA
  LS age: 1830
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(AS Boundary Router)
  Link State ID: 150.1.4.4 (AS Boundary Router address)
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000001
  Checksum: 0x6357
  Length: 28
  Network Mask: /0
        TOS: 0  Metric: 64
```

SW2 sees that 150.1.4.4 (R4) is known via the advertising router 150.1.5.5 (R5) with a metric of 64. An intra-area lookup is now performed on 150.1.5.5.

```
Rack1SW2#show ip ospf database router 150.1.5.5

            OSPF Router with ID (150.1.8.8) (Process ID 1)

                Router Link States (Area 3)

  Routing Bit Set on this LSA
  LS age: 12
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.5.5
  Advertising Router: 150.1.5.5
  LS Seq Number: 8000000C
  Checksum: 0xE8DA
  Length: 60
  Area Border Router
  Number of Links: 3

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 155.1.5.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.58.8
     (Link Data) Router Interface address: 155.1.58.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

<snip>
```

Copyright © 2010 Internetwork Expert

The recursion process continues until SW2 sees that it is adjacent with the DR 155.1.58.8 as well, meaning that R5 is reachable out Vlan58. Based on this SW2 knows that packets for 30.1.0.0 should go towards R5, who sends them towards R4, who in turn sends them to the final destination.

The key point about this external lookup between areas, however, is that for all external destinations outside of area 3, SW2 will see that R5 is the ABR that it must transit. This redundant information can be seen in the database view of area 3 as follows.

```
Rack1SW4#show ip ospf database | begin Type-5
                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
30.0.0.0        150.1.4.4       1864        0x80000001 0x00D80A 0
30.1.0.0        150.1.4.4       1864        0x80000001 0x00CC15 0
30.2.0.0        150.1.4.4       1864        0x80000001 0x00C020 0
30.3.0.0        150.1.4.4       1864        0x80000001 0x00B42B 0
31.0.0.0        150.1.4.4       1864        0x80000001 0x00CB16 0
31.1.0.0        150.1.4.4       1864        0x80000001 0x00BF21 0
31.2.0.0        150.1.4.4       1864        0x80000001 0x00B32C 0
31.3.0.0        150.1.4.4       1864        0x80000001 0x00A737 0
51.51.51.51     192.10.1.254    1226        0x80000002 0x009A6D 0
54.1.1.0        150.1.6.6       1878        0x80000001 0x006E56 0
200.0.0.0       150.1.6.6       1885        0x80000001 0x001420 0
200.0.1.0       150.1.6.6       1885        0x80000001 0x00092A 0
200.0.2.0       150.1.6.6       1885        0x80000001 0x00FD34 0
200.0.3.0       150.1.6.6       1885        0x80000001 0x00F23E 0
204.12.1.0      150.1.4.4       1870        0x80000001 0x005EC8 0
```

SW4 sees multiple Type-5 LSAs, most reachable via R4 and R6. Recursion for both R4 and R6 points to R5, as R5 is the only ABR servicing area 3. In a design such as this, stub areas can be used to optimize the OSPF database by replacing the redundant Type-5 External and Type-4 ASBR routing information with default information. In this particular case the changes seen are as follows.

```
Rack1R5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#router ospf 1
Rack1R5(config-router)#area 3 stub
Rack1R5(config-router)#end
Rack1R5#

Rack1SW2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW2(config)#router ospf 1
Rack1SW2(config-router)#area 3 stub
Rack1SW2(config-router)#end
Rack1SW2#
```

Copyright © 2010 Internetwork Expert

```
Rack1SW4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW4(config)#router ospf 1
Rack1SW4(config-router)#area 3 stub
Rack1SW4(config-router)#end
Rack1SW4#
```

With area 3 configured as a stub area, Type-5 External LSAs and Type-4 ASBR Summary LSAs no longer exist in the area 3 database.

```
Rack1SW4#show ip ospf database external 30.1.0.0

            OSPF Router with ID (150.1.10.10) (Process ID 1)

Rack1SW4#show ip ospf database asbr-summary 150.1.4.4

            OSPF Router with ID (150.1.10.10) (Process ID 1)
```

This implies that SW4 no longer has a specific route to 30.1.0.0, due to the deletion of the Type-4 and Type-5 LSAs.

```
Rack1SW4#show ip route 30.1.0.0
% Network not in table
```

What has been added to the database, however, is a default route as a Type-3 Summary LSA via the ABR.

```
Rack1SW4#show ip ospf database summary 0.0.0.0

            OSPF Router with ID (150.1.10.10) (Process ID 1)

                Summary Net Link States (Area 3)

  Routing Bit Set on this LSA
  LS age: 580
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 0.0.0.0 (summary Network Number)
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000002
  Checksum: 0x1B80
  Length: 28
  Network Mask: /0
        TOS: 0  Metric: 1
```

Copyright © 2010 Internetwork Expert

The result of this new default route is that area 3 maintains connectivity to the external routes by using the default route, but the size of the OSPF database and the routing table is much smaller.

```
Rack1SW4#show ip route | include _0.0.0.0
Gateway of last resort is 155.1.108.8 to network 0.0.0.0
O*IA 0.0.0.0/0 [110/3] via 155.1.108.8, 00:32:01, Port-channel1

Rack1SW4#traceroute 30.1.0.1

Type escape sequence to abort.
Tracing the route to 30.1.0.1

  1 155.1.108.8 4 msec 0 msec 4 msec
  2 155.1.58.5 0 msec 4 msec 0 msec
  3 155.1.45.4 16 msec
  4 204.12.1.254 24 msec *  24 msec
```

Copyright © 2010 Internetwork Expert

6.25 OSPF Totally Stubby Areas

- Configure OSPF area 3 so that R5 filters Inter-Area and External routes as they are sent from area 0 to area 3.
- Devices in area 3 should still have reachability to routes external to the OSPF domain.

Configuration

```
R5:
router ospf 1
 area 3 stub no-summary
```

Verification

> **Note**
>
> In the previous case with we saw that with area 3 converted to a stub area, the size of the routing table and OSPF database was reduced with no negative impact on connectivity. Specifically SW4's view of the topology was as follows.

```
Rack1SW4#show ip ospf database

            OSPF Router with ID (150.1.10.10) (Process ID 1)

                Router Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum Link count
150.1.5.5       150.1.5.5       807         0x8000000F 0x0001C1 3
150.1.8.8       150.1.8.8       464         0x80000014 0x00165C 4
150.1.10.10     150.1.10.10     819         0x8000000F 0x00969E 3

                Net Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.58.8      150.1.8.8       715         0x8000000D 0x00F06F
155.1.108.10    150.1.10.10     819         0x80000009 0x001A08

                Summary Net Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
0.0.0.0         150.1.5.5       41          0x80000003 0x001981
150.1.1.1       150.1.5.5       12          0x80000001 0x00D8E9
150.1.2.2       150.1.5.5       12          0x80000001 0x00C3FC
150.1.3.3       150.1.5.5       12          0x80000001 0x00AE10
150.1.4.4       150.1.5.5       12          0x80000001 0x009923
150.1.6.6       150.1.5.5       12          0x80000001 0x00793E
150.1.7.7       150.1.5.5       12          0x80000001 0x006451
150.1.9.9       150.1.5.5       12          0x80000001 0x00446C
155.1.0.1       150.1.5.5       13          0x80000001 0x009827
155.1.0.2       150.1.5.5       13          0x80000001 0x008E30
155.1.0.3       150.1.5.5       13          0x80000001 0x008439
155.1.0.4       150.1.5.5       13          0x80000001 0x007A42
155.1.0.5       150.1.5.5       13          0x80000001 0x00ED0E
155.1.7.0       150.1.5.5       13          0x80000001 0x00694E
```

Copyright © 2010 Internetwork Expert

```
155.1.9.0       150.1.5.5       13          0x80000001 0x005D57
155.1.13.0      150.1.5.5       13          0x80000001 0x0095DD
155.1.23.0      150.1.5.5       13          0x80000001 0x002742
155.1.37.0      150.1.5.5       13          0x80000001 0x001486
155.1.45.0      150.1.5.5       13          0x80000001 0x00B1E1
155.1.67.0      150.1.5.5       13          0x80000001 0x00D2A8
155.1.79.0      150.1.5.5       13          0x80000001 0x004E21
155.1.146.0     150.1.5.5       13          0x80000001 0x0060CC
192.10.1.0      150.1.5.5       13          0x80000001 0x00523E
```

The above output indicates that the intra-area information from Type-1 Router LSAs and Type-2 Network LSAs still exist, along with the inter-area Type-3 Summary LSA information, but Type-4 ASBR Summary LSAs and Type-5 External LSAs have been removed.

Recall that in the previous case for external routes we saw that every Type-4 ASBR Summary LSA inside of area 3 always recursed back to R5, since R5 was the only ABR connecting area 3 to area 0. By configuring area 3 as stub, this information was replaced with a default route which recursed to R5. The same connectivity resulted, but space was saved in the routing table and OSPF database. The next logical step in further optimizing the database then is to summarize the redundant Type-3 Summary LSAs that represent the inter-area routes. This is where configuring the area as totally stubby can be advantageous.

Where a stub area optimizes the database by removing external routes, and replacing it with a default route, a totally stubby area will optimize the database further by removing the inter-area and external routes, replacing them both with a default route. This is accomplished by telling the area border router not to inject Type-3 Summary Network LSAs from area 0, hence the **no-summary** argument used in conjunction with the **stub** command.

Note that only the ABR(s) connecting the stub area to area 0 need the **no-summary** argument on the stub command, since they are the only devices that are allowed to originate Type-3 LSAs. While it won't "break" the OSPF design to add the command to other routers inside the totally stubby area, it is technically incorrect to configure the option this way. As long as all devices in the area agree on the stub flag in the first place, it is the ABR(s) duty to determine if the area is totally stubby or not.

Copyright © 2010 Internetwork Expert

To verify the operation of the totally stubby area, view the changes to the database on the area 3 routers.

```
Rack1SW4#show ip ospf database

            OSPF Router with ID (150.1.10.10) (Process ID 1)

                Router Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum Link count
150.1.5.5       150.1.5.5       168         0x80000010 0x00FEC2 3
150.1.8.8       150.1.8.8       1854        0x80000014 0x00165C 4
150.1.10.10     150.1.10.10     184         0x80000010 0x00949F 3

                Net Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.58.8      150.1.8.8       86          0x8000000E 0x00EE70
155.1.108.10    150.1.10.10     185         0x8000000A 0x001809

                Summary Net Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
0.0.0.0         150.1.5.5       170         0x80000004 0x001782
```

Note the major change that has occurred is that SW4 no longer has specific inter-area routing information listed under the *Summary Net Link States* field, which represents the Type-3 Summary LSAs. Since SW4 still has a default route via R5, connectivity is not affected by this change.

```
Rack1SW4#show ip route 155.1.9.9
% Subnet not in table

Rack1SW4#show ip route | include _0.0.0.0
Gateway of last resort is 155.1.108.8 to network 0.0.0.0
O*IA 0.0.0.0/0 [110/3] via 155.1.108.8, 00:05:36, Port-channel1

Rack1SW4#traceroute 155.1.9.9

Type escape sequence to abort.
Tracing the route to 155.1.9.9

  1 155.1.108.8 4 msec 0 msec 0 msec
  2 155.1.58.5 0 msec 4 msec 0 msec
  3 155.1.0.3 28 msec 32 msec 32 msec
  4 155.1.37.7 24 msec 24 msec 20 msec
  5 155.1.79.9 32 msec *  28 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

6.26 OSPF Not-So-Stubby Areas

- Shutdown R6's Loopback 0 interface and connection to VLAN 146.
- Configure OSPF area 2 so that R3 filters External routes out as they are sent from area 0 to area 2.
- R6 should still be allowed to redistribute EIGRP into OSPF.
- Note any reachability problems throughout the domain.

Configuration

```
R3:
router ospf 1
 area 2 nssa

R6:
interface Loopback0
 shutdown
!
interface FastEthernet0/0.146
 shutdown
!
router ospf 1
 area 2 nssa

SW1:
router ospf 1
 area 2 nssa

SW3:
router ospf 1
 area 2 nssa
```

Verification

Note

The *OSPF Not-So-Stubby Area (NSSA) Option*, as defined in RFC 3101, extends the functionality of a stub area to allow the importing of a subset of external routes into the area. Recall that with the stub area, Type-5 External LSA information is suppressed from entering the database, and is replaced with a default route originated by the ABR(s). Since all Type-5 LSAs are suppressed, this also implies that redistribution cannot occur within the area as well. This problem can be seen from the parser error generated when redistribution and stub areas are configured together, as follows.

Copyright © 2010 Internetwork Expert

```
Rack1SW4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW4(config)#router ospf 1
Rack1SW4(config-router)#redistribute connected subnets
OSPF-4-ASBR_WITHOUT_VALID_AREA: Router is currently an ASBR while having only
one area which is a stub area
```

The OSPF NSSA option changes this behavior by allowing redistribution to occur *within* the stub area, while still blocking external routes from entering the area through the ABR(s). Specifically, this is implemented through the introduction of a new link-state advertisement type, the Type-7 NSSA External LSA.

Routes that are redistributed directly into the NSSA are generated as Type-7 NSSA External LSAs. Like Type-5 External LSAs, two subtypes of Type-7 NSSA External LSAs exist, type 1 (N1) and type 2 (N2). N1, similar to E1, considers the metric that the ASBR reports into the OSPF domain along with the metric needed to reach the ABSR. N2, similar to E2, separates the metric into the flat value that the ASBR reports into the OSPF domain, which is installed in the routing table, and the value needed to reach the ASBR, known as the forward metric.

From the below output we can see that with the default redistribution values, R6 originates the Type-7 NSSA External LSAs as metric-type 2, with a metric value of 20. The detailed output from SW1's routing table indicates a metric of 20 reported in by R6, and a forward metric of 1000 – SW1's metric to reach R6.

```
Rack1R6#show run | section router ospf
router ospf 1
 router-id 150.1.6.6
 area 1 virtual-link 150.1.1.1
 area 2 nssa
 redistribute eigrp 10 subnets

Rack1SW1#show ip route | include N1|N2
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
O N2 200.0.0.0/24 [110/20] via 155.1.67.6, 07:10:50, Vlan67
O N2    54.1.1.0  [110/20] via 155.1.67.6, 07:10:50, Vlan67
O N2 200.0.1.0/24 [110/20] via 155.1.67.6, 07:10:50, Vlan67
O N2 200.0.2.0/24 [110/20] via 155.1.67.6, 07:10:50, Vlan67
O N2 200.0.3.0/24 [110/20] via 155.1.67.6, 07:10:50, Vlan67

Rack1SW1#show ip route 200.0.0.0
Routing entry for 200.0.0.0/24
  Known via "ospf 1", distance 110, metric 20, type NSSA extern 2, forward
metric 1000
  Last update from 155.1.67.6 on Vlan67, 07:10:58 ago
  Routing Descriptor Blocks:
  * 155.1.67.6, from 150.1.6.6, 07:10:58 ago, via Vlan67
      Route metric is 20, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

When the Type-7 NSSA External LSA is received by the ABR, and is moved into area 0, the information contained in the Type-7 LSA is translated to a normal Type-5 External LSA. If multiple ABRs exist, only one of them performs the translation through an election process, which is discussed in depth in a later task. In this fashion OSPF devices outside of the NSSA do not know that the NSSA exists, which is analogous to how a Confederation works in BGP.

In this particular case R3 learns the Type-7 NSSA External LSAs that are originated by R6.

```
Rack1R3#show ip ospf database | begin Type-7
                Type-7 AS External Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum Tag
54.1.1.0        150.1.6.6       129         0x80000001 0x005480 0
200.0.0.0       150.1.6.6       129         0x80000001 0x00F94A 0
200.0.1.0       150.1.6.6       129         0x80000001 0x00EE54 0
200.0.2.0       150.1.6.6       129         0x80000001 0x00E35E 0
200.0.3.0       150.1.6.6       129         0x80000001 0x00D868 0
<snip>
```

The detailed information contained in the LSA can be viewed with the **show ip ospf database nssa-external [*Link ID*]** command.

```
Rack1R3#show ip ospf database nssa-external 200.0.0.0

            OSPF Router with ID (150.1.3.3) (Process ID 1)

                Type-7 AS External Link States (Area 2)

  Routing Bit Set on this LSA
  LS age: 312
  Options: (No TOS-capability, Type 7/5 translation, DC)
  LS Type: AS External Link
  Link State ID: 200.0.0.0 (External Network Number )
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000001
  Checksum: 0xF94A
  Length: 36
  Network Mask: /24
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 155.1.67.6
        External Route Tag: 0
```

Copyright © 2010 Internetwork Expert

Note that R3 receives the Type-7 NSSA External LSA with the forward address set to 155.1.67.6, which happens to be R6's interface pointing into area 2. With the previous Type-5 external lookups we saw the forward address set to 0.0.0.0, which meant to route towards the advertising router to reach the final destination. In this case, the forward address is non-zero, which causes the lookup to be performed towards 155.1.67.6. This is a subtle difference in the lookup process, and this particular case results in the same path selection even if the lookup had occurred on the advertising router (150.1.6.6) instead of the forward address (155.1.67.6). There can however be certain designs where there is a shorter path to the forward address than the advertising router's address, which is explored in a later task related to multiple exit points out of the NSSA.

The result of the translation on R3 is that devices in area 0 see the routes as Type-5 External LSAs, not Type-7.

```
Rack1R5#show ip ospf database | begin Type-5
                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
30.0.0.0        150.1.4.4       982         0x80000001 0x00D80A 0
30.1.0.0        150.1.4.4       982         0x80000001 0x00CC15 0
30.2.0.0        150.1.4.4       982         0x80000001 0x00C020 0
30.3.0.0        150.1.4.4       982         0x80000001 0x00B42B 0
31.0.0.0        150.1.4.4       982         0x80000001 0x00CB16 0
31.1.0.0        150.1.4.4       982         0x80000001 0x00BF21 0
31.2.0.0        150.1.4.4       982         0x80000001 0x00B32C 0
31.3.0.0        150.1.4.4       982         0x80000001 0x00A737 0
51.51.51.51     192.10.1.254    1334        0x8000012D 0x00419A 0
54.1.1.0        150.1.3.3       781         0x80000001 0x0010D4 0
200.0.0.0       150.1.3.3       779         0x80000001 0x00B59E 0
200.0.1.0       150.1.3.3       779         0x80000001 0x00AAA8 0
200.0.2.0       150.1.3.3       780         0x80000001 0x009FB2 0
200.0.3.0       150.1.3.3       780         0x80000001 0x0094BC 0
204.12.1.0      150.1.4.4       984         0x80000001 0x005EC8 0
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip ospf database external 200.0.0.0

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 173
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 200.0.0.0 (External Network Number )
  Advertising Router: 150.1.3.3
  LS Seq Number: 80000002
  Checksum: 0xB39F
  Length: 36
  Network Mask: /24
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 155.1.67.6
        External Route Tag: 0
```

R5 performs a lookup on the now Type-5 External LSA, and like R3, sees the forward address set to 155.1.67.6. Again, note that the lookup process for this translated Type-7 LSA is performed differently than a normal inter-area Type-5 External LSA lookup, because R5 computes its metric towards 155.1.67.6, and not a Type-4 LSA describing the ASBR. Furthermore, note that R3 does not generate a Type-4 ASBR Summary LSA describing R6.

```
Rack1R5#show ip ospf database asbr-summary 150.1.6.6

            OSPF Router with ID (150.1.5.5) (Process ID 1)

Rack1R5#show ip route 155.1.67.6
Routing entry for 155.1.67.0/24
  Known via "ospf 1", distance 110, metric 1065, type inter area
  Last update from 155.1.0.3 on Serial0/0/0, 00:35:16 ago
  Routing Descriptor Blocks:
  * 155.1.0.3, from 150.1.3.3, 00:35:16 ago, via Serial0/0/0
      Route metric is 1065, traffic share count is 1
```

R5's metric to the forwarding address 155.1.67.6 is 1065 via R3. This is the value installed as the forward metric for the translated Type-7 LSA, with a metric of 20 from the Type-5 LSA itself.

```
Rack1R5#show ip route 200.0.0.0
Routing entry for 200.0.0.0/24
  Known via "ospf 1", distance 110, metric 20, type extern 2, forward metric
1065
  Last update from 155.1.0.3 on Serial0/0/0, 00:35:51 ago
  Routing Descriptor Blocks:
  * 155.1.0.3, from 150.1.3.3, 00:35:51 ago, via Serial0/0/0
      Route metric is 20, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

Similar to the stub area, the NSSA flag must be agreed upon by all devices in the area, or adjacency cannot occur. This implies that the area is a normal area, a stub area, or an NSSA, but no combination of the three.

Furthermore, like the stub area, Type-5 External LSAs are blocked from entering the NSSA area on the ABR(s).

```
Rack1SW1#show ip route | include E1|E2
       E1 - OSPF external type 1, E2 - OSPF external type 2

Rack1SW1#show ip ospf database | begin Type-5
```

Pitfall

The other key difference between stub and NSSA areas is how default routing works. The stub area removes external LSAs and replaces them with a default route. The totally stubby area extends this by replacing external LSAs and inter-area LSAs with a default route. However *with the NSSA a default route is not automatically originated by the ABR*. This means that devices within the NSSA will have reachability to their own area and to other areas, but not to destinations outside of the OSPF domain.

```
Rack1R3#show ip route 30.0.0.1
Routing entry for 30.0.0.0/16
  Known via "ospf 1", distance 110, metric 20, type extern 2, forward metric
845
  Last update from 155.1.0.5 on Serial1/0, 00:02:09 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.4.4, 00:02:09 ago, via Serial1/0
      Route metric is 20, traffic share count is 1

Rack1R3#ping 30.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 104/113/128 ms

Rack1SW1#show ip route 30.0.0.1
% Network not in table

Rack1SW1#show ip route | include Gateway
Gateway of last resort is not set

Rack1SW1#ping 30.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
```

Copyright © 2010 Internetwork Expert

6.27 OSPF Not-So-Stubby Areas and Default Routing

- Configure R3 to advertise a default route into area 2.
- This default route should have a cost of 500.

Configuration

```
R3:
router ospf 1
 area 2 nssa default-information-originate
 area 2 default-cost 500
```

Verification

> **Note**
>
> Unlike the stub area, totally-stubby area, or not-so-totally-stubby area, the ABR(s) of an NSSA do not automatically originate a default route. A default route can be originated as a Type-7 NSSA External LSA into the NSSA by adding the **`default-information-originate`** option onto the **`area [id] nssa`** statement. The cost that the ABR advertises for the default can be modified with the **`area [id] default-cost`** command.

```
Rack1SW1#show ip route | include _0.0.0.0
Gateway of last resort is 155.1.37.3 to network 0.0.0.0
O*N2 0.0.0.0/0 [110/500] via 155.1.37.3, 00:00:17, FastEthernet0/3

Rack1SW1#show ip ospf database | begin Type-7

                Type-7 AS External Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum Tag
0.0.0.0         150.1.3.3       31          0x80000001 0x00EB6E 0
54.1.1.0        150.1.6.6       56          0x80000010 0x00368F 0
200.0.0.0       150.1.6.6       57          0x80000010 0x00DB59 0
200.0.1.0       150.1.6.6       57          0x80000010 0x00D063 0
200.0.2.0       150.1.6.6       57          0x80000010 0x00C56D 0
200.0.3.0       150.1.6.6       57          0x80000010 0x00BA77 0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

6.28 OSPF Not-So-Totally-Stubby Areas

- Remove the previous default routing configuration on R3.
- Modify the area 2 configuration so that R3 filters Inter-Area and External routes as they are sent from area 0 to area 2.
- R6 should still be allowed to redistribute EIGRP into OSPF.

Configuration

```
R3:
router ospf 1
 no area 2 nssa
 area 2 nssa no-summary
```

Verification

> **Note**
>
> The not-so-totally-stubby area is the combination of the totally-stubby area and the NSSA. Like the totally-stubby area, Type-3 Summary LSAs, Type-4 ASBR Summary LSAs, and Type-5 External LSAs are removed and replaced with a Type-3 Summary LSA default route. Like the NSSA, Type-7 NSSA External LSAs are allowed to be originated inside the area.
>
> The combination of these two result in the blocking of all inter-area OSPF routes & routes external to the OSPF domain, replacing them with a default route, and allowing redistribution to occur. SW3's routing table output indicates this because only intra-area, NSSA external, and an inter-area default route are installed. Note that the cost of the default route is derived from the intra-area cost to R3, plus the **`area 2 default-cost 500`** previously configured on R3.

```
Rack1SW3#show ip route ospf
     155.1.0.0/24 is subnetted, 5 subnets
O       155.1.7.0 [110/2] via 155.1.79.7, 00:00:53, Vlan79
O       155.1.37.0 [110/2] via 155.1.79.7, 00:00:53, Vlan79
O       155.1.67.0 [110/1001] via 155.1.79.7, 00:00:53, Vlan79
O N2 200.0.0.0/24 [110/20] via 155.1.79.7, 00:00:53, Vlan79
     54.0.0.0/24 is subnetted, 1 subnets
O N2    54.1.1.0 [110/20] via 155.1.79.7, 00:00:53, Vlan79
O N2 200.0.1.0/24 [110/20] via 155.1.79.7, 00:00:53, Vlan79
O N2 200.0.2.0/24 [110/20] via 155.1.79.7, 00:00:53, Vlan79
O N2 200.0.3.0/24 [110/20] via 155.1.79.7, 00:00:54, Vlan79
     150.1.0.0/16 is variably subnetted, 2 subnets, 2 masks
O       150.1.7.7/32 [110/2] via 155.1.79.7, 00:00:54, Vlan79
O*IA 0.0.0.0/0 [110/502] via 155.1.79.7, 00:00:54, Vlan79
```

Copyright © 2010 Internetwork Expert

The database output indicates that the only Type-3 Summary LSA is the default route originated by the ABR.

```
Rack1SW3#show ip ospf database

            OSPF Router with ID (150.1.9.9) (Process ID 1)

                Router Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum Link
count
150.1.3.3       150.1.3.3       512         0x80000015 0x00AD9F 1
150.1.6.6       150.1.6.6       1213        0x80000014 0x0007FC 1
150.1.7.7       150.1.7.7       60          0x8000001A 0x007D8D 5
150.1.9.9       150.1.9.9       59          0x80000015 0x004327 3

                Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.37.7      150.1.7.7       1235        0x80000012 0x00244D
155.1.67.7      150.1.7.7       1236        0x80000012 0x002A23
155.1.79.9      150.1.9.9       60          0x80000001 0x00DA6B

                Summary Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
0.0.0.0         150.1.3.3       566         0x80000001 0x004F54

                Type-7 AS External Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum Tag
54.1.1.0        150.1.6.6       1214        0x80000010 0x00368F 0
200.0.0.0       150.1.6.6       1215        0x80000010 0x00DB59 0
200.0.1.0       150.1.6.6       1215        0x80000010 0x00D063 0
200.0.2.0       150.1.6.6       1215        0x80000010 0x00C56D 0
200.0.3.0       150.1.6.6       1215        0x80000010 0x00BA77 0
```

SW3 does not have a longer match to 30.0.0.1, but can use its default information to reach it.

```
Rack1SW3#show ip route 30.0.0.1
% Network not in table

Rack1SW3#ping 30.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 92/104/120 ms
```

Copyright © 2010 Internetwork Expert

6.29 OSPF Stub Areas with Multiple Exit Points

- Re-enable R6's Loopback 0 interface and connection to VLAN 146.
- Modify area 2 in such a way that traffic from devices in area 2 going to area 3 uses R6 as the exit point, while traffic from area 2 going to the RIP routes learned from BB3 uses R3.
- If R3 loses its connection to area 2, traffic for External destinations should be rerouted to R6.
- If R6 loses its connection to area 2, traffic for Inter-Area destinations should be rerouted to R3.
- Do not modify the cost of any links in area 2 to accomplish this.

Configuration

```
R3:
router ospf 1
 area 2 nssa no-summary

R6:
router ospf 1
 area 2 nssa default-information-originate

SW1:
router ospf 1
 area 2 nssa

SW3:
router ospf 1
 area 2 nssa
```

Verification

Note

In addition to database filtering, stub areas can be used for inter-area traffic engineering. In this particular case there are multiple exit points out of area 2, R3 and R6. R3 advertises only a default route as a Type-3 Summary LSA, due to its not-so-totally-stubby configuration. R6 advertises all Type-3 Summary LSAs, plus a Type-7 NSSA External default route. For inter-area routing from devices in area 2 this means that the longest match learned from R6 will always be used, and for default routing the Type-3 default will be used from R3. The default preference through R3 occurs since OSPF always prefers routes in the sequence intra-area > inter-area > external > nssa-external.

Copyright © 2010 Internetwork Expert

The result of this can be seen on SW1 in the database and the routing table.

```
Rack1SW1#show ip ospf database | begin Summary

                Summary Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
0.0.0.0         150.1.3.3       1457        0x80000002 0x004D55
150.1.1.1       150.1.6.6       323         0x80000002 0x00D81E
150.1.2.2       150.1.6.6       323         0x80000002 0x00C8AB
150.1.3.3       150.1.6.6       324         0x80000002 0x00B3BE
150.1.4.4       150.1.6.6       324         0x80000003 0x009758
150.1.5.5       150.1.6.6       324         0x80000002 0x0007A7
150.1.6.6       150.1.6.6       324         0x80000002 0x006588
150.1.8.8       150.1.6.6       324         0x80000002 0x00D1D5
150.1.10.10     150.1.6.6       324         0x80000002 0x00B1F0
<snip>
```

The only inter-area route learned from R3 is a default route. This means traffic to the inter-area destination 150.1.10.10 must use R6 as the exit point.

```
Rack1SW1#show ip route 150.1.10.10
Routing entry for 150.1.10.10/32
  Known via "ospf 1", distance 110, metric 1068, type inter area
  Last update from 155.1.67.6 on Vlan67, 00:39:46 ago
  Routing Descriptor Blocks:
  * 155.1.67.6, from 150.1.6.6, 00:39:46 ago, via Vlan67
      Route metric is 1068, traffic share count is 1

Rack1SW1#traceroute 150.1.10.10

Type escape sequence to abort.
Tracing the route to 150.1.10.10

  1 155.1.67.6 0 msec 0 msec 8 msec
  2 155.1.146.4 0 msec 8 msec 0 msec
  3 155.1.0.5 34 msec 34 msec 25 msec
  4 155.1.58.8 33 msec 34 msec 33 msec
  5 155.1.108.10 34 msec *  25 msec
```

SW1 does not have a longer match for the destination 30.0.0.1, but it does have default information through R3.

```
Rack1SW1#show ip route 30.0.0.1
% Network not in table

Rack1SW1#show ip route | include _0.0.0.0
Gateway of last resort is 155.1.37.3 to network 0.0.0.0
O*IA 0.0.0.0/0 [110/501] via 155.1.37.3, 00:40:44, FastEthernet0/3
```

Copyright © 2010 Internetwork Expert

The result is that for routes outside of the OSPF domain, such as the RIP routes redistributed into OSPF on R4, R3 is used as the exit point.

```
Rack1SW1#traceroute 30.0.0.1

Type escape sequence to abort.
Tracing the route to 30.0.0.1

  1 155.1.37.3 0 msec 0 msec 0 msec
  2 155.1.0.5 26 msec 33 msec 34 msec
  3 155.1.45.4 17 msec
  4 204.12.1.254 26 msec *  92 msec
```

Note that in the database SW1 has two default routes, one from R3 as a Type-3 Summary LSA, and one from R6 as a Type-7 NSSA External route. Even though the metric of the NSSA route is lower than the Type-3 route, the Type-3 route via R3 is always preferred.

```
Rack1SW1#show ip ospf database summary 0.0.0.0

            OSPF Router with ID (150.1.7.7) (Process ID 1)

                Summary Net Link States (Area 2)

  Routing Bit Set on this LSA
  LS age: 1619
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 0.0.0.0 (summary Network Number)
  Advertising Router: 150.1.3.3
  LS Seq Number: 80000002
  Checksum: 0x4D55
  Length: 28
  Network Mask: /0
        TOS: 0  Metric: 500
```

```
Rack1SW1#show ip ospf database nssa-external 0.0.0.0

            OSPF Router with ID (150.1.7.7) (Process ID 1)

                Type-7 AS External Link States (Area 2)

  LS age: 493
  Options: (No TOS-capability, No Type 7/5 translation, DC)
  LS Type: AS External Link
  Link State ID: 0.0.0.0 (External Network Number )
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000002
  Checksum: 0xE343
  Length: 36
  Network Mask: /0
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 1
        Forward Address: 155.1.67.6
        External Route Tag: 0
```

Only once R3 withdraws its inter-area default route can R6 be used as the default exit point.

```
Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#interface Fa0/0
Rack1R3(config-if)#shutdown
Rack1R3(config-if)#

Rack1SW1#show ip route | include _0.0.0.0
Gateway of last resort is 155.1.67.6 to network 0.0.0.0
O*N2 0.0.0.0/0 [110/1] via 155.1.67.6, 00:00:00, Vlan67

Rack1SW1#traceroute 30.0.0.1

Type escape sequence to abort.
Tracing the route to 30.0.0.1

  1 155.1.67.6 9 msec 0 msec 0 msec
  2 155.1.146.4 8 msec 0 msec 0 msec
  3 204.12.1.254 8 msec *  0 msec
```

Copyright © 2010 Internetwork Expert

Likewise if R6 loses connectivity to area 0, inter-area destinations are rerouted to R3 since there isn't a longer match than the default route.

```
Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#interface Fa0/0
Rack1R3(config-if)#no shutdown
Rack1R3(config-if)#

Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface Fa0/0.146
Rack1R6(config-subif)#shutdown
Rack1R6(config-subif)#

Rack1SW1#show ip route 150.1.10.10
% Subnet not in table

Rack1SW1#show ip route | include _0.0.0.0
Gateway of last resort is 155.1.37.3 to network 0.0.0.0
O*IA 0.0.0.0/0 [110/501] via 155.1.37.3, 00:00:05, FastEthernet0/3

Rack1SW1#traceroute 150.1.10.10

Type escape sequence to abort.
Tracing the route to 150.1.10.10

  1 155.1.37.3 0 msec 0 msec 9 msec
  2 155.1.0.5 25 msec 34 msec 33 msec
  3 155.1.58.8 34 msec 33 msec 34 msec
  4 155.1.108.10 33 msec *  26 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

6.30 OSPF NSSA Type-7 to Type-5 Translator Election

- Create a new Loopback 9 interface on SW3 with the IP address 9.9.9.9/32, and redistribute it into OSPF.
- Modify the OSPF domain so that only R3 advertises this route into area 0, but traffic from R5 going to this destination transits R6.
- If R6 is down this traffic should be rerouted to R3.

Configuration

```
R3:
interface FastEthernet0/0
 ip ospf cost 1000
!
router ospf 1
 router-id 150.1.30.30

SW3:
interface Loopback9
 ip address 9.9.9.9 255.255.255.255
!
router ospf 1
 redistribute connected subnets
```

Verification

Note

As previously seen with OSPF Not-So-Stubby Areas, Type-7 NSSA External LSAs are translated to Type-5 External LSAs by the ABR connecting the NSSA to area 0. When multiple ABRs connect the NSSA to area 0, the ABR with the highest router-id is elected as the Type-7 to 5 translator, and is responsible for re-originating the Type-5 LSA into area 0. This election process is an optimization of the OSPF database, and relates to how the Type-7 NSSA External route uses the forward address field to ensure optimal routing.

Recall that with normal external routes, only one Type-5 LSA is originated by the router performing the redistribution. When the route moves between areas, each ABR originates a Type-4 ASBR Summary LSA advertising their reachability to the ASBR. This means that for all Type-5 External LSA inter-area lookups OSPF would require *Ext_Routes* + *Num_ABRs* + *Num_Routers* LSAs, where *Ext_Routes* is the number of Type-5 LSAs, *Num_ABRs* is the number of ABRs generating Type-4 ASBR summaries, and *Num_Routers* is the number of Type-1 LSAs from the routers in the local area.

Copyright © 2010 Internetwork Expert

Now with Type-7 LSAs the situation becomes more complicated, because this information needs to be re-originated at the ABR level as the route moves into area 0. Let's suppose for the sake of argument that each ABR connecting the NSSA to area 0 *did* do a translation of Type-7 to 5. This would mean for all inter-area lookups on a Type-5 External LSAs that were translated from Type-7, there would be (*NSSA_Routes* * *Num_ABRs*) + *Num_ABRs* + *Num_Routers* LSAs, where *NSSA_Routes* is the number of Type-7 LSAs to start.

This operation would be highly redundant and inefficient, because each ABR would re-originate the same Type-5 LSA, each with the same forwarding address. To avoid this only one ABR performs the Type-7 to 5 translation, but maintains the forward address field, essentially separating the relationship between the routing advertisement and the traffic flow. This principle can be illustrated as follows.

Before any modification in the OSPF domain, R5 performs a lookup on the Type-5 LSA for 9.9.9.9 that was translated from a Type-7 LSA. At this point R3 has an OSPF Router-ID of 150.1.3.3, and R6 has 150.1.6.6. The advertising router R5 sees is 150.1.6.6 (R6), since R6 won the translator election due to the higher RID. Note however, the forward address is set to 150.1.9.9 (SW3). This means that R5 next needs to figure out how to route towards 150.1.9.9.

```
Rack1R5#show ip ospf database external 9.9.9.9

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 224
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 9.9.9.9 (External Network Number )
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000001
  Checksum: 0x1F10
  Length: 36
  Network Mask: /32
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 150.1.9.9
        External Route Tag: 0
```

Since 150.1.9.9 does not belong to a device in the same area as R5, an inter-area lookup is performed on the Type-3 LSA. R5 finds that two ABRs are advertising the route to 150.1.9.9, 150.1.3.3 (R3) and 150.1.6.6 (R6), both with a metric of 3.

```
Rack1R5#show ip ospf database summary 150.1.9.9

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Summary Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 603
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.9.9 (summary Network Number)
  Advertising Router: 150.1.3.3
  LS Seq Number: 80000001
  Checksum: 0xBD35
  Length: 28
  Network Mask: /32
        TOS: 0  Metric: 3

  LS age: 4 (DoNotAge)
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.9.9 (summary Network Number)
  Advertising Router: 150.1.6.6
  LS Seq Number: 80000001
  Checksum: 0x9656
  Length: 28
  Network Mask: /32
        TOS: 0  Metric: 3
```

Copyright © 2010 Internetwork Expert

R5 now needs to find what the metric is to reach these ABRs. R5 checks its locally originated Type-1 Router LSA and finds that 150.1.1.1 (R1) and 150.1.3.3 (R3) are directly attached, both with a metric of 64.

```
Rack1R5#show ip ospf database router 150.1.5.5 self-originate

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Router Link States (Area 0)

  LS age: 1620
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.5.5
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000019
  Checksum: 0x7DAE
  Length: 108
  Area Border Router
  Number of Links: 7

<snip>
    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.1.1
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 64
<snip>

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.3.3
     (Link Data) Router Interface address: 155.1.0.5
      Number of TOS metrics: 0
       TOS 0 Metrics: 64
<snip>
```

Copyright © 2010 Internetwork Expert

R5 asks R1 who it is adjacent with, and finds 150.1.6.6 (R6) is connected over a Virtual-Link with a metric of 1.

```
Rack1R5#show ip ospf database router 150.1.1.1

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Router Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 773
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.1.1
  Advertising Router: 150.1.1.1
  LS Seq Number: 8000001B
  Checksum: 0xFA2E
  Length: 72
  Area Border Router
  Number of Links: 4

    Link connected to: a Virtual Link
     (Link ID) Neighboring Router ID: 150.1.6.6
     (Link Data) Router Interface address: 155.1.146.1
      Number of TOS metrics: 0
       TOS 0 Metrics: 1
```

This means that R5's intra-area cost to R3 is 64, and to R6 is 65. Since both R3 and R6 reported a cost of 3 to the forward address 150.1.9.9, the total forward metric through R6 is 65+3 = 68, but is only 64+3 = 67 through R3. Therefore although the route is originated into area 0 by R6, the Type-7 to 5 translator, the traffic does not actually flow through R6. Instead the path through R3 installed with the default redistribution metric of 20 for the E2 route, and a forward metric of 67 through R3.

```
Rack1R5#show ip route 9.9.9.9
Routing entry for 9.9.9.9/32
  Known via "ospf 1", distance 110, metric 20, type extern 2, forward metric 67
  Last update from 155.1.0.3 on Serial0/0, 00:17:27 ago
  Routing Descriptor Blocks:
  * 155.1.0.3, from 150.1.6.6, 00:17:27 ago, via Serial0/0
      Route metric is 20, traffic share count is 1

Rack1R5#traceroute 9.9.9.9

Type escape sequence to abort.
Tracing the route to 9.9.9.9

  1 155.1.0.3 28 msec 32 msec 32 msec
  2 155.1.37.7 32 msec 32 msec 32 msec
  3 155.1.79.9 28 msec *  28 msec
```

Copyright © 2010 Internetwork Expert

This illustrates why a Type-5 External route that was translated from a Type-7 NSSA External route does not use a Type-4 ASBR Summary LSA, because the forward address lookup replaces the need for the ASBR Summary lookup. Since the forward address is preserved only one router needs to do the translation, while the calculation of the final forwarding path stays independent

The Type-7 to 5 translator election can be modified by increasing R3's router-id to be higher than R6's.

```
Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#router ospf 1
Rack1R3(config-router)#router-id 150.1.30.30
Reload or use "clear ip ospf process" command, for this to take effect
Rack1R3(config-router)#end
Rack1R3#clear ip ospf 1 process
Reset OSPF process? [no]: yes
Rack1R3#

Rack1R5#show ip ospf database external 9.9.9.9

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 42
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 9.9.9.9 (External Network Number )
  Advertising Router: 150.1.30.30
  LS Seq Number: 80000001
  Checksum: 0xE519
  Length: 36
  Network Mask: /32
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 150.1.9.9
        External Route Tag: 0
```

R5 now sees the advertising router as 150.1.30.30 (R3), because this is the highest router-id of the ABRs connecting the NSSA to area 0.

Copyright © 2010 Internetwork Expert

Although the advertising router has changed, the forward address is still 150.1.9.9, which means the traffic flow has not changed.

```
Rack1R5#traceroute 9.9.9.9

Type escape sequence to abort.
Tracing the route to 9.9.9.9

  1 155.1.0.3 28 msec 32 msec 28 msec
  2 155.1.37.7 32 msec 32 msec 32 msec
  3 155.1.79.9 28 msec *  28 msec
```

Only once the forward metric via R3 is higher than the forward metric via R6 will the path selection change. This can be accomplished by changing the OSPF cost on R3's link to SW1.

```
Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#interface Fa0/0
Rack1R3(config-if)#ip ospf cost 1000
Rack1R3(config-if)#end
```

Now when R5 computes the forward metric through R3 it sees the same intra-area cost of 64 to R3, but R3 increased the advertised metric to 150.1.9.9 to 1002.

```
Rack1R5#show ip ospf database summary 150.1.9.9 adv-router 150.1.30.30

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Summary Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 70
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.9.9 (summary Network Number)
  Advertising Router: 150.1.30.30
  LS Seq Number: 80000002
  Checksum: 0x854B
  Length: 28
  Network Mask: /32
        TOS: 0  Metric: 1002
```

The final result is that the route is installed via R6 with a metric of 20 reported by the ASBR, plus the forward metric of 68 through R6.

```
Rack1R5#show ip route 9.9.9.9
Routing entry for 9.9.9.9/32
  Known via "ospf 1", distance 110, metric 20, type extern 2, forward
metric 68
  Last update from 155.1.0.1 on Serial0/0, 00:02:06 ago
  Routing Descriptor Blocks:
  * 155.1.0.1, from 150.1.30.30, 00:02:06 ago, via Serial0/0
      Route metric is 20, traffic share count is 1

Rack1R5#traceroute 9.9.9.9

Type escape sequence to abort.
Tracing the route to 9.9.9.9

  1 155.1.0.1 29 msec 28 msec 28 msec
  2 155.1.146.6 28 msec 32 msec 28 msec
  3 155.1.67.7 32 msec 32 msec 28 msec
  4 155.1.79.9 32 msec *  28 msec
```

If R6 loses connectivity to area 0 its route to the forward address is lost, and traffic is rerouted to R3.

```
Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface Fa0/0.146
Rack1R6(config-subif)#shutdown
Rack1R6(config-subif)#


Rack1R5#traceroute 9.9.9.9

Type escape sequence to abort.
Tracing the route to 9.9.9.9

  1 155.1.0.3 28 msec 32 msec 28 msec
  2 155.1.37.7 28 msec 28 msec 28 msec
  3 155.1.79.9 32 msec *  28 msec
```

Copyright © 2010 Internetwork Expert

6.31 OSPF NSSA Redistribution Filtering

- Create new Loopback interfaces on R5 and SW4 with IP addresses 5.5.5.5/32 and 10.10.10.10/32 respectively, and redistribute them into the OSPF domain.
- Configure area 3 as an NSSA so R5 blocks all LSA types 3, 4, and 5 and replaces them with a default route.
- Modify area 3 so that R5's redistributed Loopback is injected into area 0 as LSA Type-5, but is not injected into area 3 as LSA Type-7.

Configuration

```
R5:
interface Loopback5
 ip address 5.5.5.5 255.255.255.255
!
router ospf 1
 redistribute connected subnets
 area 3 nssa no-redistribution no-summary

SW2:
router ospf 1
 area 3 nssa

SW4:
interface Loopback10
 ip address 10.10.10.10 255.255.255.255
!
router ospf 1
 redistribute connected subnets
 area 3 nssa
```

Verification

Note

In certain NSSA designs the ABR can be an ASBR at the same time. When routes are redistributed directly on the ABR they are originated into area 0 as Type-5 External LSAs, and into the NSSA as Type-7 NSSA External LSAs. The origination as Type-7 into the NSSA may be unnecessary overhead if the ABR performing the redistribution is the only exit point out of the area. In this particular case R5 is both an ABR and ASBR, and is the only exit point for SW2 and SW4 to route packets into area 0.

Copyright © 2010 Internetwork Expert

By configuring the **area 3 nssa no-summary** option on R5, along with the **area 3 nssa** option on SW2 and SW4, the amount of routes contained in the area 3 database are minimized, while still allowing redistribution on SW4. Before R5 performs redistribution the database in area 3 looks as follows.

```
Rack1SW2#show ip ospf database

            OSPF Router with ID (150.1.8.8) (Process ID 1)

                Router Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum Link
count
150.1.5.5       150.1.5.5       142         0x80000035 0x004250 3
150.1.8.8       150.1.8.8       138         0x80000033 0x005FEB 4
150.1.10.10     150.1.10.10     638         0x80000030 0x00E128 3

                Net Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.58.8      150.1.8.8       138         0x80000030 0x003203
155.1.108.10    150.1.10.10     638         0x8000002E 0x00579D

                Summary Net Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
0.0.0.0         150.1.5.5       148         0x80000003 0x00A0F1

                Type-7 AS External Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum Tag
10.10.10.10     150.1.10.10     643         0x80000001 0x0043D3 0
```

SW2 knows about the three routers in the area via Type-1 Router LSAs, the two DRs in the area via Type-2 Network LSAs, an inter-area default route originated by R5 as a Type-3 Summary LSA, and SW4's redistributed Loopback via a Type-7 NSSA External LSA. This is essentially the minimal information needed in the database to perform intra-area SPF, use default routing to leave the area, and still allow redistribution. Next, R5 performs redistribution into OSPF.

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#router ospf 1
Rack1R5(config-router)#redistribute connected subnets
```

Copyright © 2010 Internetwork Expert

R5 originates the link 5.5.5.5/32 into area 3 as a Type-7 NSSA External LSA, as well as into area 0 as a Type-5 External LSA.

```
Rack1R5#show ip ospf database | begin Type-7
                Type-7 AS External Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum Tag
5.5.5.5         150.1.5.5       156         0x80000001 0x005098 0
10.10.10.10     150.1.10.10     1219        0x80000001 0x0043D3 0

                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
5.5.5.5         150.1.5.5       156         0x80000001 0x006C7E 0
9.9.9.9         150.1.30.30     1484        0x8000001A 0x00B332 0
10.10.10.10     150.1.5.5       701         0x80000001 0x001912 0
<snip>
```

The problem with this design is that unnecessary information is now in the database of area 3. Since the area 3 routers already had a default route via R5, having specific reachability information about the network 5.5.5.5/32 is redundant.

```
Rack1SW2#show ip route ospf
     155.1.0.0/24 is subnetted, 5 subnets
O       155.1.10.0 [110/2] via 155.1.108.10, 00:14:41, Port-channel1
O       155.1.5.0 [110/2] via 155.1.58.5, 00:14:41, Vlan58
     5.0.0.0/32 is subnetted, 1 subnets
O N2    5.5.5.5 [110/20] via 155.1.58.5, 00:05:32, Vlan58
     10.0.0.0/32 is subnetted, 1 subnets
O N2    10.10.10.10 [110/20] via 155.1.108.10, 00:14:41, Port-channel1
     150.1.0.0/16 is variably subnetted, 3 subnets, 2 masks
O       150.1.10.10/32 [110/2] via 155.1.108.10, 00:14:41, Port-channel1
O       150.1.5.5/32 [110/2] via 155.1.58.5, 00:14:41, Vlan58
O*IA 0.0.0.0/0 [110/2] via 155.1.58.5, 00:14:41, Vlan58
```

Therefore this design is a good candidate for Type-7 LSA suppression on the ABR itself. By adding the **no-redistribution** keyword onto the **area 3 nssa** statement of R5, Type-7 LSAs are not generated for locally redistributed routes. This does not, however, prevent other devices inside the NSSA from performing redistribution, such as SW4.

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#router ospf 1
Rack1R5(config-router)#area 3 nssa no-redistribution no-summary
```

Copyright © 2010 Internetwork Expert

Devices in area 3 no longer have a specific route to 5.5.5.5/32, but can reach it using the default route. Also a Type-7 NSSA External LSA still exists for 10.10.10.10/32.

```
Rack1SW2#show ip route ospf
     155.1.0.0/24 is subnetted, 5 subnets
O       155.1.10.0 [110/2] via 155.1.108.10, 00:21:20, Port-channel1
O       155.1.5.0 [110/2] via 155.1.58.5, 00:21:20, Vlan58
     10.0.0.0/32 is subnetted, 1 subnets
O N2    10.10.10.10 [110/20] via 155.1.108.10, 00:04:49, Port-channel1
     150.1.0.0/16 is variably subnetted, 3 subnets, 2 masks
O       150.1.10.10/32 [110/2] via 155.1.108.10, 00:21:20, Port-channel1
O       150.1.5.5/32 [110/2] via 155.1.58.5, 00:21:20, Vlan58
O*IA 0.0.0.0/0 [110/2] via 155.1.58.5, 00:04:54, Vlan58

Rack1SW2#ping 5.5.5.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 5.5.5.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/9 ms
```

Devices in area 0 and beyond have a specific route for 5.5.5.5/32 as a Type-5 External LSA.

```
Rack1R3#show ip route 5.5.5.5
Routing entry for 5.5.5.5/32
  Known via "ospf 1", distance 110, metric 20, type extern 2, forward
metric 781
  Last update from 155.1.0.5 on Serial1/0, 00:12:20 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.5.5, 00:12:20 ago, via Serial1/0
      Route metric is 20, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

6.32 OSPF LSA Type-3 Filtering

- Configure LSA Type-3 Filtering on R5 so that devices in area 0 do not have reachability information about the layer 3 EtherChannel link between SW2 and SW4 or SW4's Loopback 0 interface.
- Configure LSA Type-3 Filtering on R2 so that devices in area 51 do not have reachability information about the Loopback 0 interface of R5; this should not affect devices in area 5.

Configuration

```
R2:
ip prefix-list R5_LOOPBACK deny 150.1.5.5/32
ip prefix-list R5_LOOPBACK permit 0.0.0.0/0 le 32
!
router ospf 1
 area 51 filter-list prefix R5_LOOPBACK in

R5:
ip prefix-list AREA_3_ROUTES deny 150.1.10.10/32
ip prefix-list AREA_3_ROUTES deny 155.1.108.0/24
ip prefix-list AREA_3_ROUTES permit 0.0.0.0/0 le 32
!
router ospf 1
 area 3 filter-list prefix AREA_3_ROUTES out
```

Verification

Note

LSA Type-3 Filtering, like stub areas, is used to remove LSAs from the database as advertisements move between areas. Unlike stub areas however, Type-3 LSA Filtering can be used to permit or deny any arbitrary inter-area routes based on a prefix-list.

The filter-list syntax supports the arguments *in* and *out*, which are used to allow more control on ABRs that terminate multiple areas. In the case of R5, who terminates only area 0 and area 3, the syntax **`area 3 filter-list prefix AREA_3_ROUTES out`** has the same result as **`area 0 filter-list prefix AREA_3_ROUTES in`**. However on R2, who terminates areas 0, 5, and 51, the syntax **`area 51 filter-list prefix R5_LOOPBACK in`** applies only to prefixes leaving areas 0 and 5 going into area 51, while the syntax **`area 0 filter-list prefix R5_LOOPBACK out`** would apply to prefixes leaving area 0 going into area 5 and 51.

Copyright © 2010 Internetwork Expert

In other words if **area 0 filter-list prefix R5_LOOPBACK out** was applied, R5's loopback would not enter area 5 or 51 on R2, but with **area 51 filter-list prefix R5_LOOPBACK in** applied, R5's loopback enters area 5 but not area 51. This configuration can be verified by viewing the database and the routing table. Prior to filtering R2 originated 150.1.5.5 into area 51 as a Type-3 Summary LSA.

```
Rack1R2#show ip ospf database | begin Area 51

                Router Link States (Area 51)

Link ID         ADV Router      Age         Seq#       Checksum Link count
150.1.2.2       150.1.2.2       1062        0x8000002F 0x00C76C 1
192.10.1.254    192.10.1.254    707         0x8000015E 0x00CFCE 1

                Net Link States (Area 51)

Link ID         ADV Router      Age         Seq#       Checksum
192.10.1.254    192.10.1.254    707         0x8000002E 0x0039D7

                Summary Net Link States (Area 51)

Link ID         ADV Router      Age         Seq#       Checksum
150.1.1.1       150.1.2.2       310         0x8000002E 0x000A4F
150.1.2.2       150.1.2.2       1062        0x8000002E 0x00EFE7
150.1.3.3       150.1.2.2       1827        0x8000001B 0x000662
150.1.4.4       150.1.2.2       310         0x8000002E 0x00CA88
150.1.5.5       150.1.2.2       310         0x8000002E 0x00335E
150.1.7.7       150.1.2.2       1827        0x80000023 0x00D497
150.1.8.8       150.1.2.2       1323        0x80000002 0x005660
150.1.9.9       150.1.2.2       1827        0x80000023 0x00B4B2
<snip>
```

After the filter is applied this route is withdrawn from area 51, but not from area 5.

```
Rack1R2#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R2(config)#ip prefix-list R5_LOOPBACK deny 150.1.5.5/32
Rack1R2(config)#ip prefix-list R5_LOOPBACK permit 0.0.0.0/0 le 32
Rack1R2(config)#router ospf 1
Rack1R2(config-router)#area 51 filter-list prefix R5_LOOPBACK in
Rack1R2(config-router)#end
Rack1R2#
```

Copyright © 2010 Internetwork Expert

```
Rack1R2#show ip ospf database | begin Area 51
                Router Link States (Area 51)

Link ID         ADV Router      Age         Seq#       Checksum Link
count
150.1.2.2       150.1.2.2       1080        0x8000002F 0x00C76C 1
192.10.1.254    192.10.1.254    725         0x8000015E 0x00CFCE 1

                Net Link States (Area 51)

Link ID         ADV Router      Age         Seq#       Checksum
192.10.1.254    192.10.1.254    725         0x8000002E 0x0039D7

                Summary Net Link States (Area 51)

Link ID         ADV Router      Age         Seq#       Checksum
150.1.1.1       150.1.2.2       329         0x8000002E 0x000A4F
150.1.2.2       150.1.2.2       1080        0x8000002E 0x00EFE7
150.1.3.3       150.1.2.2       1845        0x8000001B 0x000662
150.1.4.4       150.1.2.2       329         0x8000002E 0x00CA88
150.1.7.7       150.1.2.2       1845        0x80000023 0x00D497
150.1.8.8       150.1.2.2       1341        0x80000002 0x005660
150.1.9.9       150.1.2.2       1845        0x80000023 0x00B4B2
<snip>

Rack1R2#show ip ospf database | begin Area 5
                Router Link States (Area 5)

Link ID         ADV Router      Age         Seq#       Checksum Link
count
150.1.2.2       150.1.2.2       376         0x80000031 0x008B73 2
150.1.30.30     150.1.30.30     427         0x8000001D 0x00E651 2

                Summary Net Link States (Area 5)

Link ID         ADV Router      Age         Seq#       Checksum
150.1.1.1       150.1.2.2       870         0x8000002E 0x000A4F
150.1.1.1       150.1.30.30     427         0x8000001C 0x00DB87
150.1.2.2       150.1.2.2       1622        0x8000002E 0x00EFE7
150.1.2.2       150.1.30.30     427         0x8000001C 0x00C69A
150.1.3.3       150.1.2.2       376         0x8000001C 0x000463
150.1.3.3       150.1.30.30     427         0x8000001C 0x00911E
150.1.4.4       150.1.2.2       870         0x8000002E 0x00CA88
150.1.4.4       150.1.30.30     427         0x8000001C 0x009CC0
150.1.5.5       150.1.2.2       870         0x8000002E 0x00335E
150.1.5.5       150.1.30.30     427         0x8000001C 0x000596
150.1.7.7       150.1.2.2       376         0x80000024 0x00D298
150.1.7.7       150.1.30.30     428         0x8000001E 0x006D4C
150.1.8.8       150.1.2.2       1884        0x80000002 0x005660
150.1.8.8       150.1.30.30     1698        0x80000002 0x0004AA
<snip>
```

Copyright © 2010 Internetwork Expert

R5 knows how to reach SW4's Loopback 0, along with the link between SW2 and SW4, but devices in area 0 do not.

```
Rack1R5#show ip route | include 150.1.10.10|155.1.108.0
O       155.1.108.0/24 [110/2] via 155.1.58.8, 00:33:37, FastEthernet0/0
O       150.1.10.10/32 [110/3] via 155.1.58.8, 00:33:37, FastEthernet0/0

Rack1R3#show ip route | include 150.1.10.10|155.1.108.0
<null>
```

Pitfall

Note that devices in area 0 have the Type-5 External LSA 10.10.10.10/32 installed in the database, but not in the routing table. This problem is related to a lookup failure on the forward address 150.1.10.10, which is explored in detail in the next section.

```
Rack1R3#show ip ospf database external 10.10.10.10

            OSPF Router with ID (150.1.30.30) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 1605
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 10.10.10.10 (External Network Number )
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000001
  Checksum: 0x1912
  Length: 36
  Network Mask: /32
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 150.1.10.10
        External Route Tag: 0

Rack1R3#show ip route 10.10.10.10
% Network not in table
```

Copyright © 2010 Internetwork Expert

6.33 OSPF Forwarding Address Suppression

- Modify R5's NSSA configuration so that devices outside of area 3 maintain connectivity to SW4's redistributed Loopback 10.10.10.10/32.

Configuration

```
R5:
router ospf 1
 area 3 nssa no-redistribution no-summary translate type7 suppress-fa
```

Verification

> **Note**
>
> Recall that with OSPF database lookups on external routes, the *Forward Address* field determines who the next recursive lookup should be performed towards. With typical Type-5 External LSAs, such as the RIP routes R4 redistributes into OSPF, the forward address is normally set to 0.0.0.0. This means that the next lookup should be performed towards the *Advertising Router*.
>
> For example, when R4 redistributes the route 30.0.0.0/16 from RIP into OSPF, R1 performs a lookup on the Type-5 External LSA as follows.

```
Rack1R1#show ip ospf database external 30.0.0.0

            OSPF Router with ID (150.1.1.1) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 1595
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 30.0.0.0 (External Network Number )
  Advertising Router: 150.1.4.4
  LS Seq Number: 8000002E
  Checksum: 0x7E37
  Length: 36
  Network Mask: /16
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 0.0.0.0
        External Route Tag: 0
```

R1 sees the forward address field set to 0.0.0.0, which means a lookup should be performed on the advertising router 150.1.4.4.

Copyright © 2010 Internetwork Expert

R1 sees that on VLAN 146 it is adjacent with the DR 155.1.146.1, with a cost of 1.

```
Rack1R1#show ip ospf database router 150.1.1.1 self-originate | begin Area 1

               Router Link States (Area 1)

  LS age: 725
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.1.1
  Advertising Router: 150.1.1.1
  LS Seq Number: 80000036
  Checksum: 0x1F46
  Length: 36
  Area Border Router
  Virtual Link Endpoint
  Number of Links: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.146.1
     (Link Data) Router Interface address: 155.1.146.1
      Number of TOS metrics: 0
       TOS 0 Metrics: 1
```

R1 asks the DR who it is adjacent with, and finds that 150.1.4.4 is on the local segment.

```
Rack1R1#show ip ospf database network 155.1.146.1

            OSPF Router with ID (150.1.1.1) (Process ID 1)

                Net Link States (Area 1)

  Routing Bit Set on this LSA
  LS age: 749
  Options: (No TOS-capability, DC)
  LS Type: Network Links
  Link State ID: 155.1.146.1 (address of Designated Router)
  Advertising Router: 150.1.1.1
  LS Seq Number: 80000036
  Checksum: 0x73E6
  Length: 36
  Network Mask: /24
        Attached Router: 150.1.1.1
        Attached Router: 150.1.4.4
        Attached Router: 150.1.6.6
```

Copyright © 2010 Internetwork Expert

This means that when R1 installs 30.0.0.0/16 in the routing table the metric will be 20, from R4's default metric during redistribution, and a forward metric of 1 to reach R4.

```
Rack1R1#show ip route 30.0.0.1
Routing entry for 30.0.0.0/16
  Known via "ospf 1", distance 110, metric 20, type extern 2, forward metric 1
  Last update from 155.1.146.4 on FastEthernet0/0, 00:14:03 ago
  Routing Descriptor Blocks:
  * 155.1.146.4, from 150.1.4.4, 00:14:03 ago, via FastEthernet0/0
      Route metric is 20, traffic share count is 1
```

Now let's compare this normal Type-5 External LSA lookup to a Type-5 External LSA that was translated from a Type-7 NSSA External LSA. In this design SW4 redistributes the route 10.10.10.10/32 into area 3 as a Type-7 NSSA External LSA, and R5 translates it into a Type-5 External LSA as it moves to area 0.

R3 performs a lookup on the external route, and sees the advertising router set to 150.1.5.5 (R5), and the forward address set to 150.1.10.10.

```
Rack1R3#show ip ospf database external 10.10.10.10

            OSPF Router with ID (150.1.30.30) (Process ID 1)

                Type-5 AS External Link States

  LS age: 463
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 10.10.10.10 (External Network Number )
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000003
  Checksum: 0x1514
  Length: 36
  Network Mask: /32
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 150.1.10.10
        External Route Tag: 0
```

Since the forward address is non-zero, the next recursive lookup is performed towards 150.1.10.10, instead of the advertising router 150.1.5.5.

```
Rack1R3#show ip route 150.1.10.10
% Subnet not in table
```

Copyright © 2010 Internetwork Expert

The problem with this design however, is that the prefix 150.1.10.10 was filtered out from the previous Type-3 LSA Filter. The result is that recursion towards the external route fails, and it cannot be installed in the routing table.

```
Rack1R3#show ip route 10.10.10.10
% Subnet not in table
```

To resolve this problem a very specific feature can be configured on R5, which is known as OSPF Forwarding Address Suppression in Translated Type-5 LSAs. This feature, configured by adding the **translate type7 suppress-fa** argument onto the **area 3 nssa** statement, instructs the ABR to not preserve the value in the forward address field as a Type-7 NSSA External LSA is translated into a Type-5 External LSA.

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#router ospf 1
Rack1R5(config-router)#area 3 nssa no-redistribution no-summary
translate type7 suppress-fa
Rack1R5(config-router)#end
Rack1R5#

Rack1R3#show ip ospf database external 10.10.10.10

            OSPF Router with ID (150.1.30.30) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 36
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 10.10.10.10 (External Network Number )
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000004
  Checksum: 0x7F54
  Length: 36
  Network Mask: /32
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 0.0.0.0
        External Route Tag: 0
```

The result of this configuration is that R3 now sees the forward address for the Type-5 External LSA 10.10.10.10/32 set to 0.0.0.0, which means a lookup must now be performed on the advertising router 150.1.5.5.

Copyright © 2010 Internetwork Expert

In this particular case R3 finds that 150.1.5.5 is directly connected with a metric of 781 via a point-to-point adjacency.

```
Rack1R3#show ip ospf database router 150.1.30.30 self-originate

            OSPF Router with ID (150.1.30.30) (Process ID 1)

                Router Link States (Area 0)

  LS age: 1098
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.30.30
  Advertising Router: 150.1.30.30
  LS Seq Number: 8000001D
  Checksum: 0x5668
  Length: 60
  Area Border Router
  AS Boundary Router
  Number of Links: 3

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 150.1.5.5
     (Link Data) Router Interface address: 155.1.0.3
      Number of TOS metrics: 0
       TOS 0 Metrics: 781
```

The final result is that the external route is installed with a metric of 20, which is derived from SW4's default redistribution metric, plus a forward metric of 781 to reach R5.

```
Rack1R3#show ip route 10.10.10.10
Routing entry for 10.10.10.10/32
  Known via "ospf 1", distance 110, metric 20, type extern 2, forward
metric 781
  Last update from 155.1.0.5 on Serial1/0, 00:00:57 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 150.1.5.5, 00:00:57 ago, via Serial1/0
      Route metric is 20, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

Although this feature fixes the problem introduced by the Type-3 LSA Filter, suboptimal routing may be introduced when there are multiple exit points out of the NSSA. As previously discussed, normally the Type-7 to 5 translator election and the forward address calculation are kept separate, which means the control plane advertisement of the route does not need to follow the traffic forwarding plane, but with forwarding address suppression enabled the traffic will always flow through the Type-7 to 5 translator.

This can be seen with the previously configured case regarding SW3's Loopback interface 9.9.9.9/32 that was redistributed into area 2. R5 sees the external LSA 9.9.9.9/32 with an advertising router of 150.1.30.30 – R3, the Type-7 to 5 translator – and a forward address of 150.1.9.9.

```
Rack1R5#show ip ospf database external 9.9.9.9

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 436
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 9.9.9.9 (External Network Number )
  Advertising Router: 150.1.30.30
  LS Seq Number: 8000001D
  Checksum: 0xAD35
  Length: 36
  Network Mask: /32
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 150.1.9.9
        External Route Tag: 0
```

The lookup towards the forward address 150.1.9.9 results in using R6 as an exit point. This means that although R3 is advertising the route into area 0, the traffic flows through R6.

```
Rack1R5#show ip route 150.1.9.9
Routing entry for 150.1.9.9/32
  Known via "ospf 1", distance 110, metric 68, type inter area
  Last update from 155.1.0.1 on Serial0/0/0, 00:23:35 ago
  Routing Descriptor Blocks:
  * 155.1.0.1, from 150.1.6.6, 00:23:35 ago, via Serial0/0/0
      Route metric is 68, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#traceroute 9.9.9.9

Type escape sequence to abort.
Tracing the route to 9.9.9.9

  1 155.1.0.1 28 msec 28 msec 28 msec
  2 155.1.146.6 28 msec 32 msec 28 msec
  3 155.1.67.7 32 msec 32 msec 28 msec
  4 155.1.79.9 32 msec *  28 msec
```

Now let's see what happens when R3, the Type-7 to 5 translator, suppresses the forward address field.

```
Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#router ospf 1
Rack1R3(config-router)#area 2 nssa no-summary translate type7 suppress-fa

Rack1R5#show ip ospf database external 9.9.9.9

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 10
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 9.9.9.9 (External Network Number )
  Advertising Router: 150.1.30.30
  LS Seq Number: 8000001E
  Checksum: 0x3358
  Length: 36
  Network Mask: /32
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 20
        Forward Address: 0.0.0.0
        External Route Tag: 0
```

R5 now sees the forward address as 0.0.0.0, which means the recursive lookup should be performed on the advertising router 150.1.30.30. The result is that the traffic flows through R3, the Type-7 to 5 translator, which technically is sub-optimal in this design.

```
Rack1R5#traceroute 9.9.9.9

Type escape sequence to abort.
Tracing the route to 9.9.9.9

  1 155.1.0.3 32 msec 28 msec 28 msec
  2 155.1.37.7 32 msec 28 msec 32 msec
  3 155.1.79.9 32 msec *  28 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Note

Revert all devices to the *Full OSPF Routing* initial configurations before continuing.

6.34 OSPF Default Routing

- Configure R6 with a static default route pointing to BB1.
- Configure R6 to originate an external type-2 default route with a metric of 60 into OSPF as long as the static default route is installed in the routing table.
- Do not use a route-map to accomplish this.
- Configure R4 to originate an external type-1 default route with a metric of 40 regardless of whether it has a default route already installed in the routing table.

Configuration

```
R4:
router ospf 1
 default-information originate always metric 40 metric-type 1

R6:
ip route 0.0.0.0 0.0.0.0 54.1.1.254
!
router ospf 1
 default-information originate metric 60
```

Verification

Note

Default routing for non-stub areas in OSPF is accomplished through the origination of Type-5 External LSAs via the **`default-information originate`** command. Without any additional arguments, the OSPF process first checks to see if a default route is installed in the routing table. If a default route is already installed, such as via a static route or learned via BGP, the OSPF default route is originated. If the default route is not found, no origination occurs. This behavior is typically desirable in designs with multiple exit points out of the OSPF domain to upstream networks.

Copyright © 2010 Internetwork Expert

For example, imagine an OSPF network with exit points A and B out to the Internet. Both router A and B are running BGP with upstream peers, and learning a default route via BGP. As long as both devices maintain their upstream peerings, a default route can be advertised into OSPF. However, if A's link to the upstream neighbor is lost, and hence its default route via BGP is lost, its OSPF default route advertisement is withdrawn. The result of this design is that an individual exit point will only collect default traffic if they themselves have a default exit point to upstream networks.

This behavior can be modified by adding the **always** argument to the **default-information originate** statement, which essentially skips the checking for a default route already being installed in the table.

The below view of the OSPF database on R1 indicates that both R4 and R6 are originating a default route. Without additional arguments on the command the default route would have been advertised as a Type-2 External route with a metric of 20. The same route lookup logic is applied to these default routes as normal Type-5 External LSAs, where E1 is preferred over E2, and if multiple E2 routes exist with the same metric, the forward metrics are compared.

```
Rack1R1#show ip ospf database external 0.0.0.0

            OSPF Router with ID (150.1.1.1) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 297
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 0.0.0.0 (External Network Number )
  Advertising Router: 150.1.4.4
  LS Seq Number: 80000001
  Checksum: 0xB7B4
  Length: 36
  Network Mask: /0
        Metric Type: 1 (Comparable directly to link state metric)
        TOS: 0
        Metric: 40
        Forward Address: 0.0.0.0
        External Route Tag: 1
```

```
LS age: 136
Options: (No TOS-capability, DC)
LS Type: AS External Link
Link State ID: 0.0.0.0 (External Network Number )
Advertising Router: 150.1.6.6
LS Seq Number: 80000002
Checksum: 0xE7EA
Length: 36
Network Mask: /0
      Metric Type: 2 (Larger than any link state path)
      TOS: 0
      Metric: 60
      Forward Address: 0.0.0.0
      External Route Tag: 1
```

The conditional checking for the already installed default route on R6 can be verified as seen below. As long as R6 has the static default route installed in the routing table, a default route is originated into OSPF. Once the route is removed, only R4 continues to originate default information.

```
Rack1R6#show ip route static
S*   0.0.0.0/0 [1/0] via 54.1.1.254
Rack1R6#show ip ospf database | include _0.0.0.0
0.0.0.0         150.1.4.4       608         0x80000001 0x00B7B4 1
0.0.0.0         150.1.6.6       11          0x80000001 0x00E9E9 1

Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#no ip route 0.0.0.0 0.0.0.0 54.1.1.254
Rack1R6(config)#do show ip ospf database | include _0.0.0.0
0.0.0.0         150.1.4.4       621         0x80000001 0x00B7B4 1
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

6.35 OSPF Conditional Default Routing

- Remove the previous default route advertisement on R6.
- Configure R6 to originate a default route into OSPF, but only if the local Serial interface connecting to BB1 is up.

Configuration

```
R6:
router ospf 1
 default-information originate always route-map TRACK_LINK_TO_BB1
!
ip prefix-list LINK_TO_BB1 seq 5 permit 54.1.1.0/24
!
route-map TRACK_LINK_TO_BB1 permit 10
 match ip address prefix-list LINK_TO_BB1
```

Verification

> **Note**
>
> Conditional default-information origination in OSPF uses a route-map to check for the existence of a specific prefix in the IP routing table before the default route is originated. Recall from the previous section that the default condition is to check for a default route already installed in the IP routing table. In this example the check for the existing default route is circumvented with the **`always`** keyword added to the **`default-information originate`** statement.
>
> In this design R6 is configured to check for the prefix 54.1.1.0/24 in the routing table. If the route exists, the default route is originated. As seen below once this prefix is no longer in the routing table, the default advertisement is withdrawn.

```
Rack1R6#show ip ospf database | include _0.0.0.0_
0.0.0.0         150.1.4.4       856         0x80000001 0x00B7B4 1
0.0.0.0         150.1.6.6       47          0x80000001 0x009975 1

Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface Serial0/0/0
Rack1R6(config-if)#frame-relay lmi-type q933a
Rack1R6(config-if)#end
Rack1R6#
%SYS-5-CONFIG_I: Configured from console by console
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 10: Neighbor 54.1.1.254 (Serial0/0/0) is
down: holding time expired
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed
state to down

Rack1R6#show ip ospf database | include _0.0.0.0
0.0.0.0         150.1.4.4       916         0x80000001 0x00B7B4 1
```

Copyright © 2010 Internetwork Expert

6.36 OSPF Reliable Conditional Default Routing

- Remove the previous default route advertisement on R4.
- Configure an IP SLA instance on R4 to check ICMP reachability to BB3 every five seconds.
- R4 should advertise a default route into OSPF, but only if the SLA monitoring reports its status as OK.

Configuration

```
R4:
ip sla 1
 icmp-echo 204.12.30.254
 timeout 2000
 frequency 5
ip sla schedule 1 life forever start-time now
!
track 1 rtr 1
!
ip route 169.254.0.1 255.255.255.255 Null0 track 1
!
ip prefix-list PLACEHOLDER seq 5 permit 169.254.0.1/32
!
route-map TRACK_PLACEHOLDER permit 10
 match ip address prefix-list PLACEHOLDER
!
router ospf 1
 default-information originate always route-map TRACK_PLACEHOLDER
```

Verification

Note

In the previous example R6 was configured to track the status of its Frame Relay interface connecting to BB1. When the link was down, and its IP prefix withdrawn from the routing table, the default route origination was withdrawn based on the conditional checking. For this redundancy design to work properly, we would have to assume that the end-to-end reachability status on the Frame Relay circuit to BB1 can be accurately determined by the Layer 2 interface status. However, as we have seen in previous designs before, tracking an interface status directly is *not* a good indication of end-to-end reachability, as the interface could be UP/UP locally, but the circuit itself could be down.

Copyright © 2010 Internetwork Expert

Another example that is common in today's networks is with Metro Ethernet. Since the router's local Ethernet interface only tracks link status to its attached switch, end-to-end reachability cannot be inferred by checking this link status. In the output below we see a case where R4 wants to originate a default route only when the link to BB3 is viable. As the prefix 204.12.1.0/24 is currently in the routing table, the default route is originated.

```
R4:
router ospf 1
 default-information originate always route-map TRACK_LINK_TO_BB3
!
ip prefix-list LINK_TO_BB3 seq 5 permit 204.12.1.0/24
!
route-map TRACK_LINK_TO_BB3 permit 10
 match ip address prefix-list LINK_TO_BB3

Rack1R4#show ip ospf database | include _0.0.0.0
0.0.0.0         150.1.4.4       72          0x80000001 0x00B35F 1
0.0.0.0         150.1.6.6       392         0x80000001 0x009975 1
```

Now we simulate a case where a failure occurs in the network, but not on the segment attached directly to R4. In this example the switchport between SW3 and BB3 is shutdown. This effectively breaks the end-to-end connectivity between R4 and BB3, but leaves the local link status of R4 as UP/UP, which meets the default route condition.

```
Rack1SW3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW3(config)#interface Fa0/24 Rack1SW3(config-
if)#shutdown
Rack1SW3(config-if)#

Rack1R4#ping 204.12.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 204.12.1.254, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)

Rack1R4#show ip ospf database | include  0.0.0.0
0.0.0.0         150.1.4.4       96          0x80000001 0x00B35F 1
0.0.0.0         150.1.6.6       416         0x80000001 0x009975 1
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ip route 204.12.1.0
Routing entry for 204.12.1.0/24
  Known via "connected", distance 0, metric 0 (connected, via
interface)
  Redistributing via ospf 1, rip
  Advertised by ospf 1
                rip
  Routing Descriptor Blocks:
  * directly connected, via FastEthernet0/0
      Route metric is 0, traffic share count is 1
```

As we can see, even though R4's uplink is no longer viable to end-to-end connectivity, the default route is still advertised since 204.12.1.0/24 is still in the routing table. To add more intelligence to this design, an IP SLA tracking instance can be used for more reliable information than the layer 2 interface's status.

```
Rack1R4(config)#ip sla 1
Rack1R4(config-ip-sla)#icmp-echo 204.12.30.254
Rack1R4(config-sla-echo)#timeout 2000
Rack1R4(config-sla-echo)#frequency 5
Rack1R4(config-sla-echo)#exit
Rack1R4(config)#ip sla schedule 1 start now life forever
Rack1R4(config)#track 1 rtr 1
Rack1R4(config-track)#exit
Rack1R4(config)#ip route 169.254.0.1 255.255.255.255 null0 track 1
Rack1R4(config)#ip prefix-list PLACEHOLDER permit 169.254.0.1/32
Rack1R4(config)#route-map TRACK_PLACEHOLDER
Rack1R4(config-route-map)#match ip address prefix-list PLACEHOLDER
Rack1R4(config-route-map)#end
Rack1R4#
```

The IP SLA instance is instructed to ping 204.12.1.254 every five seconds, and wait two seconds for an echo-reply. The SLA instance is called from a tracked object, which in turn is called from an arbitrary placeholder static route. As seen below, with the tracked object reporting the status code OK, the default route is originated.

```
Rack1R4#show track
Track 1
  Response Time Reporter 1 state
  State is Up
    3 changes, last change 00:00:04
  Latest operation return code: OK
  Latest RTT (millisecs) 3
  Tracked by:
    STATIC-IP-ROUTING 0
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ip ospf database | include  0.0.0.0
0.0.0.0         150.1.4.4       4           0x80000001 0x00B35F 1
0.0.0.0         150.1.6.6       806         0x80000001 0x009975 1
```

Now the same indirect failure scenario is simulated as previously seen. Specifically SW3's link to BB3 is disabled, while R4's local link stays up.

```
Rack1SW3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW3(config)#interface Fa0/24
Rack1SW3(config-if)#shutdown
Rack1SW3(config-if)#
```

R4 detects the failure of the SLA instance, and the tracked object transitions to down.

```
Rack1R4#
%TRACKING-5-STATE: 1 rtr 1 state Up->Down
Rack1R4#show track
Track 1
  Response Time Reporter 1 state
  State is Down
    4 changes, last change 00:00:04
  Latest operation return code: Timeout
  Tracked by:
    STATIC-IP-ROUTING 0
```

The failure of the tracked object causes the static route to be withdrawn. Since the route-map condition for the default-information originate statement is looking for this prefix to be installed in the routing table, the default route is withdrawn when 169.254.0.1 is withdrawn.

```
Rack1R4#show ip route 169.254.0.1
% Network not in table

Rack1R4#show ip ospf database | include _0.0.0.0
0.0.0.0         150.1.6.6       785         0x80000001 0x009975 1
```

Copyright © 2010 Internetwork Expert

6.37 OSPF Filtering with Distribute-Lists

- Configure distribute-list filtering on R5, SW2, and SW4 so that these devices do not install routes to the Loopback 0 networks of R1 and R2.

Configuration

```
R5:
router ospf 1
 distribute-list 1 in
!
access-list 1 deny   150.1.1.1
access-list 1 deny   150.1.2.2
access-list 1 permit any

SW2:
router ospf 1
 distribute-list 1 in
!
access-list 1 deny   150.1.1.1
access-list 1 deny   150.1.2.2
access-list 1 permit any

SW4:
router ospf 1
 distribute-list 1 in
!
access-list 1 deny   150.1.1.1
access-list 1 deny   150.1.2.2
access-list 1 permit any
```

Verification

Note

Recall that to properly compute SPF, all routers within an OSPF area must agree on their view of the database. This implies that OSPF filtering in the database can be accomplished between areas, but not within an area. Inter-area filtering has been previously demonstrated with stub areas, and the Type-3 LSA Filter. Intra-area filtering *can* be accomplished in OSPF with an inbound distribute-list; however this filtering only affects the local routing table, not the OSPF database.

Copyright © 2010 Internetwork Expert

Once the distribute-list has been applied on R5, the routes 150.1.2.2/32 and 150.1.1.1/32 no longer appear in the routing table.

```
Rack1R5#show ip route | include 150.
     150.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
C       150.1.5.0/24 is directly connected, Loopback0
O       150.1.10.10/32 [110/3] via 155.1.58.8, 00:01:42,
FastEthernet0/0
O IA    150.1.9.9/32 [110/67] via 155.1.0.3, 00:01:42, Serial0/0/0
O       150.1.8.8/32 [110/2] via 155.1.58.8, 00:01:42, FastEthernet0/0
O IA    150.1.7.7/32 [110/66] via 155.1.0.3, 00:01:42, Serial0/0/0
O IA    150.1.6.6/32 [110/66] via 155.1.45.4, 00:01:42, Serial0/1/0
O       150.1.4.4/32 [110/65] via 155.1.45.4, 00:01:42, Serial0/1/0
O       150.1.3.3/32 [110/65] via 155.1.0.3, 00:01:42, Serial0/0/0
```

Viewing the OSPF database on SW4, however, indicates that the Type-3 (Inter-Area) LSA's 150.1.1.1 and 150.1.2.2 still exist.

```
Rack1SW4#show ip ospf database summary 150.1.1.1

            OSPF Router with ID (150.1.10.10) (Process ID 1)

                Summary Net Link States (Area 3)

  Routing Bit Set on this LSA
  LS age: 181
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.1.1 (summary Network Number)
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000003
  Checksum: 0xB608
  Length: 28
  Network Mask: /32
        TOS: 0  Metric: 65
```

Pitfall

This type of design can result in traffic black holes unless implemented carefully. If R5 was configured with the distribute-list, but SW2 and SW4 were not, traffic from SW4 towards 150.1.1.1 would be sent to SW2, from SW2 to R5, and then black holed on R5. When implementing inbound distribute-list filtering ensure that all routers still agree on the forwarding paths in the network.

Copyright © 2010 Internetwork Expert

6.38 OSPF Summarization and Discard Routes

- Configure R5 to advertise the summary 150.X.0.0/22 into area 3.
- Ensure that R5, SW2, and SW4 can still reach the Loopback 0 networks of R1 and R2, even with the previously configured distribute-list filter applied.

Configuration

```
R5:
router ospf 1
 no discard-route internal
 area 0 range 150.1.0.0 255.255.252.0
```

Verification

> **Note**
>
> When summarization is configured in OSPF, similar to EIGRP and BGP, a matching route to Null0 for the summary is installed locally in the routing table. This "discard" route is used to prevent the forwarding of traffic towards a shorter match, such as a default route, if no specific route towards the actual destination exists in the network.
>
> The automatic origination of the discard route can be disabled with the no discard-route [internal | external], where internal refers to inter-area summarization performed with the area range command, while external refers to redistributed summarization performed with the summary-address command. The operation of the discard route can be illustrated as follows.
>
> Per the previous task, R5, SW2, and SW4 have the prefixes 150.1.1.1/32 and 150.1.2.2/32 filtered out of the routing table with a distribute-list. Additionally, R5 is originating the summary 150.1.0.0/22 into area 3, which encompasses addresses 150.1.0.0 through 150.1.3.255.

```
Rack1SW2#show ip route | include 150.
     150.1.0.0/16 is variably subnetted, 8 subnets, 3 masks
O       150.1.10.10/32 [110/2] via 155.1.108.10, 00:06:54, Port-
channel1
O IA    150.1.9.9/32 [110/68] via 155.1.58.5, 00:06:54, Vlan58
O IA    150.1.7.7/32 [110/67] via 155.1.58.5, 00:06:54, Vlan58
O IA    150.1.6.6/32 [110/67] via 155.1.58.5, 00:06:54, Vlan58
O       150.1.5.5/32 [110/2] via 155.1.58.5, 00:06:54, Vlan58
O IA    150.1.4.4/32 [110/66] via 155.1.58.5, 00:06:54, Vlan58
O IA    150.1.0.0/22 [110/66] via 155.1.58.5, 00:00:41, Vlan58
C       150.1.8.0/24 is directly connected, Loopback0
```

Copyright © 2010 Internetwork Expert

Reachability to the network 150.1.3.3 is obtained from SW2, but reachability to 150.1.1.1 is not.

```
Rack1SW2#ping 150.1.3.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.3.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 58/60/67 ms

Rack1SW2#ping 150.1.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.1.1, timeout is 2 seconds:
U.U.U
Success rate is 0 percent (0/5)
```

This is due to the fact that R5's longest match to 150.1.1.1 is the discard route via Null0. Although R5 does have a valid default path to 150.1.1.1 via R4, this cannot be used because /22 is a longer match than /0.

```
Rack1R5#show ip route 150.1.1.1
Routing entry for 150.1.0.0/22
  Known via "ospf 1", distance 110, metric 0, type intra area
  Routing Descriptor Blocks:
  * directly connected, via Null0
      Route metric is 0, traffic share count is 1

Rack1R5#show ip route 0.0.0.0
Routing entry for 0.0.0.0/0, supernet
  Known via "ospf 1", distance 110, metric 1, candidate default path
  Tag 1, type extern 2, forward metric 64
  Last update from 155.1.0.4 on Serial0/0, 00:14:54 ago
  Routing Descriptor Blocks:
  * 155.1.45.4, from 150.1.4.4, 00:14:54 ago, via Serial0/1
      Route metric is 1, traffic share count is 1
      Route tag 1
    155.1.0.4, from 150.1.4.4, 00:14:54 ago, via Serial0/0
      Route metric is 1, traffic share count is 1
      Route tag 1
```

Copyright © 2010 Internetwork Expert

With the discard route removed on R5, the longest match to 150.1.1.1 is now 0.0.0.0/0.

```
Rack1R5(config)#router ospf 1
Rack1R5(config-router)#area 0 range 150.1.0.0 255.255.252.0
Rack1R5(config-router)#end
Rack1R5#
%SYS-5-CONFIG_I: Configured from console by console

Rack1R5#show ip route 150.1.1.1
% Subnet not in table

Rack1R5#show ip route 0.0.0.0
Routing entry for 0.0.0.0/0, supernet
  Known via "ospf 1", distance 110, metric 1, candidate default path
  Tag 1, type extern 2, forward metric 64
  Last update from 155.1.0.4 on Serial0/0, 00:00:09 ago
  Routing Descriptor Blocks:
  * 155.1.45.4, from 150.1.4.4, 00:00:09 ago, via Serial0/1
      Route metric is 1, traffic share count is 1
      Route tag 1
    155.1.0.4, from 150.1.4.4, 00:00:09 ago, via Serial0/0
      Route metric is 1, traffic share count is 1
      Route tag 1
```

The final result is that SW2 uses the 150.1.0.0/22 prefix to route traffic for 150.1.1.1 towards R5, while R5 uses the 0.0.0.0/0 prefix to route the traffic towards R4.

```
Rack1SW2#ping 150.1.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 58/60/67 ms

Rack1SW2#traceroute 150.1.1.1

Type escape sequence to abort.
Tracing the route to 150.1.1.1

  1 155.1.58.5 0 msec 9 msec 0 msec
  2 155.1.0.4 33 msec 34 msec 25 msec
  3 155.1.146.1 34 msec *  25 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

6.39 OSPF Filtering with Administrative Distance

- Configure administrative distance filtering on R5 so that traffic going towards the VLAN 67 network is sent towards R3.

Configuration

```
R5:
router ospf 1
 distance 255 150.1.6.6 0.0.0.0 67
!
access-list 67 permit 155.1.67.0
```

Verification

> **Note**
>
> Like the other routing protocols, administrative distance can be changed on a per-prefix and per-neighbor basis in OSPF. One key difference however, is the address field in the **`distance`** command refers to the originator of the prefix into the area, not necessarily the neighbor you are learning the route from.
>
> In the below output we can see that R5 has two routes to the prefix 155.1.67.0/24. Both are reachable out Serial0/0/0, one via the next hop 155.1.0.1 (R1), and one via the next-hop 155.1.0.1 (R3). Note however, the difference between the next-hop values and the "from" field.
>
> The prefix reachable via 155.1.0.3 is from 150.1.3.3, which is the router-id of R3. The prefix reachable via 155.1.0.1 is from 150.1.6.6, which is the router-id of R6. This means that R3 is using itself as the ABR exit point to the inter-area destination, while R1 is using R6 as the ABR. Therefore in order to change the distance of one of these prefixes over the other, the from 150.1.3.3 or 150.1.6.6 fields should be matched in the distance command.

```
Rack1R5#show ip route 155.1.67.0
Routing entry for 155.1.67.0/24
  Known via "ospf 1", distance 110, metric 66, type inter area
  Last update from 155.1.0.1 on Serial0/0, 00:06:18 ago
  Routing Descriptor Blocks:
  * 155.1.0.3, from 150.1.3.3, 00:06:18 ago, via Serial0/0
      Route metric is 66, traffic share count is 1
    155.1.0.1, from 150.1.6.6, 00:06:18 ago, via Serial0/0
      Route metric is 66, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#access-list 67 permit 155.1.67.0
Rack1R5(config)#router ospf 1
Rack1R5(config-router)#distance 255 150.1.6.6 0.0.0.0 67
Rack1R5(config-router)#end

Rack1R5#show ip route 155.1.67.0
%SYS-5-CONFIG_I: Configured from console by console
Rack1R5#show ip route 155.1.67.0
Routing entry for 155.1.67.0/24
  Known via "ospf 1", distance 110, metric 66, type inter area
  Last update from 155.1.0.3 on Serial0/0, 00:00:03 ago
  Routing Descriptor Blocks:
  * 155.1.0.3, from 150.1.3.3, 00:00:03 ago, via Serial0/0
      Route metric is 66, traffic share count is 1
```

Once the distance has be modified the only route left is the one via R3, hence the modified traffic flow as seen below.

```
Rack1R5#traceroute 155.1.67.6

Type escape sequence to abort.
Tracing the route to 155.1.67.6

  1 155.1.0.3 32 msec 28 msec 28 msec
  2 155.1.37.7 33 msec 32 msec 32 msec
  3 155.1.67.6 28 msec *  28 msec
```

Copyright © 2010 Internetwork Expert

6.40 OSPF Filtering with Route-Maps

- Configure route-map filtering on R6 so that traffic going towards the Loopback 0 network of R3 is sent towards R1.

Configuration

```
R6:
router ospf 1
 distribute-list route-map DENY_R3_LOOPBACK_FROM_R4 in
!
access-list 3 permit 150.1.3.3
access-list 4 permit 155.1.146.4
!
route-map DENY_R3_LOOPBACK_FROM_R4 deny 10
 match ip address 3
 match ip next-hop 4
!
route-map DENY_R3_LOOPBACK_FROM_R4 permit 20
```

Verification

Note

Referencing a route-map with a distribute-list in OSPF extends the filtering capability with additional match criteria. Specifically, the matching of interface (outgoing interface in the routing table), ip address, ip next-hop, ip route-source (router-id of the originating router), metric, route-type (intra-area, inter-area, etc.), and tagging are supported inside the route-map.

Note the limitation of this feature is that only inbound filtering is supported, and the filter still is only local to the router's routing table, i.e. the filter does not affect the OSPF database advertisements.

In the below output we can see that R6 has two routes to the prefix 150.1.3.3/32. The first route is via 155.1.146.4 via ABR 150.1.3.3, and the second via 155.1.146.1 via the ABR 150.1.3.3. Both are reachable out the same interface.

```
Rack1R6#show ip route 150.1.3.3
Routing entry for 150.1.3.3/32
  Known via "ospf 1", distance 110, metric 130, type intra area
  Redistributing via eigrp 10
  Advertised by eigrp 10 metric 100000 1000 255 1 1500
  Last update from 155.1.146.1 on FastEthernet0/0.146, 00:00:02 ago
  Routing Descriptor Blocks:
    155.1.146.4, from 150.1.3.3, 00:00:02 ago, via FastEthernet0/0.146
      Route metric is 130, traffic share count is 1
  * 155.1.146.1, from 150.1.3.3, 00:00:02 ago, via FastEthernet0/0.146
      Route metric is 130, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

With a normal distribute-list filter referencing an access-list, there would be no way to distinguish between these two prefixes since they have the same address. Likewise with an administrative distance filter, there is no way to distinguish between them because they both are via the same ABR. By matching both the address and the next-hop, one prefix can be discarded in favor of the other.

```
Rack1R6#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#router ospf 1
Rack1R6(config-router)# distribute-list route-map
DENY_R3_LOOPBACK_FROM_R4 in
Rack1R6(config-router)#end

Rack1R6#clear ip route *
Rack1R6#show ip route 150.1.3.3
Routing entry for 150.1.3.3/32
  Known via "ospf 1", distance 110, metric 130, type intra area
  Redistributing via eigrp 10
  Advertised by eigrp 10 metric 100000 1000 255 1 1500
  Last update from 155.1.146.1 on FastEthernet0/0.146, 00:00:02 ago
  Routing Descriptor Blocks:
  * 155.1.146.1, from 150.1.3.3, 00:00:02 ago, via FastEthernet0/0.146
      Route metric is 130, traffic share count is 1

Rack1R6#ping 150.1.3.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.3.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 100/101/105
ms
```

The result in this case is that traffic no longer follows the path via R4, and is sent only to R1.

```
Rack1R6#traceroute 150.1.3.3

Type escape sequence to abort.
Tracing the route to 150.1.3.3

  1 155.1.146.1 4 msec 0 msec 0 msec
  2 155.1.0.5 32 msec 28 msec 32 msec
  3 155.1.0.3 61 msec *  57 msec
Rack1R6#
```

Copyright © 2010 Internetwork Expert

6.41 OSPF NSSA ABR External Prefix Filtering

- Disable R6's link to VLAN 146 and Loopback 0.
- Configure area 2 as an NSSA.
- Configure summarization on R3 so that devices outside of area 2 do not have a route to the network 200.0.0.0/24.
- This filter should not affect any other prefixes.

Configuration

```
R3:
router ospf 1
 area 2 nssa
 summary-address 200.0.0.0 255.255.255.0 not-advertise
R6:
interface FastEthernet0/0.146
 shutdown
!
interface Loopback0
 shutdown
!
router ospf 1
 area 2 nssa
SW1:
router ospf 1
 area 2 nssa
SW3:
router ospf 1
 area 2 nssa
```

Verification

Note

As previously discussed, only one ABR exit point out of an NSSA converts a Type-7 NSSA External LSA to a Type-5 External LSA, based on the translator election process. The NSSA translator can be configured to suppress the origination of the Type-5 External LSA into area 0 via the **`summary-address`** command.

Copyright © 2010 Internetwork Expert

As seen below, R5 currently learns the prefix 200.0.0.0/24 as a Type-5 External route, originated by the NSSA translator R3.

```
Rack1R5#show ip route | include E2(.*)via 155.1.0.3
O E2 200.0.0.0/24 [110/20] via 155.1.0.3, 00:00:37, Serial0/0/0
O E2    54.1.1.0 [110/20] via 155.1.0.3, 00:06:56, Serial0/0/0
O E2 200.0.1.0/24 [110/20] via 155.1.0.3, 00:06:56, Serial0/0/0
O E2 200.0.2.0/24 [110/20] via 155.1.0.3, 00:06:56, Serial0/0/0
O E2 200.0.3.0/24 [110/20] via 155.1.0.3, 00:06:56, Serial0/0/0
```

To suppress the advertisement of this prefix into area 0, R3 configures a **summary-address** with an identical mask of the original NSSA external route, but adds the **not-advertise** argument to the summary.

```
Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#router ospf 1
Rack1R3(config-router)#summary-address 200.0.0.0 255.255.255.0 not-
advertise
Rack1R3(config-router)#end
Rack1R3#

Rack1R5#show ip route | include E2(.*)via 155.1.0.3
O E2    54.1.1.0 [110/20] via 155.1.0.3, 00:07:13, Serial0/0/0
O E2 200.0.1.0/24 [110/20] via 155.1.0.3, 00:07:13, Serial0/0/0
O E2 200.0.2.0/24 [110/20] via 155.1.0.3, 00:07:13, Serial0/0/0
O E2 200.0.3.0/24 [110/20] via 155.1.0.3, 00:07:13, Serial0/0/0
```

The key difference with this filtering technique, versus the previously seen distribute-lists and administrative distance filters, is that the prefix is filtered out of the *database*, not just the routing table. This can be verified by R5's lack of LSA information as seen below.

```
Rack1R5#show ip ospf database external 200.0.0.0

            OSPF Router with ID (150.1.5.5) (Process ID 1)
Rack1R5#
```

Copyright © 2010 Internetwork Expert

6.42 OSPF Database Filtering

- Configure SW1 so that SW3 cannot learn any OSPF routes from SW1, but SW1 can still learn OSPF routes from SW3.
- Configure R5 so that R2 cannot learn from R5, but R5 can still learn OSPF routes from R2.

Configuration

```
R5:
router ospf 1
 neighbor 155.1.0.2 database-filter all out

SW1:
interface Vlan79
 ip ospf database-filter all out
```

Verification

> **Note**
>
> The OSPF command **`database-filter all out`** is similar in operation to the **`passive-interface`** command in RIPv2. This feature allows the formation of OSPF neighbors, since hello packets are not filtered out, but stops the advertisements of all LSAs out the interface or to the neighbor in question.
>
> As seen below, SW1 learns prefixes from SW3, however SW3 has no LSAs in the database, with the exception of locally originated ones.

```
Rack1SW1#show ip route ospf | include Vlan79
O       155.1.9.0/24 [110/2] via 155.1.79.9, 00:00:54, Vlan79
O       150.1.9.9/32 [110/2] via 155.1.79.9, 00:00:54, Vlan79

Rack1SW3#show ip route ospf

Rack1SW3#show ip ospf database

            OSPF Router with ID (150.1.9.9) (Process ID 1)

                Router Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum Link
count
150.1.9.9       150.1.9.9       140         0x80000002 0x006914 3

                Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
155.1.79.9      150.1.9.9       140         0x80000001 0x00DA6B
```

Likewise R5 learns LSAs from R2, but R2 cannot learn any LSAs from R5.

```
Rack1R2#show ip route ospf
     51.0.0.0/32 is subnetted, 1 subnets
O E2    51.51.51.51 [110/20] via 192.10.1.254, 00:00:30,
FastEthernet0/0

Rack1R5#show ip route ospf | include via 155.1.0.2
O E2    51.51.51.51 [110/20] via 155.1.0.2, 00:00:41, Serial0/0/0
O IA    155.1.23.0/24 [110/128] via 155.1.0.2, 00:00:51, Serial0/0/0
O       155.1.0.2/32 [110/64] via 155.1.0.2, 00:00:51, Serial0/0/0
O IA 192.10.1.0/24 [110/65] via 155.1.0.2, 00:00:51, Serial0/0/0

Rack1R2#show ip ospf database

            OSPF Router with ID (150.1.2.2) (Process ID 1)
                Router Link States (Area 0)
Link ID         ADV Router      Age         Seq#       Checksum Link
count
150.1.2.2       150.1.2.2       31          0x80000002 0x0041DE 3
                Summary Net Link States (Area 0)
Link ID         ADV Router      Age         Seq#       Checksum
155.1.23.0      150.1.2.2       32          0x80000001 0x00ADFF
192.10.1.0      150.1.2.2       32          0x80000001 0x00D8FB
                Summary ASB Link States (Area 0)
Link ID         ADV Router      Age         Seq#       Checksum
150.1.3.3       150.1.2.2       32          0x80000001 0x009F23
192.10.1.254    150.1.2.2       22          0x80000001 0x00D4FF
<output omitted>
```

Copyright © 2010 Internetwork Expert

6.43 OSPF Stub Router Advertisement

- Configure R4 to advertise the maximum metric value to all neighbors inside its Type-1 Router LSA.

Configuration

```
R4:
router ospf 1
 max-metric router-lsa
```

Verification

> **Note**
>
> The OSPF Stub Router Advertisement feature, not to be confused with OSPF stub areas, is used to prevent traffic black holes due to device adds or removes from the OSPF topology. Essentially, this feature causes the router to advertise a maximum metric for non-stub destinations, making it the worst cost path to all destinations. The result is that upon initializing the OSPF process, transit traffic will not flow through the stub router unless it is the only possible path. Once the routing domain is fully converged, the max metric value can be withdrawn, allowing normal forwarding to occur through the device.
>
> The **`max-metric router-lsa`** syntax unconditionally advertises the maximum metric until the command is removed, while the **`max-metric router-lsa on-startup wait-for-bgp`** option causes the router to advertise the maximum metric until BGP keepalives are received from all neighbors (keepalives in BGP indicate convergence is complete), and the **`max-metric router-lsa on-startup announce-time`** controls how long the router should advertise the maximum metric after a reload.
>
> In the below output we can see that R5 learns the prefix 155.1.146.0/24 from R4 via multiple links, all with a metric of 65.

```
Rack1R5#show ip route 155.1.146.0
Routing entry for 155.1.146.0/24
  Known via "ospf 1", distance 110, metric 65, type intra area
  Last update from 155.1.0.4 on Serial0/0/0, 00:00:08 ago
  Routing Descriptor Blocks:
    155.1.45.4, from 150.1.4.4, 00:00:08 ago, via Serial0/1/0
      Route metric is 65, traffic share count is 1
    155.1.0.4, from 150.1.4.4, 00:00:08 ago, via Serial0/0/0
      Route metric is 65, traffic share count is 1
  * 155.1.0.1, from 150.1.4.4, 00:00:08 ago, via Serial0/0/0
      Route metric is 65, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip ospf database router 150.1.4.4

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Router Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 19
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.4.4
  Advertising Router: 150.1.4.4
  LS Seq Number: 80000014
  Checksum: 0x238B
  Length: 96
  AS Boundary Router
  Number of Links: 6

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.146.4
     (Link Data) Router Interface address: 155.1.146.4
      Number of TOS metrics: 0
       TOS 0 Metrics: 1
<output omitted>
```

R4 currently advertises a metric of 1 to its transit link of VLAN 146.

After the max-metric router-lsa command is configured, R4 advertises the maximum cost of 65535 for VLAN 146, essentially making it the worst cost path.

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#router ospf 1
Rack1R4(config-router)#max-metric router-lsa
Rack1R4(config-router)#end
Rack1R4#

Rack1R5#show ip ospf database router 150.1.4.4

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Router Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 10
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 150.1.4.4
  Advertising Router: 150.1.4.4
  LS Seq Number: 80000015
  Checksum: 0xF935
  Length: 96
  AS Boundary Router
  Number of Links: 6

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 155.1.146.4
     (Link Data) Router Interface address: 155.1.146.4
      Number of TOS metrics: 0
       TOS 0 Metrics: 65535
```

The final result is that R5 no longer uses R4 to reach this destination.

```
Rack1R5#show ip route 155.1.146.0
Routing entry for 155.1.146.0/24
  Known via "ospf 1", distance 110, metric 65, type intra area
  Last update from 155.1.0.1 on Serial0/0, 00:00:04 ago
  Routing Descriptor Blocks:
  * 155.1.0.1, from 150.1.4.4, 00:00:04 ago, via Serial0/0
      Route metric is 65, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

6.44 OSPF Interface Timers

- Configure R5 to send OSPF hello packets out its link to the Frame Relay network every five seconds, and wait for seven seconds before declaring a neighbor down.
- Configure R4 to send OSPF hello packets every 250ms on the point-to-point link to R5.
- Configure R4 to send OSPF hello packets every 250ms on the point-to-point link to R5.

Configuration

```
R1:
interface Serial0/0
 ip ospf hello-interval 5
 ip ospf dead-interval 7

R2:
interface Serial0/0
 ip ospf hello-interval 5
 ip ospf dead-interval 7

R3:
interface Serial1/0
 ip ospf hello-interval 5
 ip ospf dead-interval 7

R4:
interface Serial0/0/0
 ip ospf hello-interval 5
 ip ospf dead-interval 7
!
interface Serial0/1/0
 ip ospf dead-interval minimal hello-multiplier 4

R5:
interface Serial0/0/0
 ip ospf hello-interval 5
 ip ospf dead-interval 7
!
interface Serial0/1/0
 ip ospf dead-interval minimal hello-multiplier 4
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> OSPF hello and dead timers must match for adjacency to occur. When the **ip ospf hello-interval** command is modified without the **ip ospf dead-interval** command, the dead timer is automatically set to be four times the configured hello.
>
> With the OSPF sub-second hello feature, the dead timer is set to one second, and the hello interval is set to 1000ms/hello-multiplier. Per the above command **ip ospf dead-interval minimal hello-multiplier 4**, the dead interval is one second, and the hello interval is 250ms (four times per second).

```
Rack1R5#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.4.4         0   FULL/  -        944 msec    155.1.45.4
Serial0/1/0
150.1.4.4         0   FULL/  -        00:00:02    155.1.0.4
Serial0/0/0
150.1.1.1         0   FULL/  -        00:00:02    155.1.0.1
Serial0/0/0
150.1.2.2         0   FULL/  -        00:00:05    155.1.0.2
Serial0/0/0
150.1.3.3         0   FULL/  -        00:00:05    155.1.0.3
Serial0/0/0
150.1.8.8         1   FULL/DR         00:00:38    155.1.58.8
FastEthernet0/0

Rack1R5#show ip ospf interface
Serial0/1/0 is up, line protocol is up
  Internet Address 155.1.45.5/24, Area 0
  Process ID 1, Router ID 150.1.5.5, Network Type POINT_TO_POINT, Cost:
64
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State POINT TO POINT,
  Timer intervals configured, Hello 250 msec, Dead 1, Wait 1,
Retransmit 5
    oob-resync timeout 40
    Hello due in 22 msec
<output omitted>
Serial0/0/0 is up, line protocol is up
  Internet Address 155.1.0.5/24, Area 0
  Process ID 1, Router ID 150.1.5.5, Network Type POINT_TO_MULTIPOINT,
Cost: 64
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State POINT TO MULTIPOINT,
  Timer intervals configured, Hello 5, Dead 7, Wait 7, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:03
<output omitte
```

6.45 OSPF Global Timers

- Modify R4 and R5's OSPF timers as follows:
  - Configure SPF throttling to start new re-calculation at least 100ms after a new LSA arrives. The second SPF calculation should occur in no less than 1s after the first one, and the maximum wait time should be no more than 10 seconds.
  - Configure LSA pacing to wait at least 50ms between consecutive link-state updates. LSA retransmissions should be paced at least 75ms apart.
  - Configure LSA throttling to generate subsequent LSAs after 10ms, to wait at least 4 seconds to generate the next LSA, and no more than 6 seconds between generation of the same LSA.
  - Configure LSA arrival throttling to wait 2 seconds between reception of the same LSA from a neighbor.
  - of the re-calculation at least 100ms after a new LSA arrives. The second SPF calculation should occur in no less than 1s after the first one, and the maximum wait time should be no more than 10 seconds.
- Configure R4 and R5 to assume that LSA transmission takes 2 seconds on the point-to-point link between them. LSA retransmission should occur if an acknowledgement is not received within 10 seconds over this link.

Configuration

```
R4:
router ospf 1
 timers throttle spf 100 1000 10000
 timers pacing flood 50
 timers pacing retransmission 75
 timers throttle lsa all 10 4000 6000
 timers lsa arrival 2000
!
interface Serial0/1
 ip ospf transmit-delay 2
 ip ospf retransmit-interval 10

R5:
router ospf 1
 timers throttle spf 100 1000 10000
 timers pacing flood 50
 timers pacing retransmission 75
 timers throttle lsa all 10 4000 6000
 timers lsa arrival 2000
!
interface Serial0/1/0
 ip ospf transmit-delay 2
 ip ospf retransmit-interval 10
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> OSPF packet and SPF pacing/throttling timers control how fast OSPF responds to convergence events. In the majority of deployments the default values should not need modification. Within the scope of the CCIE Lab Exam, determining which timers control which events should be self explanatory based on the usage guidelines of the commands in the OSPF command reference section of the documentation. These timers can be verified as follows.

```
Rack1R5#show ip ospf
 Routing Process "ospf 1" with ID 150.1.5.5
 Start time: 00:00:31.038, Time elapsed: 01:30:15.971
 Supports only single TOS(TOS0) routes
 Supports opaque LSA
 Supports Link-local Signaling (LLS)
 Supports area transit capability
 It is an area border router
 Router is not originating router-LSAs with maximum metric
 Initial SPF schedule delay 100 msecs
 Minimum hold time between two consecutive SPFs 1000 msecs
 Maximum wait time between two consecutive SPFs 10000 msecs
 Incremental-SPF disabled
 Initial LSA throttle delay 10 msecs
 Minimum hold time for LSA throttle 4000 msecs
 Maximum wait time for LSA throttle 6000 msecs
 Minimum LSA arrival 2000 msecs
 LSA group pacing timer 240 secs
 Interface flood pacing timer 50 msecs
 Retransmission pacing timer 75 msecs
 Number of external LSA 15. Checksum S
<output omitted>
```

Copyright © 2010 Internetwork Expert

6.46 OSPF Resource Limiting

- Configure R4 so that no more than 5000 LSAs can exist in the database.
- No more than 500 of these routes should be originated through redistribution.
- R4's OSPF process should be allowed to use up to 20% CPU utilization before allowing interrupts from higher priority processes.

Configuration

```
R4:
router ospf 1
 max-lsa 5000
 redistribute maximum-prefix 500
 process-min-time percent 20
```

Verification

> **Note**
>
> LSA and redistributed prefix limiting in OSPF is used to prevent against attacks or misconfigurations in the OSPF domain which could interrupt normal forwarding, such as if the full Internet BGP table is accidentally redistributed into IGP. Likewise the **process-min-time** command rate limits OSPF's CPU usage so that other processes, such as IP Input, can be serviced.

```
Rack1R4#show ip ospf
 Routing Process "ospf 1" with ID 150.1.4.4
 Start time: 00:00:31.423, Time elapsed: 01:39:24.674
 Supports only single TOS(TOS0) routes
 Supports opaque LSA
 Supports Link-local Signaling (LLS)
 Supports area transit capability
 Maximum number of non self-generated LSA allowed 5000
    Threshold for warning message 75%
    Ignore-time 5 minutes, reset-time 10 minutes
    Ignore-count allowed 5, current ignore-count 0
 It is an area border and autonomous system boundary router
 Redistributing External Routes from,
    rip, includes subnets in redistribution
    Maximum limit of redistributed prefixes 500
    Threshold for warning message 75%
<output omitted>
```

Copyright © 2010 Internetwork Expert

6.47 Miscellaneous OSPF Features

- Configure SW4 so that it goes not generate a log message upon receipt of a Type-6 LSA advertisement.
- Configure SW4 so that it does not take the MTU value into account when establishing adjacencies on its Port-Channel1 interface.
- Configure SW4 to reflect the following output:

```
Rack1SW4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address
Interface
Rack1SW2          1   FULL/BDR        00:00:38    155.1.108.8     Port-
channel1
```

Configuration

```
SW4:
ip host Rack1SW2 150.1.8.8
!
ip ospf name-lookup
!
interface Port-channel1
 ip ospf mtu-ignore
!
router ospf 1
 ignore lsa mospf
```

Verification

Note

Cisco's implementation of OSPFv2 does not support Multicast OSPF, which is advertised through LSA Type-6. Upon receipt of this LSA type from a non-Cisco OSPF router, a log message is generated. To disable this issue the ignore lsa mospf command under the OSPF process.

A neighbor relationship cannot occur if two OSPF neighbors have different MTU values on their interfaces. In the case that the MTU difference is by design, the interface level command **`ip ospf mtu-ignore`** removes this requirement from the adjacency establishment.

The **`ip ospf name-lookup`** command performs DNS resolution on the OSPF router-id value in show commands to simplify the identification of neighbors.

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
Rack1SW2          1   FULL/BDR        00:00:38    155.1.108.8     Port-channel1
```

Copyright © 2010 Internetwork Expert

BGP Solutions

> **Note**
>
> Load the *Initial BGP* initial configurations prior to starting.

7.1 Establishing iBGP Peerings

- Configure BGP on all internal devices using AS 100.
- Create a full mesh of iBGP peerings between these devices without using their Loopback interfaces.
- Advertise the Loopback 0 interfaces of these devices into BGP.
- Ensure full reachability to these Loopback 0 interfaces from all internal devices.

Configuration

```
R1:
router bgp 100
 network 150.1.1.0 mask 255.255.255.0
 neighbor 155.1.0.2 remote-as 100
 neighbor 155.1.0.3 remote-as 100
 neighbor 155.1.0.4 remote-as 100
 neighbor 155.1.0.5 remote-as 100
 neighbor 155.1.58.8 remote-as 100
 neighbor 155.1.67.7 remote-as 100
 neighbor 155.1.79.9 remote-as 100
 neighbor 155.1.108.10 remote-as 100
 neighbor 155.1.146.6 remote-as 100

R2:
router bgp 100
 network 150.1.2.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100
 neighbor 155.1.0.3 remote-as 100
 neighbor 155.1.0.4 remote-as 100
 neighbor 155.1.0.5 remote-as 100
 neighbor 155.1.37.7 remote-as 100
 neighbor 155.1.58.8 remote-as 100
 neighbor 155.1.79.9 remote-as 100
 neighbor 155.1.108.10 remote-as 100
 neighbor 155.1.146.6 remote-as 100
```

Copyright © 2010 Internetwork Expert

```
R3:
router bgp 100
 network 150.1.3.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100
 neighbor 155.1.0.2 remote-as 100
 neighbor 155.1.0.4 remote-as 100
 neighbor 155.1.0.5 remote-as 100
 neighbor 155.1.37.7 remote-as 100
 neighbor 155.1.58.8 remote-as 100
 neighbor 155.1.79.9 remote-as 100
 neighbor 155.1.108.10 remote-as 100
 neighbor 155.1.146.6 remote-as 100

R4:
router bgp 100
 network 150.1.4.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100
 neighbor 155.1.0.2 remote-as 100
 neighbor 155.1.0.3 remote-as 100
 neighbor 155.1.0.5 remote-as 100
 neighbor 155.1.58.8 remote-as 100
 neighbor 155.1.67.7 remote-as 100
 neighbor 155.1.79.9 remote-as 100
 neighbor 155.1.108.10 remote-as 100
 neighbor 155.1.146.6 remote-as 100

R5:
router bgp 100
 network 150.1.5.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100
 neighbor 155.1.0.2 remote-as 100
 neighbor 155.1.0.3 remote-as 100
 neighbor 155.1.0.4 remote-as 100
 neighbor 155.1.37.7 remote-as 100
 neighbor 155.1.58.8 remote-as 100
 neighbor 155.1.79.9 remote-as 100
 neighbor 155.1.108.10 remote-as 100
 neighbor 155.1.146.6 remote-as 100

R6:
router bgp 100
 network 150.1.6.0 mask 255.255.255.0
 neighbor 155.1.0.2 remote-as 100
 neighbor 155.1.0.3 remote-as 100
 neighbor 155.1.0.5 remote-as 100
 neighbor 155.1.58.8 remote-as 100
 neighbor 155.1.67.7 remote-as 100
 neighbor 155.1.79.9 remote-as 100
 neighbor 155.1.108.10 remote-as 100
 neighbor 155.1.146.1 remote-as 100
 neighbor 155.1.146.4 remote-as 100
```

Copyright © 2010 Internetwork Expert

```
SW1:
router bgp 100
 network 150.1.7.0 mask 255.255.255.0
 neighbor 155.1.0.5 remote-as 100
 neighbor 155.1.23.2 remote-as 100
 neighbor 155.1.37.3 remote-as 100
 neighbor 155.1.58.8 remote-as 100
 neighbor 155.1.67.6 remote-as 100
 neighbor 155.1.79.9 remote-as 100
 neighbor 155.1.108.10 remote-as 100
 neighbor 155.1.146.1 remote-as 100
 neighbor 155.1.146.4 remote-as 100

SW2:
router bgp 100
 network 150.1.8.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100
 neighbor 155.1.0.2 remote-as 100
 neighbor 155.1.0.3 remote-as 100
 neighbor 155.1.0.4 remote-as 100
 neighbor 155.1.37.7 remote-as 100
 neighbor 155.1.58.5 remote-as 100
 neighbor 155.1.79.9 remote-as 100
 neighbor 155.1.108.10 remote-as 100
 neighbor 155.1.146.6 remote-as 100

SW3:
router bgp 100
 network 150.1.9.0 mask 255.255.255.0
 neighbor 155.1.0.5 remote-as 100
 neighbor 155.1.23.2 remote-as 100
 neighbor 155.1.37.3 remote-as 100
 neighbor 155.1.58.8 remote-as 100
 neighbor 155.1.67.6 remote-as 100
 neighbor 155.1.79.7 remote-as 100
 neighbor 155.1.108.10 remote-as 100
 neighbor 155.1.146.1 remote-as 100
 neighbor 155.1.146.4 remote-as 100

SW4:
router bgp 100
 network 150.1.10.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100
 neighbor 155.1.0.2 remote-as 100
 neighbor 155.1.0.3 remote-as 100
 neighbor 155.1.0.4 remote-as 100
 neighbor 155.1.37.7 remote-as 100
 neighbor 155.1.58.5 remote-as 100
 neighbor 155.1.79.9 remote-as 100
 neighbor 155.1.108.8 remote-as 100
 neighbor 155.1.146.6 remote-as 100
```

Copyright © 2010 Internetwork Expert

Verification

Note

The first step in any BGP configuration is always to establish peering relationships between the BGP speaking devices. Recall that since BGP does not have its own transport protocol, underlying IGP reachability must already be established to allow the TCP port 179 sessions to be successful between neighbors.

BGP is a normal TCP application, which means that a TCP client initiates the session to the TCP server with a SYN packet going to the well known port of 179. If the BGP server is configured to accept the session, a reply with SYN/ACK comes from port 179, going to the high port that is negotiated between them. In the case that both BGP peers attempt to establish the connection at the same time, RFC 4271 (A Border Gateway Protocol 4) defines a "BGP Connection Collision Detection" mechanism, in which essentially the session originated from the device with the higher BGP router-id is maintained, and the secondary session is dropped.

The below debug output shows the step-by-step formation of the iBGP peering between R1 and R2. Note that access-list 100 is used to filter the debug output and only show the output pertinent to the BGP session between R1 and R2.

```
Rack1R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#access-list 100 permit tcp any host 155.1.0.2
Rack1R1(config)#access-list 100 permit tcp host 155.1.0.2 any
Rack1R1(config)#do debug ip packet detail 100
IP packet debugging is on (detailed) for access list 100
Rack1R1(config)#router bgp 100
Rack1R1(config-router)#neighbor 155.1.0.2 remote-as 100
Rack1R1(config-router)#
IP: tableid=0, s=155.1.0.2 (Serial0/0.1), d=155.1.0.1 (Serial0/0.1), routed via
RIB
IP: s=155.1.0.2 (Serial0/0.1), d=155.1.0.1 (Serial0/0.1), len 44, rcvd 3
    TCP src=51292, dst=179, seq=3391098696, ack=0, win=16384 SYN
```

R2 is already configured with the neighbor statement pointing towards R1, and a SYN is received from R2 to initiate the session. Note the source port of 51292, and the destination port of 179.

Copyright © 2010 Internetwork Expert

```
IP: tableid=0, s=155.1.0.1 (local), d=155.1.0.2 (Serial0/0.1), routed via FIB
IP: s=155.1.0.1 (local), d=155.1.0.2 (Serial0/0.1), len 44, sending
    TCP src=179, dst=51292, seq=1712691987, ack=3391098697, win=16384 ACK SYN
```

Since R1 has a neighbor statement pointing back towards the address 155.1.0.2, a reply of ACK/SYN is sent, with the source port of 179 and the destination port matching what R2 asked for, 51292. These two steps indicate that R2 is the client and R1 is the server.

```
IP: tableid=0, s=155.1.0.2 (Serial0/0.1), d=155.1.0.1 (Serial0/0.1), routed via
RIB
IP: s=155.1.0.2 (Serial0/0.1), d=155.1.0.1 (Serial0/0.1), len 40, rcvd 3
    TCP src=51292, dst=179, seq=3391098697, ack=1712691988, win=16384 ACK
```

R2 replies with ACK, completing the 3-way TCP handshake, and opening the session for BGP attribute negotiation. Only once necessary parameters are correctly negotiated, such as remote-as numbers, authentication, and address-family support, with the BGP session actually be declared up.

```
IP: tableid=0, s=155.1.0.2 (Serial0/0.1), d=155.1.0.1 (Serial0/0.1), routed via
RIB
IP: s=155.1.0.2 (Serial0/0.1), d=155.1.0.1 (Serial0/0.1), len 85, rcvd 3
    TCP src=51292, dst=179, seq=3391098697, ack=1712691988, win=16384 ACK PSH
<output omitted>
%BGP-5-ADJCHANGE: neighbor 155.1.0.2 Up
```

The details of the peering negotiation between R1 and R2, such as the router-id and timers, can be seen below. The neighbor capability of "Address family IPv4 Unicast: advertised and received" means that by default, they can exchange IPv4 routes, but not other routes such as IPv6 Unicast or MPLS.

```
Rack1R1(config-router)#do show ip bgp neighbor 155.1.0.2
BGP neighbor is 155.1.0.2,  remote AS 100, internal link
  BGP version 4, remote router ID 150.1.2.2
  BGP state = Established, up for 00:00:26
  Last read 00:00:26, last write 00:00:26, hold time is 180, keepalive interval
is 60 seconds
  Neighbor capabilities:
    Route refresh: advertised and received(old & new)
    Address family IPv4 Unicast: advertised and received
  Message statistics:
    InQ depth is 0
    OutQ depth is 0
                         Sent       Rcvd
    Opens:                  1          1
    Notifications:          0          0
    Updates:                1          1
    Keepalives:             3          3
    Route Refresh:          0          0
    Total:                  5          5
  Default minimum time between advertisement runs is 0 seconds
```

```
 For address family: IPv4 Unicast
  BGP table version 13, neighbor version 13/0
 Output queue size : 0
  Index 1, Offset 0, Mask 0x2
  1 update-group member
                                 Sent       Rcvd
  Prefix activity:               ----       ----
    Prefixes Current:               1          1 (Consumes 52 bytes)
    Prefixes Total:                 1          1
    Implicit Withdraw:              1          0
    Explicit Withdraw:              0          0
    Used as bestpath:             n/a          1
    Used as multipath:            n/a          0

                                  Outbound    Inbound
  Local Policy Denied Prefixes:   --------    -------
    Bestpath from this peer:             1        n/a
    Bestpath from iBGP peer:             8        n/a
    Total:                               9          0
  Number of NLRIs in the update sent: max 1, min 1
```

The IPv4 Unicast address family details above show that one prefix has been received from the neighbor, and one prefix has been advertised to the neighbor.

```
  Connections established 1; dropped 0
  Last reset never
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled, Mininum incoming TTL 0, Outgoing TTL 255
Local host: 155.1.0.1, Local port: 179
Foreign host: 155.1.0.2, Foreign port: 51292
<output omitted>
```

The TTL of the outbound session is set to 255, since this is an iBGP session. This means that the iBGP neighbors don't have to be directly connected, as long as IGP reachability exists between them. EBGP sessions have a TTL of 1 by default, which means that neighbors must be directly connected, unless further configuration is applied.

Also note the "Local port: 179" and "Foreign port: 51292". These essentially mean the source and destination ports from R1's perspective, which again enforces the notion that R1 is the server for this session, and R2 is the client.

The following **`show ip bgp summary`** output shows a concise aggregation of all configured neighbors, with the important fields being the local AS, router-id, table size in both prefixes and memory, peer addresses, remote ASes, neighbor uptime, and number of routes learned.

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip bgp summary
BGP router identifier 150.1.1.1, local AS number 100
BGP table version is 11, main routing table version 11
10 network entries using 1170 bytes of memory
10 path entries using 520 bytes of memory
3/2 BGP path/bestpath attribute entries using 372 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 2062 total bytes of memory
BGP activity 10/0 prefixes, 10/0 paths, scan interval 60 secs

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
155.1.0.2       4   100       7       7       11    0    0 00:03:46        1
155.1.0.3       4   100       7       7       11    0    0 00:03:14        1
155.1.0.4       4   100       7       7       11    0    0 00:03:13        1
155.1.0.5       4   100       7       7       11    0    0 00:03:12        1
155.1.58.8      4   100       5       6       11    0    0 00:02:41        1
155.1.67.7      4   100       5       6       11    0    0 00:02:53        1
155.1.79.9      4   100       5       6       11    0    0 00:02:39        1
155.1.108.10    4   100       5       6       11    0    0 00:02:27        1
155.1.146.6     4   100       7       7       11    0    0 00:03:11        1
```

Note that for devices running multiple address-families, such as IPv4 unicast and MPLS VPN routing, show commands are expressed as **`show bgp [AFI] [SAFI] [args]`**, such as the **`show bgp ipv4 unicast summary`** seen below. Although the output is the same as the above **`show ip bgp summary`**, it can quickly become confusing what output you are viewing in larger scale BGP deployments without this logical separation.

```
Rack1R1#show bgp ipv4 unicast summary
BGP router identifier 150.1.1.1, local AS number 100
BGP table version is 13, main routing table version 13
10 network entries using 1170 bytes of memory
10 path entries using 520 bytes of memory
3/2 BGP path/bestpath attribute entries using 372 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 2062 total bytes of memory
BGP activity 11/1 prefixes, 11/1 paths, scan interval 60 secs

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down
State/PfxRcd
155.1.0.2       4   100      38      38       13    0    0 00:33:36        1
155.1.0.3       4   100      43      43       13    0    0 00:39:50        1
155.1.0.4       4   100      43      43       13    0    0 00:39:49        1
155.1.0.5       4   100      43      43       13    0    0 00:39:48        1
155.1.58.8      4   100      42      43       13    0    0 00:39:17        1
155.1.67.7      4   100      41      43       13    0    0 00:39:29        1
155.1.79.9      4   100      42      43       13    0    0 00:39:15        1
155.1.108.10    4   100      42      43       13    0    0 00:39:03        1
155.1.146.6     4   100      43      43       13    0    0 00:39:47        1
```

Copyright © 2010 Internetwork Expert

Once the peering relationships are established, the actual BGP prefixes learned can be viewed with **show ip bgp**, or **show bgp ipv4 unicast**. This output is crucial to completely understand, especially when multiple paths to the same destination exist.

```
Rack1R1#show ip bgp
BGP table version is 11, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 150.1.1.0/24     0.0.0.0                  0         32768 i
*>i150.1.2.0/24     155.1.0.2                0    100      0 i
*>i150.1.3.0/24     155.1.0.3                0    100      0 i
*>i150.1.4.0/24     155.1.0.4                0    100      0 i
*>i150.1.5.0/24     155.1.0.5                0    100      0 i
*>i150.1.6.0/24     155.1.146.6              0    100      0 i
*>i150.1.7.0/24     155.1.67.7               0    100      0 i
*>i150.1.8.0/24     155.1.58.8               0    100      0 i
*>i150.1.9.0/24     155.1.79.9               0    100      0 i
*>i150.1.10.0/24    155.1.108.10             0    100      0 i

Rack1R1#show ip route bgp
     150.1.0.0/24 is subnetted, 10 subnets
B       150.1.7.0 [200/0] via 155.1.67.7, 00:01:45
B       150.1.6.0 [200/0] via 155.1.146.6, 00:01:48
B       150.1.5.0 [200/0] via 155.1.0.5, 00:01:46
B       150.1.4.0 [200/0] via 155.1.0.4, 00:01:46
B       150.1.3.0 [200/0] via 155.1.0.3, 00:01:46
B       150.1.2.0 [200/0] via 155.1.0.2, 00:01:52
B       150.1.10.0 [200/0] via 155.1.108.10, 00:01:46
B       150.1.9.0 [200/0] via 155.1.79.9, 00:01:46
B       150.1.8.0 [200/0] via 155.1.58.8, 00:01:49
```

One of the most important fields in the above outputs is the next-hop value. Note that this field is set to the peering address for the neighbor the route is learned from. For example the prefix 150.1.7.0/24 is via the next-hop 155.1.67.7. To reach this prefix, a recursive lookup must now be performed on the next-hop until an outgoing interface is found.

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip route 150.1.7.0
Routing entry for 150.1.7.0/24
  Known via "bgp 100", distance 200, metric 0, type internal
  Last update from 155.1.67.7 00:45:44 ago
  Routing Descriptor Blocks:
  * 155.1.67.7, from 155.1.67.7, 00:45:44 ago
      Route metric is 0, traffic share count is 1
      AS Hops 0

Rack1R1#show ip route 155.1.67.7
Routing entry for 155.1.67.0/24
  Known via "eigrp 100", distance 90, metric 30720, type internal
  Redistributing via eigrp 100
  Last update from 155.1.146.6 on FastEthernet0/0, 00:53:08 ago
  Routing Descriptor Blocks:
  * 155.1.146.6, from 155.1.146.6, 00:53:08 ago, via FastEthernet0/0
      Route metric is 30720, traffic share count is 1
      Total delay is 200 microseconds, minimum bandwidth is 100000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1

Rack1R1#show ip route 155.1.146.6
Routing entry for 155.1.146.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Redistributing via eigrp 100
  Routing Descriptor Blocks:
  * directly connected, via FastEthernet0/0
      Route metric is 0, traffic share count is 1
```

In the above case recursion towards 150.1.7.0/24 continues until the outgoing interface FastEthernet0/0 is found. Note that unless route-recursion towards the next-hop of a BGP prefix is successful, the route cannot be considered for best path selection, which also implies it cannot be installed in the IP routing table or advertised to any other BGP peer. This issue will be explored in detail in the coming tasks.

Also note that in this task, a full-mesh of iBGP peerings is established. This is due to the design requirement than an iBGP learned route cannot be advertised to another iBGP neighbor, unless exceptions such as route-reflection or confederation are implemented. The result seen is that the only routes advertised to the other BGP peers are the local Loopback 0 interfaces, but not any of the routes learned from the other neighbors. This also implies that in this design, if any individual peering breaks, connectivity between those peers also breaks.

```
Rack1R1#show ip bgp neighbors 155.1.0.2 advertised-routes
BGP table version is 13, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 150.1.1.0/24     0.0.0.0                  0         32768 i

Total number of prefixes 1
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.2 Establishing EBGP Peerings

- BB1 and BB3 are in AS 54.
- Configure EBGP peerings between R4 & BB3 and R6 & BB1 using their directly connected links.
- Advertise the external links between R4 & BB3 and R6 & BB1 into IGP.
- Ensure full reachability to all prefixes learned from AS 54 from all internal devices when sourcing traffic from the Loopback 0 interfaces.

Configuration

```
R4:
router eigrp 100
 passive-interface FastEthernet0/0
 network 204.12.1.0
!
router bgp 100
 neighbor 204.12.1.254 remote-as 54

R6:
router eigrp 100
 passive-interface Serial0/0/0
 network 54.0.0.0
!
router bgp 100
 neighbor 54.1.1.254 remote-as 54
```

Verification

> **Note**
>
> Configuration-wise, the only difference between an iBGP peering and an EBGP peering is that with an EBGP peering, the remote-as is different than the local-as. In practice however, many important attributes differ between the two. As seen in the following output, R4 knows that the peering occurs on a directly connected external link, and that the TTL should be 1 instead of 255. This implies that the neighbors must be directly connected for peering to be established, otherwise the TTL would be exceeded in transit and the TCP frames would be dropped.

```
Rack1R4#show ip bgp neighbors 204.12.1.254
BGP neighbor is 204.12.1.254,  remote AS 54, external link
  BGP version 4, remote router ID 31.3.0.1
  BGP state = Established, up for 00:02:04
<output omitted>

Connection is ECN Disabled, Mininum incoming TTL 0, Outgoing TTL 1
Local host: 204.12.1.4, Local port: 179
Foreign host: 204.12.1.254, Foreign port: 31806
<output omitted>
```

Copyright © 2010 Internetwork Expert

Since both BB1 and BB3 are in AS 54, the same view should be coming in from both peers. In the below output R4 sees 10 prefixes learned from BB3, and 11 prefixes learned from R6 (R6's Loopback0 plus the ten AS 54 routes).

```
Rack1R4#show ip bgp summary
BGP router identifier 150.1.4.4, local AS number 100
BGP table version is 31, main routing table version 31
20 network entries using 2340 bytes of memory
30 path entries using 1560 bytes of memory
9/6 BGP path/bestpath attribute entries using 1116 bytes of memory
2 BGP AS-PATH entries using 48 bytes of memory
1 BGP community entries using 24 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 5088 total bytes of memory
BGP activity 20/0 prefixes, 30/0 paths, scan interval 60 secs

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down
State/PfxRcd
155.1.0.1       4   100      67      71       31    0    0 01:03:50        1
155.1.0.2       4   100      67      71       31    0    0 01:04:00        1
155.1.0.3       4   100      67      71       31    0    0 01:03:48        1
155.1.0.5       4   100      67      71       31    0    0 01:03:36        1
155.1.58.8      4   100      65      71       31    0    0 01:03:34        1
155.1.67.7      4   100      65      71       31    0    0 01:03:43        1
155.1.79.9      4   100      65      71       31    0    0 01:03:26        1
155.1.108.10    4   100      65      71       31    0    0 01:03:06        1
155.1.146.6     4   100      71      71       31    0    0 01:03:35       11
204.12.1.254    4    54       9       9       31    0    0 00:00:10       10
```

Since duplicate routing information is learned, the BGP Bestpath Selection process must be run to choose one path over the other. The path that is active, known as the "best" path, can be seen from the greater-than sign (>) in the left column of the **`show ip bgp`** output.

```
Rack1R1#show ip bgp
BGP table version is 33, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   204.12.1.254             0    100      0 54 i
* i                 54.1.1.254               0    100      0 54 i
*>i28.119.17.0/24   204.12.1.254             0    100      0 54 i
* i                 54.1.1.254               0    100      0 54 i
*>i112.0.0.0        204.12.1.254             0    100      0 54 50 60 i
* i                 54.1.1.254               0    100      0 54 50 60 i
*>i113.0.0.0        204.12.1.254             0    100      0 54 50 60 i
* i                 54.1.1.254               0    100      0 54 50 60 i
*>i114.0.0.0        204.12.1.254             0    100      0 54 i
* i                 54.1.1.254               0    100      0 54 i
*>i115.0.0.0        204.12.1.254             0    100      0 54 i
* i                 54.1.1.254               0    100      0 54 i
```

Copyright © 2010 Internetwork Expert

The best path is the only route that is installed in the routing table, and the only route that is advertised to other BGP neighbors. From the above output of R1 we can see that the best path for all AS 54 routes is via 204.12.1.254. Why this selection occurs can be seen from the detailed output below.

```
Rack1R1#show ip bgp 112.0.0.0 255.0.0.0
BGP routing table entry for 112.0.0.0/8, version 29
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Not advertised to any peer
  54 50 60
    204.12.1.254 (metric 30720) from 155.1.0.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, internal, best
  54 50 60
    54.1.1.254 (metric 2172416) from 155.1.146.6 (150.1.6.6)
      Origin IGP, metric 0, localpref 100, valid, internal
```

Note the disconnect between the next-hop value and the neighbor that the prefix is learned from. The first highlighted value, 204.12.1.254, is the next-hop that R1 needs to be able to perform route-recursion towards in order to use the route. The next value, 155.1.0.4, is the peer address R1 uses to reach R4. The final address, 150.1.4.4, is the BGP router-id of R4. In this case the route through R4 is chosen over R6's route due to the lower metric of 30720 towards the next-hop 204.12.1.254, as opposed to the metric 2172416 towards 54.1.1.254. Bestpath selection will be discussed in detail in further tasks.

When R1 does its final lookup on 112.0.0.0/8, recursion continues towards the next-hop 204.12.1.254, resulting in the outgoing interface of FastEthernet0/0 towards 155.1.146.4.

```
Rack1R1#show ip route 112.0.0.0
Routing entry for 112.0.0.0/8
  Known via "bgp 100", distance 200, metric 0
  Tag 54, type internal
  Last update from 204.12.1.254 00:25:29 ago
  Routing Descriptor Blocks:
  * 204.12.1.254, from 155.1.0.4, 00:25:29 ago
      Route metric is 0, traffic share count is 1
      AS Hops 3
      Route tag 54

Rack1R1#show ip route 204.12.1.254
Routing entry for 204.12.1.0/24
  Known via "eigrp 100", distance 90, metric 30720, type internal
  Redistributing via eigrp 100
  Last update from 155.1.146.4 on FastEthernet0/0, 00:29:16 ago
  Routing Descriptor Blocks:
  * 155.1.146.4, from 155.1.146.4, 00:29:16 ago, via FastEthernet0/0
      Route metric is 30720, traffic share count is 1
      Total delay is 200 microseconds, minimum bandwidth is 100000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip route 155.1.146.4
Routing entry for 155.1.146.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Redistributing via eigrp 100
  Routing Descriptor Blocks:
  * directly connected, via FastEthernet0/0
      Route metric is 0, traffic share count is 1

Rack1R1#traceroute 112.0.0.1 source Loopback 0

Type escape sequence to abort.
Tracing the route to 112.0.0.1

  1 155.1.146.4 36 msec 36 msec 36 msec
  2 204.12.1.254 36 msec 37 msec 40 msec
  3 172.16.4.1 24 msec *  16 msec
```

Pitfall

Note that R4 and R6 did not update the next-hop values when advertising an EBGP learned route to their iBGP peers. In the case that the iBGP peers do not have a route to the next-hop value in the prefix, bestpath selection fails and the route cannot be used. Fixes for this problem will be explored in depth in further tasks.

Copyright © 2010 Internetwork Expert

7.3 BGP Update Source Modification

- Advertise the Loopback 0 interfaces of R4 and R5 into IGP.
- Modify the BGP peering between these devices so that if either the Frame Relay or Point-to-Point Serial link between them goes down, the BGP peering is not affected.

Configuration

```
R4:
router eigrp 100
 network 150.1.0.0
!
router bgp 100
 neighbor 150.1.5.5 remote-as 100
 neighbor 150.1.5.5 update-source Loopback0

R5:
router eigrp 100
 network 150.1.0.0
!
router bgp 100
 neighbor 150.1.4.4 remote-as 100
 neighbor 150.1.4.4 update-source Loopback0
```

Verification

Note

Since BGP peerings use TCP for transport, it is not a requirement that neighbors be directly connected. When neighbors are not directly connected, the choice of IP addresses used in peering can greatly affect the redundancy design of a BGP network. In the previous case, the peering between R4 and R5 was configured using their connected Frame Relay interface IP addresses. This implies that if the Frame Relay link were to go down, the BGP peering would also go down, even if alternate routes still existed between the devices. To fix this redundancy issue, the **`update-source`** for a BGP peering session can be changed on a per-neighbor basis.

Copyright © 2010 Internetwork Expert

Normally the IP source address used in a BGP packet is the IP address of the outgoing interface in the routing table. For example before the above modifications, R5 used the address 155.1.0.4 to reach R4 in the BGP peering.

```
Rack1R5#show ip route 155.1.0.4
Routing entry for 155.1.0.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Redistributing via eigrp 100
  Routing Descriptor Blocks:
  * directly connected, via Serial0/0/0
      Route metric is 0, traffic share count is 1

Rack1R5#show ip interface brief | include Serial0/0/0
Serial0/0/0                155.1.0.5       YES manual up
up
```

Based on the fact that R5 routes out Serial0/0 to reach 155.1.0.4, it means that the source address in the IP packet is 155.1.0.5. Observe what occurs with the BGP session when this interface is down.

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#shutdown
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to administratively down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed state to
down
Rack1R5(config-if)#end
%SYS-5-CONFIG_I: Configured from console by console
Rack1R5#show ip route 155.1.0.4
% Subnet not in table
BGP: 155.1.0.4 reset due to BGP Notification sent
%BGP-5-ADJCHANGE: neighbor 155.1.0.4 Down BGP Notification sent
```

With no route to 155.1.0.4, the BGP session is lost, even though the point-to-point link could have been used for rerouting. Now let's change the neighbor statement on R5 from 155.1.0.4 to 150.1.4.4, but not change the **update-source** yet.

Copyright © 2010 Internetwork Expert

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#access-list 100 permit tcp any host 150.1.4.4
Rack1R5(config)#access-list 100 permit tcp host 150.1.4.4 any
Rack1R5(config)#router bgp 100
Rack1R5(config-router)#no neighbor 155.1.0.4
%BGP-5-ADJCHANGE: neighbor 155.1.0.4 Down Neighbor deleted
Rack1R5(config-router)#neighbor 150.1.4.4 remote-as 100
Rack1R5(config-router)#end
Rack1R5#
Rack1R5#debug ip packet detail 100
IP packet debugging is on (detailed) for access list 100
Rack1R5#debug ip bgp
BGP debugging is on for address family: IPv4 Unicast
BGP: 150.1.4.4 open active, local address 155.1.45.5
IP: tableid=0, s=155.1.45.5 (local), d=150.1.4.4 (Serial0/1/0), routed via FIB
IP: s=155.1.45.5 (local), d=150.1.4.4 (Serial0/1/0), len 44, sending
    TCP src=21397, dst=179, seq=2795126014, ack=0, win=16384 SYN
IP: tableid=0, s=150.1.4.4 (Serial0/1/0), d=155.1.45.5 (Serial0/1/0), routed
via RIB
IP: s=150.1.4.4 (Serial0/1/0), d=155.1.45.5 (Serial0/1/0), len 40, rcvd 3
    TCP src=179, dst=21397, seq=0, ack=2795126015, win=0 ACK RST
BGP: 150.1.4.4 open failed: Connection refused by remote host, open active
delayed 34040ms (35000ms max, 28% jitter)
```

In the above output, only the peering statement on R5 has been updated towards R4's Loopback 0 network. Based on the debug we can see that the session attempts establishment, but R4 replies with ACK RST, refusing and closing the session. The reason why is that R5's route to 150.1.4.4 is out the Point-to-Point Serial link, causing the source IP address to be 155.1.45.5. However, R4 has its neighbor statement pointing at 155.1.0.5, not 155.1.45.5, so the connection is refused.

The key point to remember here is that *the TCP server of the BGP session must approve where the session is coming from*. If the SYN packet arrives from an address that is not in a neighbor statement, the connection is refused. To remedy this, both neighbors are modified with the new peering source and destination addresses.

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#router bgp 100
Rack1R4(config-router)#no neighbor 155.1.0.5
Rack1R4(config-router)#neighbor 150.1.5.5 remote-as 100
Rack1R4(config-router)#neighbor 150.1.5.5 update-source Loopback0
Rack1R4(config-router)#end
Rack1R4#
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#router bgp 100
Rack1R5(config-router)#neighbor 150.1.4.4 update-source Loopback0
Rack1R5(config-router)#end
Rack1R5#
Rack1R5#debug ip packet detail 100
IP packet debugging is on (detailed) for access list 100
IP: tableid=0, s=150.1.4.4 (Serial0/1/0), d=150.1.5.5 (Loopback0), routed via
RIB
IP: s=150.1.4.4 (Serial0/1/0), d=150.1.5.5, len 44, rcvd 4
    TCP src=33926, dst=179, seq=1620549161, ack=0, win=16384 SYN
```

Now R4 sends a SYN to 150.1.5.5 (R5's Loopback 0), sourced from 150.1.4.4 (R4's Loopback 0). Since R5 already has a neighbor statement for 150.1.4.4, SYN ACK is returned and the session opens.

```
IP: tableid=0, s=150.1.5.5 (local), d=150.1.4.4 (Serial0/0/0), routed via FIB
IP: s=150.1.5.5 (local), d=150.1.4.4 (Serial0/0/0), len 44, sending
    TCP src=179, dst=33926, seq=676456389, ack=1620549162, win=16384 ACK SYN
IP: tableid=0, s=150.1.4.4 (Serial0/1/0), d=150.1.5.5 (Loopback0), routed via
RIB
IP: s=150.1.4.4 (Serial0/1/0), d=150.1.5.5, len 40, rcvd 4
    TCP src=33926, dst=179, seq=1620549162, ack=676456390, win=16384 ACK
IP: tableid=0, s=150.1.4.4 (Serial0/1/0), d=150.1.5.5 (Loopback0), routed via
RIB
IP: s=150.1.4.4 (Serial0/1/0), d=150.1.5.5, len 85, rcvd 4
    TCP src=33926, dst=179, seq=1620549162, ack=676456390, win=16384 ACK PSH
<output omitted>
%BGP-5-ADJCHANGE: neighbor 150.1.4.4 Up
```

Now if one of the links between the neighbors goes down, the peering is simply rerouted based on the convergence of IGP.

```
Rack1R5#show ip route 150.1.4.4
Routing entry for 150.1.4.0/24
  Known via "eigrp 100", distance 90, metric 2297856, type internal
  Redistributing via eigrp 100
  Last update from 155.1.45.4 on Serial0/1, 01:48:03 ago
  Routing Descriptor Blocks:
  * 155.1.45.4, from 155.1.45.4, 01:48:03 ago, via Serial0/1/0
      Route metric is 2297856, traffic share count is 1
      Total delay is 25000 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
    155.1.0.4, from 155.1.0.4, 01:48:03 ago, via Serial0/0/0
      Route metric is 2297856, traffic share count is 1
      Total delay is 25000 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/1/0
Rack1R5(config-if)#shutdown
Rack1R5(config-if)#end
Rack1R5#
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.45.4 (Serial0/1/0) is down:
interface down
%SYS-5-CONFIG_I: Configured from console by console
%LINK-5-CHANGED: Interface Serial0/1/0, changed state to administratively down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0, changed state to
down
Rack1R5#show ip route 150.1.4.4
Routing entry for 150.1.4.0/24
  Known via "eigrp 100", distance 90, metric 2297856, type internal
  Redistributing via eigrp 100
  Last update from 155.1.0.4 on Serial0/0/0, 00:00:06 ago
  Routing Descriptor Blocks:
  * 155.1.0.4, from 155.1.0.4, 00:00:06 ago, via Serial0/0/0
      Route metric is 2297856, traffic share count is 1
      Total delay is 25000 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
```

Note that technically, only one neighbor needs to add the update-source command, as long as both agree on the destination of the peering. If R4 sets the update source to Loopback 0, but R5 does not, this will ensure that R4 is always the TCP client, and R5 is always the TCP server. In most designs the update sources are both modified for clarity.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.4 Multihop EBGP Peerings

- Create a new Loopback 1 interface on SW4 with the IP address 204.12.X.10/32, and advertise it into IGP.
- Configure an EBGP peering between SW4 and BB3 using this new interface as the source of the peering.

Configuration

```
SW4:
interface Loopback1
 ip address 204.12.1.10 255.255.255.255
!
router eigrp 100
 network 204.12.1.10 0.0.0.0
!
router bgp 100
 neighbor 204.12.1.254 remote-as 54
 neighbor 204.12.1.254 ebgp-multihop 255
 neighbor 204.12.1.254 update-source loopback1
```

Verification

> **Note**
>
> As seen in previous output, the default TTL for EBGP peers is 1. This means that non-directly connected EBGP peers cannot be established, since the TTL will expire in transit. By issuing the **`ebgp-multihop [ttl]`** command, the TTL can be increased to support this type of design.

```
Rack1SW4#show ip bgp summary | include 204.12.1.254
204.12.1.254    4    54      19      17       74    0    0 00:09:36       10

Rack1SW4#show ip bgp neighbors 204.12.1.254
BGP neighbor is 204.12.1.254,  remote AS 54, external link
  BGP version 4, remote router ID 31.3.0.1
  BGP state = Established, up for 00:01:23
  Last read 00:00:22, last write 00:00:51, hold time is 180, keepalive interval
is 60 seconds
<output omitted>
  External BGP neighbor may be up to 255 hops away.
  Transport(tcp) path-mtu-discovery is enabled
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Mininum incoming TTL 0, Outgoing TTL 255
Local host: 204.12.1.10, Local port: 179
Foreign host: 204.12.1.254, Foreign port: 32536
<output omitted>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.5 Neighbor Disable-Connected-Check

- Remove the all previous BGP configurations.
- Configure R1 & R4 in AS 100, and R5 in AS 200.
- Configure an iBGP peering between R1 and R4.
- Configure an EBGP peering between R4 and BB3, which is in AS 54.
- Configure an EBGP peering between R1 and R5.
- Configure an EBGP peering between R4 and R5 in such a way that the peering remains up as long as R4 has a connection to either the Frame Relay network or the Point-to-Point link to R5, but is torn down if both of these links are down.

Configuration

```
R1:
router bgp 100
 neighbor 155.1.0.5 remote-as 200
 neighbor 155.1.146.4 remote-as 100

R4:
router bgp 100
 neighbor 150.1.5.5 remote-as 200
 neighbor 150.1.5.5 disable-connected-check
 neighbor 150.1.5.5 update-source Loopback0
 neighbor 155.1.146.1 remote-as 100
 neighbor 204.12.1.254 remote-as 54

R5:
router bgp 200
 neighbor 150.1.4.4 remote-as 100
 neighbor 150.1.4.4 disable-connected-check
 neighbor 150.1.4.4 update-source Loopback0
 neighbor 155.1.0.1 remote-as 100
```

Copyright © 2010 Internetwork Expert

Verification

Note

Recall that with default EBGP sessions, a TTL of one prevents non-directly connected neighbors from forming. Additionally, IOS prevents the initiation of non-directly connected EBGP sessions when the TTL is one (i.e. multihop is not configured), because it assumes that the TTL will expire in transit.

As previously seen, one way of resolving this problem is to simply increase the TTL between the peers. In designs where the peers *are* connected, but the peering address is a Loopback instead of the connected interface between them, the **`disable-connected-check`** neighbor option may also be used.

Although similar in result to increasing the EBGP TTL, the difference between these features is that the **`disable-connected-check`** prevents cases where the EBGP session between two devices is routed over another transit router.

For example, in this case, R4 and R5 peer with each others Loopback 0 interfaces, but do not increase the TTL.

```
Rack1R4#show ip bgp neighbor 150.1.5.5 | include TTL
Connection is ECN Disabled, Mininum incoming TTL 0, Outgoing TTL 1

Rack1R4#show ip bgp summary
BGP router identifier 150.1.4.4, local AS number 100
BGP table version is 1, main routing table version 1

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down
State/PfxRcd
150.1.5.5       4   200       4       4        1    0    0 00:00:11        0
<output omitted>
```

Since the router doesn't decrement the TTL to a packet destined to itself, it technically only counts as one hop from R4 to R5's Loopback. Now let's see what happens when R4's direct links to R5 are down, but a route still remains to R5's Loopback 0.

Copyright © 2010 Internetwork Expert

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface Serial0/0/0
Rack1R4(config-if)#shutdown
Rack1R4(config-if)#interface Serial0/1/0
Rack1R4(config-if)#shutdown
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.0.5 (Serial0/0/0.1) is down:
interface down
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to administratively down
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.45.5 (Serial0/1/0) is down:
interface down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed state to
down
Rack1R4(config-if)#end
Rack1R4#
%BGP-5-ADJCHANGE: neighbor 150.1.5.5 Down BGP Notification sent
%BGP-3-NOTIFICATION: sent to neighbor 150.1.5.5 4/0 (hold time expired) 0 bytes

Rack1R4#show ip route 150.1.5.5
Routing entry for 150.1.5.0/24
  Known via "eigrp 100", distance 90, metric 2300416, type internal
  Redistributing via eigrp 100
  Last update from 155.1.146.1 on FastEthernet0/1, 00:25:39 ago
  Routing Descriptor Blocks:
  * 155.1.146.1, from 155.1.146.1, 00:25:39 ago, via FastEthernet0/1
      Route metric is 2300416, traffic share count is 1
      Total delay is 25100 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
```

Even though a route remains between R4 and R5's Loopback 0 networks, the BGP peering is declared down. The reason why can be seen from the BGP and ICMP debugs below.

```
Rack1R4#debug ip bgp
BGP debugging is on for address family: IPv4 Unicast
Rack1R4#debug ip icmp
ICMP packet debugging is on
ICMP: time exceeded rcvd from 155.1.146.1
ICMP: time exceeded rcvd from 155.1.146.1
ICMP: time exceeded rcvd from 155.1.146.1
BGP: 150.1.5.5 open failed: Connection timed out; remote host not responding,
open active delayed 31249ms (35000ms max, 28% jitter)
```

Since the TTL of the EBGP packet is one, time exceeds as the packet transits through R1. Note that R4 and R5 continue to attempt setup of the BGP peering since the connected check is disabled. This design can be desirable in cases where you do *not* want to reroute the BGP session around network failures. Now let's see the difference if we had changed the TTL to support the multihop peering.

Copyright © 2010 Internetwork Expert

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface serial0/0/0
Rack1R4(config-if)#no shutdown
Rack1R4(config-if)#interface serial0/1/0
Rack1R4(config-if)#no shutdown
%LINK-3-UPDOWN: Interface Serial0/0/0, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed state to
up
%LINK-3-UPDOWN: Interface Serial0/1/0, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0, changed state to
up
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.0.5 (Serial0/0/0.1) is up:
new adjacency
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.45.5 (Serial0/1/0) is up:
new adjacency
Rack1R4(config-if)#router bgp 100
Rack1R4(config-router)#no neighbor 150.1.5.5 disable-connected-check
Rack1R4(config-router)#neighbor 150.1.5.5 ebgp-multihop
Rack1R4(config-router)#end

Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#router bgp 200
Rack1R5(config-router)#no neighbor 150.1.4.4 disable-connected-check
Rack1R5(config-router)#neighbor 150.1.4.4 ebgp-multihop
Rack1R5(config-router)#end
Rack1R5#
```

R4's links to R5 are brought back up, and the disable-connected-check command is replaced with the ebgp-multihop [255] command. The BGP peering comes up, and when R4's connected links to R5 are brought down again, the BGP peering continues to stay up.

```
Rack1R4#
%BGP-5-ADJCHANGE: neighbor 150.1.5.5 Up
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface Serial0/0/0
Rack1R4(config-if)#shutdown
Rack1R4(config-if)#interface Serial0/1/0
Rack1R4(config-if)#shutdown
Rack1R4(config-if)#end
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to administratively down
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 100: Neighbor 155.1.45.5 (Serial0/1/0) is down:
interface down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed state to
down
%SYS-5-CONFIG_I: Configured from console by console
%LINK-5-CHANGED: Interface Serial0/1/0, changed state to administratively down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0, changed state to
down
Rack1R4(config-if)#end
```

Copyright © 2010 Internetwork Expert

The final result is that although R5 only has one path to reach AS 54, which is via the Frame Relay circuit to R1, there is redundant routing information in the BGP table, as seen below.

```
Rack1R5#show ip bgp
BGP table version is 21, local router ID is 150.1.5.5
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  28.119.16.0/24   150.1.4.4                              0 100 54 i
*>                  155.1.0.1                              0 100 54 i
*  28.119.17.0/24   150.1.4.4                              0 100 54 i
*>                  155.1.0.1                              0 100 54 i
*  112.0.0.0        150.1.4.4                              0 100 54 50 60 i
*>                  155.1.0.1                              0 100 54 50 60 i
*  113.0.0.0        150.1.4.4                              0 100 54 50 60 i
*>                  155.1.0.1                              0 100 54 50 60 i
*  114.0.0.0        150.1.4.4                              0 100 54 i
*>                  155.1.0.1                              0 100 54 i
*  115.0.0.0        150.1.4.4                              0 100 54 i
*>                  155.1.0.1                              0 100 54 i
*  116.0.0.0        150.1.4.4                              0 100 54 i
*>                  155.1.0.1                              0 100 54 i
*  117.0.0.0        150.1.4.4                              0 100 54 i
*>                  155.1.0.1                              0 100 54 i
*  118.0.0.0        150.1.4.4                              0 100 54 i
<output omitted>
```

Considering that the public Internet BGP table can grow to millions of paths to hundreds of thousands of prefixes, each separate view that the router has to maintain can require an extremely large amount of memory and CPU resources.

Therefore in this particular design, choosing to disable the connected check versus enabling a multihop peering can save resources on R5 while a failure scenario is in affect.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.6 Authenticating BGP Peerings

- Remove the previous BGP configuration on R2.
- Configure R2 in BGP AS 200, and configure an EBGP peering with BB2.
- BB2 is in AS 254.
- Authenticate this peering with the password "CISCO".

Configuration

```
R2:
router bgp 200
 neighbor 192.10.1.254 remote-as 254
 neighbor 192.10.1.254 password CISCO
```

Verification

> **Note**
>
> BGP authentication is implemented through TCP Option 19, the MD5 hash. Configuration is very straightforward, and requires only the additional neighbor statement with the password option. If BGP peering occurs, authentication is successful.

```
Rack1R2#show ip bgp neighbors 192.10.1.254 | include state|Flags
  BGP state = Established, up for 00:01:19
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Flags: active open, nagle, md5
```

Authentication failure results in a log message, and failure of the peering to establish.

```
Rack1R2#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R2(config)#router bgp 200
Rack1R2(config-router)#neighbor 192.10.1.254 password WRONG
Rack1R2(config-router)#end
Rack1R2#
%SYS-5-CONFIG_I: Configured from console by console
Rack1R2#clear ip bgp *
%BGP-5-ADJCHANGE: neighbor 192.10.1.254 Down User reset
%TCP-6-BADAUTH: Invalid MD5 digest from 192.10.1.254(179) to 192.10.1.2(19205)
Rack1R2#
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.7 iBGP Route Reflection

- Remove the previous BGP configuration on all devices.
- Remove the advertisement of R4 and R5's Loopback 0 networks into IGP.
- Configure BGP on all internal devices using AS 100.
- Configure iBGP peerings from R1 to all other devices in the internal network.
- Configure EBGP peerings between R4 & BB3 and R6 & BB1 using their directly connected links; BB1 and BB3 are in AS 54.
- Advertise the Loopback0 interfaces of these devices into BGP.
- Ensure full reachability to these Loopback 0 interfaces from all internal devices, and to all prefixes learned from AS 54 from all internal devices when sourcing traffic from the Loopback 0 interfaces.

Configuration

```
R1:
router bgp 100
 network 150.1.1.0 mask 255.255.255.0
 neighbor 155.1.0.2 remote-as 100
 neighbor 155.1.0.3 remote-as 100
 neighbor 155.1.0.4 remote-as 100
 neighbor 155.1.0.5 remote-as 100
 neighbor 155.1.58.8 remote-as 100
 neighbor 155.1.67.7 remote-as 100
 neighbor 155.1.79.9 remote-as 100
 neighbor 155.1.108.10 remote-as 100
 neighbor 155.1.146.6 remote-as 100
 neighbor 155.1.0.2 route-reflector-client
 neighbor 155.1.0.3 route-reflector-client
 neighbor 155.1.0.4 route-reflector-client
 neighbor 155.1.0.5 route-reflector-client
 neighbor 155.1.58.8 route-reflector-client
 neighbor 155.1.67.7 route-reflector-client
 neighbor 155.1.79.9 route-reflector-client
 neighbor 155.1.108.10 route-reflector-client
 neighbor 155.1.146.6 route-reflector-client

R2:
router bgp 100
 network 150.1.2.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100

R3:
router bgp 100
 network 150.1.3.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100
```

Copyright © 2010 Internetwork Expert

```
R4:
router bgp 100
 network 150.1.4.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100
 neighbor 204.12.1.254 remote-as 54

R5:
router bgp 100
 network 150.1.5.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100

R6:
router bgp 100
 network 150.1.6.0 mask 255.255.255.0
 neighbor 155.1.146.1 remote-as 100
 neighbor 54.1.1.254 remote-as 54

SW1:
router bgp 100
 network 150.1.7.0 mask 255.255.255.0
 neighbor 155.1.146.1 remote-as 100

SW2:
router bgp 100
 network 150.1.8.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100

SW3:
router bgp 100
 network 150.1.9.0 mask 255.255.255.0
 neighbor 155.1.146.1 remote-as 100

SW4:
router bgp 100
 network 150.1.10.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100
```

Verification

Note

BGP route reflectors, as defined in RFC 2796, are used in large scale iBGP deployments to reduce the need for n*(n-1)/2 fully meshed peerings. Route reflectors accomplish this by creating an exception for passing advertisements between peers. Specifically this is implemented as follows.

Copyright © 2010 Internetwork Expert

A route reflector can have three types of peers, EBGP peers, client peers, and non-client peers. EBGP peers are neighbors in a different AS number, including peers in different Sub-ASes in confederation. Client peers are iBGP neighbors that have the **route-reflector-client** statement configured. Non-client peers are normal iBGP peers that do not have the **route-reflector-client** statement configured. Routing advertisements sent from the route reflector must conform to the following three rules.

1. Routes learned from EBGP peers can be sent to other EBGP peers, clients, and non-clients.

2. Routes learned from client peers can be sent to EBGP peers, other client peers, and non-clients.

3. Routes learned from non-client peers can be sent to EBGP peers, and client peers, *but not other non-clients.*

In the simplest of route-reflection designs, a central peering point is chosen for all devices in the iBGP domain, and all peers of this device are defined as clients. In this particular example, R1 is configured in this manner. When R1 receives routes from its iBGP peers, they are tagged internally as being received from a client peer, and are candidate to be advertised on to everyone. In the below output we see R1 learning R2's Loopback 0 network, with the RR-client attribute set.

```
Rack1R1#show ip bgp 150.1.2.0 255.255.255.0
BGP routing table entry for 150.1.2.0/24, version 3
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  Local, (Received from a RR-client)
    155.1.0.2 from 155.1.0.2 (150.1.2.2)
      Origin IGP, metric 0, localpref 100, valid, internal, best
```

Copyright © 2010 Internetwork Expert

When a route is advertised, or "reflected", from the route reflector to a client or non-client, BGP attributes such as the next-hop value are not updated.

```
Rack1R1#show ip bgp neighbors 155.1.0.3 advertised-routes
BGP table version is 15, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 150.1.1.0/24     0.0.0.0                  0         32768 i
*>i150.1.2.0/24     155.1.0.2                0    100      0 i
*>i150.1.3.0/24     155.1.0.3                0    100      0 i
*>i150.1.4.0/24     155.1.0.4                0    100      0 i
*>i150.1.5.0/24     155.1.0.5                0    100      0 i
*>i150.1.6.0/24     155.1.146.6              0    100      0 i
*>i150.1.7.0/24     155.1.67.7               0    100      0 i
*>i150.1.8.0/24     155.1.58.8               0    100      0 i
*>i150.1.9.0/24     155.1.79.9               0    100      0 i
*>i150.1.10.0/24    155.1.108.10             0    100      0 i

Total number of prefixes 10
```

Instead, two new attributes are added onto the reflected prefix, the Originator ID and the Cluster List.

```
Rack1R3#show ip bgp 150.1.2.0 255.255.255.0
BGP routing table entry for 150.1.2.0/24, version 10
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Not advertised to any peer
  Local
    155.1.0.2 from 155.1.0.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Originator: 150.1.2.2, Cluster list: 150.1.1.1
```

Recall that previously, loop prevention in iBGP is achieved simply by not advertising routes learned from one iBGP neighbor to another. Since route-reflection violates this rule, new loop prevention must be implemented.

The first of these, the Originator ID, is set by the route reflector as the BGP router-id of the neighbor it learned the prefix from. For the above prefix, this is the BGP router-id of R2, as seen in the parenthesis of the **`show ip bgp 150.1.2.0 255.255.255.0`** on R1. When any BGP speaker learns a route from an iBGP neighbor, and the Originator ID matches their own local router-id, the route is discarded. This is why it is essential that the BGP router-id value be unique throughout the entire routing domain, just like in OSPF and EIGRP.

Copyright © 2010 Internetwork Expert

The second new attribute, the Cluster List, contains the Cluster-IDs of the route reflectors that the route transited through in the network. Unless the **bgp cluster-id** command is manually configured under the BGP routing process, the value defaults to the router-id of the route reflector. In the above case the Cluster List contains just the router-id of R1, 150.1.1.1.

This attribute is used to prevent loops *between* route-reflectors, when a hierarchical design called Clustering is implemented. A "cluster" in BGP is defined as a route-reflector and its clients, and will be explored in more detail in later tasks. When a route-reflector learns a route from an iBGP peer, and the Cluster List includes its own Cluster-ID, the route is discarded.

Note that since the other attributes in the prefix are not updated by the route reflector, the reflector is analogous to the DR in OSPF, and in many cases traffic does not physically transit through this device. Instead the reflector is simply a central point for network control traffic, but not necessarily an aggregation point for traffic. This can be demonstrated in this design by the traffic flow between R2 and SW4, as seen below.

```
Rack1R2#show ip bgp 150.1.10.0 255.255.255.0
BGP routing table entry for 150.1.10.0/24, version 6
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Not advertised to any peer
  Local
    155.1.108.10 (metric 2174976) from 155.1.0.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Originator: 204.12.1.10, Cluster list: 150.1.1.1
```

R2 learns the prefix 150.1.10.0/24 from R1, but with a next-hop value of 155.1.108.10. Now a recursive lookup must be performed on 155.1.108.10 until the outgoing interface is found.

Copyright © 2010 Internetwork Expert

```
Rack1R2#show ip route 155.1.108.10
Routing entry for 155.1.108.0/24
  Known via "eigrp 100", distance 90, metric 2174976, type internal
  Redistributing via eigrp 100
  Last update from 155.1.0.5 on Serial0/0.1, 00:40:51 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 155.1.0.5, 00:40:51 ago, via Serial0/0.1
      Route metric is 2174976, traffic share count is 1
      Total delay is 20200 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
```

```
Rack1R2#show ip route 155.1.0.5
Routing entry for 155.1.0.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Redistributing via eigrp 100
  Routing Descriptor Blocks:
  * directly connected, via Serial0/0.1
      Route metric is 0, traffic share count is 1
```

155.1.108.10 is known via IGP from R5, out Serial0/0.1. The traceroute indicates that the traffic flow does not pass through the route reflector, even though the BGP control traffic did.

```
Rack1R2#traceroute 150.1.10.10

Type escape sequence to abort.
Tracing the route to 150.1.10.10

  1 155.1.0.5 28 msec 28 msec 32 msec
  2 155.1.58.8 28 msec 32 msec 28 msec
  3 155.1.108.10 28 msec *  28 msec
```

Copyright © 2010 Internetwork Expert

7.8 Large Scale iBGP Route Reflection with Clusters

- Remove the previous BGP configuration on all devices.
- Configure R2 in BGP AS 200, and configure an EBGP peering with BB2.
- BB2 is in AS 254, and is authenticating this peering with the password "CISCO".
- Configure BGP on all other internal devices using AS 100.
- Configure a BGP cluster between R1, R4, and R6 as follows:
  - R1 should be the route-reflector, and peer with R4 and R6.
  - R4 and R6 should peer with BB3 and BB1 respectively, who are in AS 54.
  - Use the cluster-id 150.X.1.1.
- Configure a BGP cluster between R3, SW1, and SW3 as follows:
  - R3 should be the route-reflector, and peer with SW1 and SW3.
  - Use the cluster-id 150.X.3.3.
- Configure a BGP cluster between R5, SW2, and SW4 as follows:
  - R5 should be the route-reflector, and peer with SW2 and SW4.
  - Use the cluster-id 150.X.5.5.
- R1, R3, and R5 should all peer with each other in a full-mesh, but should not propagate updates between clusters.
- Configure EBGP peerings between R2 & R3 and R2 & R5.
- Advertise the Loopback0 interfaces of all devices into BGP.
- Ensure full reachability to these Loopback0 interfaces from all internal devices, and to all prefixes learned from AS 54 and 254 from all internal devices when sourcing traffic from the Loopback0 interfaces.

Configuration

```
R1:
router bgp 100
 bgp cluster-id 150.1.1.1
 network 150.1.1.0 mask 255.255.255.0
 neighbor 155.1.146.4 remote-as 100
 neighbor 155.1.146.6 remote-as 100
 neighbor 155.1.146.4 route-reflector-client
 neighbor 155.1.146.6 route-reflector-client
 neighbor 155.1.0.5 remote-as 100
 neighbor 155.1.13.3 remote-as 100
```

Copyright © 2010 Internetwork Expert

```
R2:
router bgp 200
 network 150.1.2.0 mask 255.255.255.0
 neighbor 192.10.1.254 remote-as 254
 neighbor 192.10.1.254 password CISCO
 neighbor 155.1.23.3 remote-as 100
 neighbor 155.1.0.5 remote-as 100

R3:
router bgp 100
 bgp cluster-id 150.1.3.3
 network 150.1.3.0 mask 255.255.255.0
 neighbor 155.1.37.7 remote-as 100
 neighbor 155.1.79.9 remote-as 100
 neighbor 155.1.37.7 route-reflector-client
 neighbor 155.1.79.9 route-reflector-client
 neighbor 155.1.13.1 remote-as 100
 neighbor 155.1.0.5 remote-as 100
 neighbor 155.1.23.2 remote-as 200

R4:
router bgp 100
 network 150.1.4.0 mask 255.255.255.0
 neighbor 155.1.146.1 remote-as 100
 neighbor 204.12.1.254 remote-as 54

R5:
router bgp 100
 bgp cluster-id 150.1.5.5
 network 150.1.5.0 mask 255.255.255.0
 neighbor 155.1.58.8 remote-as 100
 neighbor 155.1.108.10 remote-as 100
 neighbor 155.1.58.8 route-reflector-client
 neighbor 155.1.108.10 route-reflector-client
 neighbor 155.1.0.1 remote-as 100
 neighbor 155.1.0.3 remote-as 100
 neighbor 155.1.0.2 remote-as 200

R6:
router bgp 100
 network 150.1.6.0 mask 255.255.255.0
 neighbor 155.1.146.1 remote-as 100
 neighbor 54.1.1.254 remote-as 54

SW1:
router bgp 100
 network 150.1.7.0 mask 255.255.255.0
 neighbor 155.1.37.3 remote-as 100

SW2:
router bgp 100
 network 150.1.8.0 mask 255.255.255.0
 neighbor 155.1.58.5 remote-as 100
```

Copyright © 2010 Internetwork Expert

```
SW3:
router bgp 100
 network 150.1.9.0 mask 255.255.255.0
 neighbor 155.1.37.3 remote-as 100

SW4:
router bgp 100
 network 150.1.10.0 mask 255.255.255.0
 neighbor 155.1.58.5 remote-as 100
```

Verification

Note

The term "cluster" refers to a route-reflector and its clients, or multiple route-reflectors that service the same clients. Clustering is used to create a balance between the amount of BGP control traffic that must be maintained through peering, and redundancy. Many large scale Service Providers use clustering to create hierarchy in their iBGP designs, by constraining clusters to different geographic regions, which reduces the amount of long-haul BGP peerings that must occur.

In this particular example, three clusters are created. Each cluster is serviced by one route-reflector, R1, R3, or R5. Inside the cluster, all iBGP peers are configured as clients of the route reflector. Between clusters, however, the route reflectors are non-clients of each other. This design helps to limit redundant updating in the BGP control plane, but sacrifices redundancy. To illustrate this let's follow the path of an update through the iBGP domain, and observe how different failure scenarios affect reachability to it.

```
Rack1R4#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 5
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     2
  54 50 60
    204.12.1.254 from 204.12.1.254 (31.3.0.1)
      Origin IGP, localpref 100, valid, external, best
```

R4 learns the prefix 112.0.0.0/8 from its EBGP peer, BB3, and passes this route onto its iBGP peer, R1.

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 38
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2
  54 50 60, (Received from a RR-client)
    204.12.1.254 (metric 30720) from 155.1.146.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, internal, best
  54 50 60, (Received from a RR-client)
    54.1.1.254 (metric 2172416) from 155.1.146.6 (150.1.6.6)
      Origin IGP, metric 0, localpref 100, valid, internal
```

R1 learns the route from R4 with a next-hop of 204.12.1.254. It also learns the identical route from R6 with a next-hop of 54.1.1.254. Since the metric towards 204.12.1.254 is lower, this route is chosen as best, and is candidate to be advertised. Also note that R1 says that these two prefixes are learned from route reflector clients.

Since these routes come from client peers, they are candidate to be advertised to EBGP peers, other clients, and non-clients. In this case R1 advertises the path through R4 to its non-clients, R3 and R5.

```
Rack1R3#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 33
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           3
  54 50 60
    204.12.1.254 (metric 33536) from 155.1.13.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Originator: 150.1.4.4, Cluster list: 150.1.1.1

Rack1R5#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 33
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           3
  54 50 60
    204.12.1.254 (metric 2172416) from 155.1.0.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Originator: 150.1.4.4, Cluster list: 150.1.1.1
```

R3 and R5 learn the prefix from R1, with the new attributes Originator ID set to 150.1.4.4 (R4), and Cluster List including 150.1.1.1 (R1). Since from R3 and R5's perspective these prefixes were not learned from route reflector clients, they are only candidate to be advertised to EBGP peers and client peers. From R5, the result is that the prefix is advertised to R2, SW2, and SW4, but not to R3.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip bgp neighbors 155.1.0.2 advertised-routes | include 112.0.0.0
*>i112.0.0.0        204.12.1.254             0    100      0 54 50 60 i

Rack1R5#$ neighbors 155.1.58.8 advertised-routes | include 112.0.0.0
*>i112.0.0.0        204.12.1.254             0    100      0 54 50 60 i

Rack1R5#$ neighbors 155.1.108.10 advertised-routes | include 112.0.0.0
*>i112.0.0.0        204.12.1.254             0    100      0 54 50 60 i

Rack1R5#show ip bgp neighbors 155.1.0.3 advertised-routes | include 112.0.0.0
```

This is the behavior we should expect, since the inter-cluster peerings are non-client peerings. In essence, the only routes that are advertised between clusters are routes that came from within the cluster, or from EBGP peers.

```
Rack1SW4#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 24
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Not advertised to any peer
  54 50 60
    204.12.1.254 (metric 2175232) from 155.1.58.5 (150.1.5.5)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Originator: 150.1.4.4, Cluster list: 150.1.5.5, 150.1.1.1
```

SW4 learns the route from R5, with the Originator ID still set to R4, but with the Cluster List now including both the Cluster-IDs of R5 and R1. Like AS-Path, the Cluster List is populated with the newest route reflector on the left. The other attributes, such as the next-hop value, have not been updated. This means that SW4 must perform a recursive lookup towards 204.12.1.254.

```
Rack1SW4#show ip route 204.12.1.254
Routing entry for 204.12.1.0/24
  Known via "eigrp 100", distance 90, metric 2175232, type internal
  Redistributing via eigrp 100
  Last update from 155.1.108.8 on Port-channel1, 00:30:56 ago
  Routing Descriptor Blocks:
  * 155.1.108.8, from 155.1.108.8, 00:30:56 ago, via Port-channel1
      Route metric is 2175232, traffic share count is 1
      Total delay is 20210 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 253/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 3

Rack1SW4#show ip route 155.1.108.8
Routing entry for 155.1.108.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Redistributing via eigrp 100
  Routing Descriptor Blocks:
  * directly connected, via Port-channel1
      Route metric is 0, traffic share count is 1
```

Copyright © 2010 Internetwork Expert

The outgoing interface is found, and a traceroute indicates the full end-to-end path. Note that R1 is not in the traffic path, even though it is in the BGP control traffic path. This is because an independent IGP lookup is performed towards the next-hop, which is unrelated to the path of the BGP peerings.

```
Rack1SW4#traceroute
Protocol [ip]:
Target IP address: 112.0.0.1
Source address: 150.1.10.10
Numeric display [n]:
Timeout in seconds [3]:
Probe count [3]:
Minimum Time to Live [1]:
Maximum Time to Live [30]:
Port Number [33434]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Type escape sequence to abort.
Tracing the route to 112.0.0.1

  1 155.1.108.8 4 msec 0 msec 0 msec
  2 155.1.58.5 0 msec 4 msec 0 msec
  3 155.1.0.4 20 msec 16 msec 20 msec
  4 204.12.1.254 24 msec 24 msec 20 msec
  5 172.16.4.1 44 msec *  36 msec
```

At this point full reachability should be obtained to all BGP learned prefixes when traffic is sourced from the Loopback 0 networks. Now let's look at the case where a failure occurs on the Frame Relay PVC between R1 and R5.

```
Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial0/0/0
Rack1R5(config-if)#no frame-relay map ip 155.1.0.1
<output omitted>
Rack1R5#show ip bgp summary | include 155.1.0.1
155.1.0.1       4   100      73      70        0    0    0 00:10:03 Active
Rack1R5#
```

The frame-relay map statement for R1 is withdrawn from R5, simulating a circuit failure. Shortly after the BGP peer is declared down. Now look at the changes in the propagation of the prefix 112.0.0.0/8

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 38
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2
  54 50 60, (Received from a RR-client)
    204.12.1.254 (metric 30720) from 155.1.146.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, internal, best
  54 50 60, (Received from a RR-client)
    54.1.1.254 (metric 2172416) from 155.1.146.6 (150.1.6.6)
      Origin IGP, metric 0, localpref 100, valid, internal
```

R1 still has the best route to 112.0.0.0/8 installed via R4, and advertises the prefix to R3. The advertisement to R5 cannot occur since the peering is down.

```
Rack1R3#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 33
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           3
  54 50 60
    204.12.1.254 (metric 33536) from 155.1.13.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Originator: 150.1.4.4, Cluster list: 150.1.1.1
```

R3 receives the prefix from R1, but cannot advertise it to R5, since both R1 and R5 are non-clients.

```
Rack1R3#show ip bgp neighbors 155.1.0.5 advertised-routes
BGP table version is 34, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 150.1.2.0/24     155.1.23.2               0             0 200 i
*> 150.1.3.0/24     0.0.0.0                  0         32768 i
*>i150.1.7.0/24     155.1.37.7               0    100      0 i
*>i150.1.9.0/24     155.1.79.9               0    100      0 i
*> 205.90.31.0      155.1.23.2                             0 200 254 ?
*> 220.20.3.0       155.1.23.2                             0 200 254 ?
*> 222.22.2.0       155.1.23.2                             0 200 254 ?
```

The final result is that R5 does not have the prefix installed, which implies that SW2 and SW4 likewise do not have the prefix.

```
Rack1R5#show ip bgp 112.0.0.0
% Network not in table

Rack1SW4#ping 112.0.0.1 source Loopback0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 112.0.0.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.10.10
.....
Success rate is 0 percent (0/5)
```

Copyright © 2010 Internetwork Expert

Now let's modify the peerings between R1, R3, and R5 so that they *are* clients of each other, and see how this affects redundancy.

```
Rack1R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#router bgp 100
Rack1R1(config-router)#neighbor 155.1.0.5 route-reflector-client
Rack1R1(config-router)#neighbor 155.1.13.3 route-reflector-client

Rack1R3#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#router bgp 100
Rack1R3(config-router)#neighbor 155.1.0.5 route-reflector-client
Rack1R3(config-router)#neighbor 155.1.13.1 route-reflector-client

Rack1R5#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#router bgp 100
Rack1R5(config-router)#neighbor 155.1.0.1 route-reflector-client
Rack1R5(config-router)#neighbor 155.1.0.3 route-reflector-client
```

Now that R1 is a client of R3, R3 can advertise prefixes received from R1 to R5, and vice versa. The resulting change is that R3 now advertises 112.0.0.0/8 to R5.

```
Rack1R3#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 91
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
     1           3
  54 50 60, (Received from a RR-client)
    204.12.1.254 (metric 33536) from 155.1.13.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Originator: 150.1.4.4, Cluster list: 150.1.1.1

Rack1R3#show ip bgp neighbors 155.1.0.5 advertised-routes | include 112.0.0.0
*>i112.0.0.0        204.12.1.254             0    100      0 54 50 60 i
```

Likewise R5 can now continue to propagate the prefix onto SW2 and SW4, and connectivity is restored.

```
Rack1SW4#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 113
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Flag: 0x820
  Not advertised to any peer
  54 50 60
    204.12.1.254 (metric 2175232) from 155.1.58.5 (150.1.5.5)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Originator: 150.1.4.4, Cluster list: 150.1.5.5, 150.1.3.3, 150.1.1.1
```

Copyright © 2010 Internetwork Expert

```
Rack1SW4#ping 112.0.0.1 source Loopback0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 112.0.0.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.10.10
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 76/76/76 ms
```

So if without client peerings between R1, R3, and R5, redundancy suffers, what is the disadvantage of configuring all the inter-cluster peers as clients of each other? The answer, is route replication overhead.

With R5's connection to R1 working, let's look at the advertisement of 112.0.0.0/8 again. To start, R4 learns the prefix 112.0.0.0/8 from BB3, and advertises it onto R1.

```
Rack1R4#show ip bgp neighbors 155.1.146.1 advertised-routes | include 112.0.0.0
*> 112.0.0.0        204.12.1.254                           0 54 50 60 i
```

R1 reflects this route to both R3 and R5, as expected.

```
Rack1R1#show ip bgp neighbors 155.1.13.3 advertised-routes | include 112.0.0.0
*>i112.0.0.0        204.12.1.254             0    100      0 54 50 60 i

Rack1R1#show ip bgp neighbors 155.1.0.5 advertised-routes | include 112.0.0.0
*>i112.0.0.0        204.12.1.254             0    100      0 54 50 60 i
```

Since R1 is now a client of both R3 and R5, both R3 and R5 advertise the prefix to each other, as expected.

```
Rack1R3#show ip bgp neighbors 155.1.0.5 advertised-routes | include 112.0.0.0
*>i112.0.0.0        204.12.1.254             0    100      0 54 50 60 i

Rack1R5#show ip bgp neighbors 155.1.0.3 advertised-routes | include 112.0.0.0
*>i112.0.0.0        204.12.1.254             0    100      0 54 50 60 i
```

Now, oddly enough however, R3 and R5 each take the advertisement they are getting in from R1, and send it back to R1. This is where the advertisement feedback occurs.

```
Rack1R3#show ip bgp neighbors 155.1.13.1 advertised-routes | include 112.0.0.0
*>i112.0.0.0        204.12.1.254             0    100      0 54 50 60 i

Rack1R5#show ip bgp neighbors 155.1.0.1 advertised-routes | include 112.0.0.0
*>i112.0.0.0        204.12.1.254             0    100      0 54 50 60 i
```

Copyright © 2010 Internetwork Expert

To see this update loop in action, limit the routes that R4 receives from BB3 to just 112.0.0.0/8, and shut down R1's peering to R6. Next, request a route refresh with the **clear ip bgp** command while **debug ip bgp** is enabled.

```
Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#ip prefix-list ONLY_112 permit 112.0.0.0/8
Rack1R4(config)#router bgp 100
Rack1R4(config-router)#neighbor 204.12.1.254 prefix-list ONLY_112 in
Rack1R4(config-router)#end
Rack1R4#clear ip bgp * in

Rack1R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#router bgp 100
Rack1R1(config-router)#neighbor 155.1.146.6 shutdown
%BGP-5-ADJCHANGE: neighbor 155.1.146.6 Down Admin. shutdown
Rack1R1(config-router)#end
Rack1R1#debug ip bgp updates
BGP updates debugging is on for address family: IPv4 Unicast
Rack1R1#clear ip bgp 155.1.146.4
Rack1R1#
BGP(0): no valid path for 112.0.0.0/8
BGP(0): no valid path for 150.1.4.0/24
%BGP-5-ADJCHANGE: neighbor 155.1.146.4 Down User reset
BGP(0): nettable_walker 112.0.0.0/8 no best path
BGP(0): nettable_walker 150.1.4.0/24 no best path
BGP(0): 155.1.0.5 send unreachable 150.1.4.0/24
BGP(0): 155.1.0.5 send UPDATE 150.1.4.0/24 -- unreachable
BGP(0): 155.1.0.5 send UPDATE 112.0.0.0/8 -- unreachable
```

With R1's peering to R4 down, an UPDATE message is sent to withdraw the routes that were reachable via R4.

```
BGP: 155.1.13.3 Route Reflector cluster loop; Received cluster-id 150.1.1.1
BGP(0): 155.1.13.3 rcv UPDATE w/ attr: nexthop 155.1.146.4, origin i, localpref
100, metric 0, originator 150.1.4.4, clusterlist 150.1.3.3 150.1.5.5 0.0.0.0,
path , community , extended community
BGP(0): 155.1.13.3 rcv UPDATE about 150.1.4.0/24 -- DENIED due to: CLUSTERLIST
contains our own cluster ID;
BGP: 155.1.13.3 Route Reflector cluster loop; Received cluster-id 150.1.1.1
BGP(0): 155.1.13.3 rcv UPDATE w/ attr: nexthop 204.12.1.254, origin i,
localpref 100, metric 0, originator 150.1.4.4, clusterlist 150.
Rack1R1#1.3.3 150.1.5.5 0.0.0.0, path 54 50 60, community , extended community
BGP(0): 155.1.13.3 rcv UPDATE about 112.0.0.0/8 -- DENIED due to: CLUSTERLIST
contains our own cluster ID;
BGP(0): updgrp 2 - 155.1.0.5 updates replicated for neighbors: 155.1.13.3
BGP(0): 155.1.0.5 rcv UPDATE about 112.0.0.0/8 -- withdrawn
BGP(0): 155.1.0.5 rcv UPDATE about 150.1.4.0/24 -- withdrawn
BGP(0): 155.1.13.3 rcv UPDATE about 150.1.4.0/24 -- withdrawn
BGP(0): 155.1.13.3 rcv UPDATE about 112.0.0.0/8 -- withdrawn
```

Copyright © 2010 Internetwork Expert

When R1 issued a withdraw message for 112.0.0.0/8, it received looped updates back in from both R3 and R5. Ultimately these were blocked because R1 saw its own Cluster ID in the Cluster List. The same occurs when the peering to R4 comes back up.

```
%BGP-5-ADJCHANGE: neighbor 155.1.146.4 Up
<output omitted>
BGP(0): 155.1.146.4 rcvd UPDATE w/ attr: nexthop 155.1.146.4, origin i,
localpref 100, metric 0
BGP(0): 155.1.146.4 rcvd 150.1.4.0/24
BGP(0): 155.1.146.4 rcvd UPDATE w/ attr: nexthop 204.12.1.254, origin i,
localpref 100, metric 0, path 54 50 60
BGP(0): 155.1.146.4 rcvd 112.0.0.0/8
BGP(0): Revise route installing 1 of 1 routes for 112.0.0.0/8 ->
204.12.1.254(main) to main IP table
BGP(0): Revise route installing 1 of 1 routes for 150.1.4.0/24 ->
155.1.146.4(main) to main IP table
BGP(0): updgrp 2 - 155.1.146.4 updates replicated for neighbors:
BGP(0): 155.1.146.4 send UPDATE (format) 150.1.4.0/24, next 155.1.146.4, metric
0, path Local
BGP(0): 155.1.146.4 send UPDATE (format) 112.0.0.0/8, next 204.12.1.254, metric
0, path 54 50 60
BGP(0): updgrp 2 - 155.1.146.4 updates replicated for neighbors: 155.1.0.5
155.1.13.3
```

R1 gets the routes 150.1.4.0/24 and 112.0.0.0/8 from R4, and replicates them to R3 and R5.

```
BGP: 155.1.13.3 Route Reflector cluster loop; Received cluster-id 150.1.1.1
BGP(0): 155.1.13.3 rcv UPDATE w/ attr: nexthop 155.1.146.4, origin i, localpref
100, metric 0, originator 150.1.4.4, clusterlist 150.1.3.3 150.1.5.5, path ,
community , extended community
BGP(0): 155.1.13.3 rcv UPDATE about 150.1.4.0/24 -- DENIED due to: CLUSTERLIST
contains our own cluster ID;
BGP: 155.1.13.3 Route Reflector cluster loop; Received cluster-id 150.1.1.1
BGP(0): 155.1.13.3 rcv UPDATE w/ attr: nexthop 204.12.1.254, origin i,
localpref 100, metric 0, originator 150.1.4.4, clusterlist 150.1.3.3 150.1.5.5,
path 54 50 60, community , extended community
BGP(0): 155.1.13.3 rcv UPDATE about 112.0.0.0/8 -- DENIED due to: CLUSTERLIST
contains our own cluster ID;
BGP: 155.1.0.5 Route Reflector cluster loop; Received cluster-id 150.1.1.1
BGP(0): 155.1.0.5 rcv UPDATE w/ attr: nexthop 155.1.146.4, origin i, localpref
100, metric 0, originator 150.1.4.4, clusterlist 150.1.5.5 150.1.5.5, path ,
community , extended community
BGP(0): 155.1.0.5 rcv UPDATE about 150.1.4.0/24 -- DENIED due to: CLUSTERLIST
contains our own cluster ID;
BGP: 155.1.0.5 Route Reflector cluster loop; Received cluster-id 150.1.1.1
BGP(0): 155.1.0.5 rcv UPDATE w/ attr: nexthop 204.12.1.254, origin i, localpref
100, metric 0, originator 150.1.4.4, clusterlist 150.1.5.5 150.1.5.5, path 54
50 60, community , extended community
BGP(0): 155.1.0.5 rcv UPDATE about 112.0.0.0/8 -- DENIED due to: CLUSTERLIST
contains our own cluster ID;
```

Copyright © 2010 Internetwork Expert

Once the peering to R4 comes back up, we can see likewise that another update loop occurs between R1, R3, and R5. Luckily since the Cluster List contains R1's Cluster ID, the loop is broken.

Ultimately the choice between making the inter-cluster peerings client or non-client peerings depends on the redundancy design. We saw first that with them configured as non-client peerings, certain network failures could have caused traffic black holes, even though there were alternate viable paths to the destinations.

On the other hand, with the inter-cluster peerings configured as client peerings, each update message sent out results in a feedback loop of update messages received back in. With only a few prefixes in the BGP table this may not seem like a big issue, but with the hundreds of thousands of prefixes in the Internet BGP table, these type of loops can quickly cause utilization problems.

Copyright © 2010 Internetwork Expert

7.9 iBGP Confederation

- Remove the BGP configuration of all devices in AS 100.
- Configure a BGP Confederation Sub-AS between R1, R4, and R6 as follows:
  - Use the Sub-AS number 65146.
  - Use the Public AS number 100.
  - Configure full-mesh peerings between R1, R4, and R6.
  - R4 and R6 should peer with BB3 and BB1 respectively.
- Configure a BGP Confederation Sub-AS between R3, SW1, and SW3 as follows:
  - Use the Sub-AS number 65379.
  - Use the Public AS number 100.
  - Configure full-mesh peerings between R3, SW1, and SW3.
- Configure a BGP Confederation Sub-AS between R5, SW2, and SW4 as follows:
  - Use the Sub-AS number 65508.
  - Use the Public AS number 100.
  - R5 should be a route-reflector, and peer with SW2 and SW4.
- R1, R3, and R5 should all peer with each other in a full-mesh.
- Configure EBGP peerings between R2 & R3 and R2 & R5.
- Advertise the Loopback 0 interfaces of all devices into BGP.
- Ensure full reachability to these Loopback 0 interfaces from all internal devices, and to all prefixes learned from AS 54 and 254 from all internal devices when sourcing traffic from the Loopback 0 interfaces.

Configuration

```
R1:
router bgp 65146
 bgp confederation identifier 100
 bgp confederation peers 65379 65508
 network 150.1.1.0 mask 255.255.255.0
 neighbor 155.1.0.5 remote-as 65508
 neighbor 155.1.13.3 remote-as 65379
 neighbor 155.1.146.4 remote-as 65146
 neighbor 155.1.146.6 remote-as 65146
```

Copyright © 2010 Internetwork Expert

```
R2:
router bgp 200
 network 150.1.2.0 mask 255.255.255.0
 neighbor 155.1.0.5 remote-as 100
 neighbor 155.1.23.3 remote-as 100
 neighbor 192.10.1.254 remote-as 254
 neighbor 192.10.1.254 password CISCO

R3:
router bgp 65379
 bgp confederation identifier 100
 bgp confederation peers 65146 65508
 network 150.1.3.0 mask 255.255.255.0
 neighbor 155.1.0.5 remote-as 65508
 neighbor 155.1.13.1 remote-as 65146
 neighbor 155.1.23.2 remote-as 200
 neighbor 155.1.37.7 remote-as 65379
 neighbor 155.1.79.9 remote-as 65379

R4:
router bgp 65146
 bgp confederation identifier 100
 network 150.1.4.0 mask 255.255.255.0
 neighbor 155.1.146.1 remote-as 65146
 neighbor 155.1.146.6 remote-as 65146
 neighbor 204.12.1.254 remote-as 54

R5:
router bgp 65508
 bgp confederation identifier 100
 bgp confederation peers 65146 65379
 network 150.1.5.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 65146
 neighbor 155.1.0.2 remote-as 200
 neighbor 155.1.0.3 remote-as 65379
 neighbor 155.1.58.8 remote-as 65508
 neighbor 155.1.58.8 route-reflector-client
 neighbor 155.1.108.10 remote-as 65508
 neighbor 155.1.108.10 route-reflector-client

R6:
router bgp 65146
 bgp confederation identifier 100
 network 150.1.6.0 mask 255.255.255.0
 neighbor 54.1.1.254 remote-as 54
 neighbor 155.1.146.1 remote-as 65146
 neighbor 155.1.146.4 remote-as 65146
```

Copyright © 2010 Internetwork Expert

```
SW1:
router bgp 65379
 bgp confederation identifier 100
 network 150.1.7.0 mask 255.255.255.0
 neighbor 155.1.37.3 remote-as 65379
 neighbor 155.1.79.9 remote-as 65379

SW2:
router bgp 65508
 bgp confederation identifier 100
 network 150.1.8.0 mask 255.255.255.0
 neighbor 155.1.58.5 remote-as 65508

SW3:
router bgp 65379
 bgp confederation identifier 100
 network 150.1.9.0 mask 255.255.255.0
 neighbor 155.1.37.3 remote-as 65379
 neighbor 155.1.79.7 remote-as 65379

SW4:
router bgp 65508
 bgp confederation identifier 100
 network 150.1.10.0 mask 255.255.255.0
 neighbor 155.1.58.5 remote-as 65508
```

Verification

Note

Defined in RFC 5065, *Autonomous System Confederations for BGP*, confederations, like route reflectors, are used to reduce the need for fully meshed iBGP peerings in large scale deployments. In confederation, a public AS is split into smaller Sub Autonomous Systems (Sub-ASes), which exhibit a hybrid behavior of both iBGP and EBGP. Inside a Sub-AS, the requirement for either fully meshed iBGP peerings or route reflection still applies, but between Sub-ASes, EBGP advertisement rules apply.

First, the BGP process is initialized using the Sub-AS number, as opposed to the normal initialization with the public AS number. Sub-AS numbers are typically in the private AS range (64512 – 65535), but technically can be any valid number, private or not. Next, the **`bgp confederation identifier`** informs the router that it is part of a confederation, with the ID number being its public AS number.

Any neighbor whose remote-as matches either the local Sub-AS, or a number listed in the **`bgp confederation peers`** statement, is considered to be part of the confederation. In the latter case, these peers are considered "confederation EBGP peers". For other neighbors whose AS matches neither the local Sub-AS nor a confederation peer AS, they are considered normal EBGP neighbors.

Copyright © 2010 Internetwork Expert

The most notable difference between confederation implementations, versus route reflection or full mesh, is the introduction of a new BGP attribute known as the AS_CONFED_SET. The confederation set, or simply confed set, is an unordered list of Sub-ASes that is prepended onto the normal AS-Path of a BGP prefix as it is passed between Sub-ASes. The key with the confed set, however, is that it is stripped and replaced by the confederation identifier when a prefix is advertised to a true EBGP peer. Take the following output from this example.

```
Rack1R1#show ip bgp 150.1.6.0
BGP routing table entry for 150.1.6.0/24, version 15
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     2
  Local
    155.1.146.6 from 155.1.146.6 (150.1.6.6)
      Origin IGP, metric 0, localpref 100, valid, confed-internal, best
```

R1 learns the prefix 150.1.6.0 from R6 with a next-hop value of 155.1.146.6. Since both R1 and R6 are in the same Sub-AS of 65146, R1 tags this route as *confed-internal*, i.e. coming from an iBGP peer. The AS-Path attribute of this prefix is not modified during R6's advertisement to R1, since they are iBGP peers. When R1 passes this route onto R3 or R5, whom are both in different Sub-ASes, the confed set is populated with R1's Sub-AS number of 65146. This can be clearly seen as a separate denotation from the normal AS-Path information, as it is listed in parentheses.

```
Rack1R5#show ip bgp 150.1.6.0
BGP routing table entry for 150.1.6.0/24, version 17
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2           3
  (65379 65146)
    155.1.146.6 (metric 2172416) from 155.1.0.3 (150.1.3.3)
      Origin IGP, metric 0, localpref 100, valid, confed-external
  (65146)
    155.1.146.6 (metric 2172416) from 155.1.0.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, confed-external, best
```

In the above case R5 learns the prefix 150.1.6.0 from both R1 and R3. Since both of these neighbors are in different Sub-ASes, they are considered *confed-external* peers, or confederation EBGP peers. Note that the path through R3 contains both R1's Sub-AS and R3's Sub-AS, while the path through R1 only contains R1's Sub-AS.

Copyright © 2010 Internetwork Expert

Note that although prefixes are passed between Sub-ASes based on EBGP advertisement rules, the majority of attributes are left unchanged, with the most notable being that ***the next-hop value is not modified***. As we can see in the below output, SW4 sees the routes learned from AS 54 with a next-hop value of 204.12.1.254, which is the unmodified next-hop of the link between R4 and BB3. Likewise prefixes such as 150.1.7.0 and 150.1.9.0 have next-hop values of the originators into the public AS, not the neighbor that SW4 is learning them from.

```
Rack1SW4#show ip bgp
BGP table version is 25, local router ID is 204.12.1.10
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   204.12.1.254             0    100      0 (65146) 54 i
*>i28.119.17.0/24   204.12.1.254             0    100      0 (65146) 54 i
*>i112.0.0.0        204.12.1.254             0    100      0 (65146) 54 50 60 i
*>i113.0.0.0        204.12.1.254             0    100      0 (65146) 54 50 60 i
*>i114.0.0.0        204.12.1.254             0    100      0 (65146) 54 i
*>i115.0.0.0        204.12.1.254             0    100      0 (65146) 54 i
*>i116.0.0.0        204.12.1.254             0    100      0 (65146) 54 i
*>i117.0.0.0        204.12.1.254             0    100      0 (65146) 54 i
*>i118.0.0.0        204.12.1.254             0    100      0 (65146) 54 i
*>i119.0.0.0        204.12.1.254             0    100      0 (65146) 54 i
*>i150.1.1.0/24     155.1.0.1                0    100      0 (65146) i
*>i150.1.2.0/24     155.1.0.2                0    100      0 200 i
*>i150.1.3.0/24     155.1.0.3                0    100      0 (65379) i
*>i150.1.4.0/24     155.1.146.4              0    100      0 (65146) i
*>i150.1.5.0/24     155.1.58.5               0    100      0 i
*>i150.1.6.0/24     155.1.146.6              0    100      0 (65146) i
*>i150.1.7.0/24     155.1.37.7               0    100      0 (65146 65379) i
*>i150.1.8.0/24     155.1.58.8               0    100      0 i
*>i150.1.9.0/24     155.1.79.9               0    100      0 (65146 65379) i
*> 150.1.10.0/24    0.0.0.0                  0         32768 i
*>i205.90.31.0      155.1.0.2                0    100      0 200 254 ?
*>i220.20.3.0       155.1.0.2                0    100      0 200 254 ?
*>i222.22.2.0       155.1.0.2                0    100      0 200 254 ?
```

Copyright © 2010 Internetwork Expert

When prefixes are advertised outside of the public AS, the confed set is stripped and replaced with the public AS number. In this manner devices outside of the confederation do not know the confederation's internal routing topology.

```
Rack1R2#show ip bgp | include 150.1.
BGP table version is 24, local router ID is 150.1.2.2
*> 150.1.1.0/24     155.1.23.3                             0 100 i
*> 150.1.2.0/24     0.0.0.0                  0         32768 i
*> 150.1.3.0/24     155.1.23.3               0             0 100 i
*> 150.1.4.0/24     155.1.23.3                             0 100 i
*> 150.1.5.0/24     155.1.23.3                             0 100 i
*> 150.1.6.0/24     155.1.23.3                             0 100 i
*> 150.1.7.0/24     155.1.23.3                             0 100 i
*> 150.1.8.0/24     155.1.23.3                             0 100 i
*> 150.1.9.0/24     155.1.23.3                             0 100 i
*> 150.1.10.0/24    155.1.23.3                             0 100 i
```

From a bestpath selection point of view, the entire AS_CONFED_SET only counts as one AS. This can sometimes result in confusing path selections, such as R5's route to 150.1.7.0 seen below.

```
Rack1R5#show ip bgp 150.1.7.0
BGP routing table entry for 150.1.7.0/24, version 18
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2           3
  (65379)
    155.1.37.7 (metric 2172416) from 155.1.0.3 (150.1.3.3)
      Origin IGP, metric 0, localpref 100, valid, confed-external
  (65146 65379)
    155.1.37.7 (metric 2172416) from 155.1.0.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, confed-external, best
```

Although R5's path through R3 has a shorter AS path, i.e. only Sub-AS 65379 as opposed to both Sub-ASes 65146 and 65379, both of these confed sets are considered equal. In the case of this selection in particular, the bestpath is chosen as R1 because it has a lower router-id. Intra-Confederation bestpath selection is covered in later tasks, as there are some important exceptions such as this that must be noted.

Copyright © 2010 Internetwork Expert

7.10 BGP Next-Hop Processing - Next-Hop-Self

- Remove the previous BGP configuration on all devices.
- Configure R2 in BGP AS 200, and configure an EBGP peering with BB2.
- BB2 is in AS 254, and is authenticating this peering with the password "CISCO".
- Configure BGP on all other internal devices using AS 100.
- Configure iBGP peerings from R1 to all other devices in AS 100.
- Configure EBGP peerings between R4 & BB3 and R6 & BB1 using their directly connected links; BB1 and BB3 are in AS 54.
- Configure EBGP peerings between R2 & R3 and R2 & R5.
- Advertise the Loopback 0 interfaces of all devices into BGP.
- Remove the advertisement of the links between R4 & BB3 and R6 & BB1 into IGP.
- Use the **`next-hop-self`** command where necessary to ensure full connectivity to the prefixes coming from AS 54.
- Ensure full reachability to the Loopback 0 interfaces from all internal devices, and to all prefixes learned from AS 54 and AS 254 from all internal devices when sourcing traffic from the Loopback 0 interfaces.

Configuration

```
R1:
router bgp 100
 network 150.1.1.0 mask 255.255.255.0
 neighbor 155.1.0.3 remote-as 100
 neighbor 155.1.0.5 remote-as 100
 neighbor 155.1.58.8 remote-as 100
 neighbor 155.1.67.7 remote-as 100
 neighbor 155.1.79.9 remote-as 100
 neighbor 155.1.108.10 remote-as 100
 neighbor 155.1.146.4 remote-as 100
 neighbor 155.1.146.6 remote-as 100
 neighbor 155.1.0.3 route-reflector-client
 neighbor 155.1.0.5 route-reflector-client
 neighbor 155.1.58.8 route-reflector-client
 neighbor 155.1.67.7 route-reflector-client
 neighbor 155.1.79.9 route-reflector-client
 neighbor 155.1.108.10 route-reflector-client
 neighbor 155.1.146.4 route-reflector-client
 neighbor 155.1.146.6 route-reflector-client
```

Copyright © 2010 Internetwork Expert

```
R2:
router bgp 200
 network 150.1.2.0 mask 255.255.255.0
 neighbor 155.1.23.3 remote-as 100
 neighbor 155.1.0.5 remote-as 100
 neighbor 192.10.1.254 remote-as 254
 neighbor 192.10.1.254 password CISCO

R3:
router bgp 100
 network 150.1.3.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100
 neighbor 155.1.23.2 remote-as 200

R4:
router bgp 100
 network 150.1.4.0 mask 255.255.255.0
 neighbor 155.1.146.1 remote-as 100
 neighbor 155.1.146.1 next-hop-self
 neighbor 204.12.1.254 remote-as 54
!
router eigrp 100
 no passive-interface FastEthernet0/0
 no network 204.12.1.0

R5:
router bgp 100
 network 150.1.5.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100

R6:
router bgp 100
 network 150.1.6.0 mask 255.255.255.0
 neighbor 155.1.146.1 remote-as 100
 neighbor 155.1.146.1 next-hop-self
 neighbor 54.1.1.254 remote-as 54
!
router eigrp 100
no passive-interface Serial0/0/0
no network 54.0.0.0

SW1:
router bgp 100
 network 150.1.7.0 mask 255.255.255.0
 neighbor 155.1.146.1 remote-as 100

SW2:
router bgp 100
 network 150.1.8.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100

SW3:
router bgp 100
 network 150.1.9.0 mask 255.255.255.0
 neighbor 155.1.146.1 remote-as 100
```

Copyright © 2010 Internetwork Expert

```
SW4:
router bgp 100
 network 150.1.10.0 mask 255.255.255.0
 neighbor 155.1.0.1 remote-as 100
```

Verification

Note

Recall from the IP Routing section how the route recursion process works. When the longest match route is found for the destination in question, the next-hop value of the prefix is checked. If the longest match to the next-hop value is a connected route, the outgoing interface is known, the layer 2 address is found – depending on the media – and the frame is built for transmission. If the next-hop value is not via a connected interface, additional routing lookups must be performed (i.e. "recursive" lookups) until an outgoing interface is found.

With IGP routing this process is usually transparent, because in the vast majority of cases routes are always learned from directly connected neighbors. For example, if an OSPF route is learned from neighbor A via interface X, it is safe to assume that interface X will be used to reach that destination. However, with BGP, a disconnect can occur between the neighbor prefixes are learned from (the control plane), and the actual path packets take towards the prefix (the forwarding/data plane). The main reason for this is that in many cases, BGP neighbors are not directly connected, but instead exchange BGP control plane information over additional hops in the network. This process can occur because IGP information provides reachability to establish the TCP transport inherent to the BGP control plane.

In order to ensure that this disconnect does not adversely affect the actual forwarding of traffic, the BGP process internally performs the route recursion process for all prefixes prior to performing Bestpath selection. If route recursion is not successful, e.g. the final outgoing interface cannot be found, the prefix cannot be considered for Bestpath selection. This implies that the prefix cannot be installed in the IP routing table, nor can it be advertised to any other BGP peers. In this particular example, this problem can be illustrated in AS 100 by the routes that are learned from AS 54.

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 34
Paths: (2 available, best #1, table Default-IP-Routing-Table)
Flag: 0x860
  Advertised to update-groups:
     1
  54 50 60, (Received from a RR-client)
    155.1.146.6 from 155.1.146.6 (150.1.6.6)
      Origin IGP, metric 0, localpref 100, valid, internal, best
  54 50 60, (Received from a RR-client)
    204.12.1.254 (inaccessible) from 155.1.146.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, internal
```

R4 and R6 both learn the prefix 112.0.0.0/8 from their EBGP peers in AS 54, and pass the route on to R1. Since the route is being advertised to an iBGP neighbor, the next-hop value is not normally updated. Recall that the next-hop value is only updated by default when prefixes are advertised to *true* EBGP peers, not iBGP peers or confederation EBGP peers. However, since the next-hop value is just another attribute of the prefix, like AS-Path or Community, it can be arbitrarily changed as the prefix is advertised or received.

In the above output we can see that on R1 the prefix learned from R6 has a next-hop value of 155.1.146.6, while the prefix from R4 has a next-hop value of 204.12.1.254. Normally R6 would be reporting the next-hop value of 54.1.1.254 to R1, which is the next-hop it learned from BB1, but the **`neighbor 155.1.146.1 next-hop-self`** command has been applied under R6's BGP process. This means that when a prefix is learned from an EBGP neighbor, and then advertised to the neighbor 155.1.146.1, the next-hop value is set to whatever local address is used for the peering towards 155.1.146.4. Since R4 does not have this option applied, the default next-hop that BB3 reports (204.12.1.254) is retained as the next-hop attribute.

The final result of this is that R1 cannot use the route via R4 for Bestpath selection, because the next-hop value is listed as "inaccessible". Inaccessible simply means that R1 does not have a route to the next-hop, which implies that successful route recursion cannot occur. This is further reinforced by the below output.

```
Rack1R1#show ip route 204.12.1.254
% Network not in table
```

Copyright © 2010 Internetwork Expert

Essentially there are only two solutions for the problem presented. R1 must either learn a route to the next-hop 204.12.1.254 via either static or dynamic routing, or the next-hop attribute needs to be changed to something R1 already has a route to. By using the **`next-hop-self`** option on R4 in addition to R6, the latter solution is obtained.

```
Rack1R1#show ip bgp
BGP table version is 24, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i28.119.16.0/24   155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
* i28.119.17.0/24   155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
* i112.0.0.0        155.1.146.6              0    100      0 54 50 60 i
*>i                 155.1.146.4              0    100      0 54 50 60 i
* i113.0.0.0        155.1.146.6              0    100      0 54 50 60 i
*>i                 155.1.146.4              0    100      0 54 50 60 i
* i114.0.0.0        155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
* i115.0.0.0        155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
* i116.0.0.0        155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
* i117.0.0.0        155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
* i118.0.0.0        155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
* i119.0.0.0        155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
*> 150.1.1.0/24     0.0.0.0                  0         32768 i
*>i150.1.2.0/24     155.1.23.2               0    100      0 200 i
*>i150.1.3.0/24     155.1.0.3                0    100      0 i
*>i150.1.4.0/24     155.1.146.4              0    100      0 i
*>i150.1.5.0/24     155.1.0.5                0    100      0 i
*>i150.1.6.0/24     155.1.146.6              0    100      0 i
*>i150.1.7.0/24     155.1.67.7               0    100      0 i
*>i150.1.8.0/24     155.1.58.8               0    100      0 i
*>i150.1.9.0/24     155.1.79.9               0    100      0 i
*>i150.1.10.0/24    155.1.108.10             0    100      0 i
*>i205.90.31.0      155.1.23.2               0    100      0 200 254 ?
*>i220.20.3.0       155.1.23.2               0    100      0 200 254 ?
*>i222.22.2.0       155.1.23.2               0    100      0 200 254 ?
```

Copyright © 2010 Internetwork Expert

Once R4 updates the next-hop value towards R1, R1 can use both prefixes via R4 and R6 for Bestpath selection. Per the below output we can see that the prefix via R4 wins, due to the lower router-id of the originator (150.1.6.6 vs. 150.1.4.4)

```
Rack1R1#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 5
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
     1
  54 50 60, (Received from a RR-client)
    155.1.146.6 from 155.1.146.6 (150.1.6.6)
      Origin IGP, metric 0, localpref 100, valid, internal
  54 50 60, (Received from a RR-client)
    155.1.146.4 from 155.1.146.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, internal, best
```

Copyright © 2010 Internetwork Expert

7.11 BGP Next-Hop Processing - Manual Modification

- Remove the previously configured next-hop-self statements.
- Configure an outbound route-map on R4, and an inbound route-map on R1 in order to resolve any next-hop reachability issues for the routes learned from AS 54.

Configuration

```
R1:
router bgp 100
 neighbor 155.1.146.6 route-map SET_NEXT_HOP_FROM_R6 in
!
route-map SET_NEXT_HOP_FROM_R6 permit 10
 set ip next-hop 155.1.146.6

R4:
router bgp 100
 no neighbor 155.1.146.1 next-hop-self
 neighbor 155.1.146.1 route-map SET_NEXT_HOP_TO_R1 out
!
route-map SET_NEXT_HOP_TO_R1 permit 10
 set ip next-hop 155.1.146.4

R6:
router bgp 100
 no neighbor 155.1.146.1 next-hop-self
```

Verification

Note

Just as in the previous task, R4 and R6 learn prefixes from AS 54, and advertise them to their iBGP neighbor, R1. Normally the next-hop value is not updated in this type of advertisement, which results in R1 seeing R4's prefix via BB3's next-hop, and R6's prefix via BB1's next-hop. Since R1 has a route to neither 54.1.1.254, nor 204.12.1.254, route recursion fails, and the prefixes cannot be considered for Bestpath selection. This can be seen in the following output through the lack of the ">" character in the status code on the left-hand side of the output, which normally denotes the Bestpath.

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip bgp
BGP table version is 44, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i28.119.16.0/24   54.1.1.254               0    100      0 54 i
* i                 204.12.1.254             0    100      0 54 i
* i28.119.17.0/24   54.1.1.254               0    100      0 54 i
* i                 204.12.1.254             0    100      0 54 i
* i112.0.0.0        54.1.1.254               0    100      0 54 50 60 i
* i                 204.12.1.254             0    100      0 54 50 60 i
* i113.0.0.0        54.1.1.254               0    100      0 54 50 60 i
* i                 204.12.1.254             0    100      0 54 50 60 i
<output omitted>
```

Likewise, just as in the previous case, there are two ways to resolve this problem. We either need to give R1 a route towards 54.1.1.254 & 204.12.1.254, e.g. through IGP or static routing, or change the next-hop to something R1 already has a route to. In the previous example the latter solution was implemented using the **next-hop-self** command on R4 and R6 out towards R1. However, since the next-hop value is treated just like any other attribute of the prefix that can be modified, it is also possible to manually change the next-hop value of BGP prefixes arbitrarily by using a route-map.

In the below output R4 applies an outbound route-map towards R1 which sets the next-hop value to 155.1.146.4. Also note the warning message "Next hop address is our address". This warning is meant to be used in policy routing implementations, to make sure that you are not trying to route a packet back to yourself. In our case we are using the route-map for a BGP attribute change, so this warning can be ignored. Lastly, note that the **clear ip bgp** command is used to send a triggered update from R4 to R1 via route refresh. This is necessary any time a manual attribute change is implemented in BGP.

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#route-map SET_NEXT_HOP_TO_R1 permit 10
Rack1R4(config-route-map)#set ip next-hop 155.1.146.4
% Warning: Next hop address is our address
Rack1R4(config-route-map)#router bgp 100
Rack1R4(config-router)#neighbor 155.1.146.1 route-map
SET_NEXT_HOP_TO_R1 out
Rack1R4(config-router)#end
Rack1R4#clear ip bgp 155.1.146.1 out
Rack1R4#
```

Copyright © 2010 Internetwork Expert

Next, on R1, a similar route-map is used to change the next-hop value of routes learned from R6 to 155.1.146.6, followed by a route refresh request from R6 to apply the new policy.

```
Rack1R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#route-map SET_NEXT_HOP_FROM_R6 permit 10
Rack1R1(config-route-map)#set ip next-hop 155.1.146.6
Rack1R1(config-route-map)#router bgp 100
Rack1R1(config-router)#neighbor 155.1.146.6 route-map
SET_NEXT_HOP_FROM_R6 in
Rack1R1(config-router)#end
Rack1R1#clear ip bgp 155.1.146.6 in
```

The final result is that prefixes R6 sends to R1 appear with a next-hop of 155.1.146.6 in the BGP table of R1, while prefix R4 sends to R1 appear with a next-hop value of 155.1.146.4. This effect is essentially the same as using the **next-hop-self** feature, but allows us more control over the BGP policy. In certain traffic engineering designs it may be necessary to change the next-hop value of a prefix to an address completely unrelated to the neighbor that the prefix is learned from.

```
Rack1R1#show ip bgp
BGP table version is 54, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i28.119.16.0/24   155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
* i28.119.17.0/24   155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
* i112.0.0.0        155.1.146.6              0    100      0 54 50 60 i
*>i                 155.1.146.4              0    100      0 54 50 60 i
* i113.0.0.0        155.1.146.6              0    100      0 54 50 60 i
*>i                 155.1.146.4              0    100      0 54 50 60 i
* i114.0.0.0        155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
* i115.0.0.0        155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
* i116.0.0.0        155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
* i117.0.0.0        155.1.146.6              0    100      0 54 i
*>i                 155.1.146.4              0    100      0 54 i
* i118.0.0.0        155.1.146.6              0    100      0 54 i
<output omitted>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.12 iBGP Synchronization

- Enable iBGP Synchronization on all devices in AS 100.
- Ensure full reachability to the Loopback 0 interfaces from all internal devices, and to all prefixes learned from AS 54 and AS 254 from all internal devices when sourcing traffic from the Loopback 0 interfaces.
- Temporarily shut down the peering between R2 and R5.

Configuration

Note

Per Cisco's BGP Bestpath Selection, "if BGP synchronization is enabled, there must be a match for the prefix in the IP routing table in order for an internal BGP (iBGP) path to be considered a valid path." In other words, for every iBGP route learned, there must be a matching IGP route already before the BGP route can be used. This rule was designed to prevent traffic black holes in legacy network designs where not all devices in the BGP transit path actually ran BGP.

With synchronization enabled, the assumption was that all routers in a transit network already ran IGP. If these devices had IGP routes to all BGP destinations, then it was safe to assume that traffic coming from other BGP ASes would not be black holed, even if some of the internal routers were not running BGP. To implement this design it was also assumed that some sort of BGP to IGP redistribution would occur.

The problem with this model however, is that IGP simply cannot scale to the size of the current Internet BGP table. Instead, current best practices dictate that if a network is used for Internet transit, the routing table should be default free (i.e. no 0.0.0.0/0 routes), and all devices should run BGP. Another common design is that other tunneling mechanisms can be used, such as MPLS, to limit the amount of devices that BGP needs to be run on.

BGP synchronization is disabled by default as of IOS 12.2(8)T, and is rarely, if ever, needed in a real implementation anymore. However it is still important to understand the problem that synchronization was designed to prevent, and what the different resolutions for this problem are.

Copyright © 2010 Internetwork Expert

```
R1:
router eigrp 100
 network 150.1.1.1 0.0.0.0
!
router bgp 100
 synchronization

R2:
router eigrp 100
 redistribute bgp 200 metric 100000 1000 255 1 1500 route-map
BGP_TO_IGP
 network 150.1.2.2 0.0.0.0
!
router bgp 200
 synchronization
!
ip as-path access-list 1 permit ^254_
!
route-map BGP_TO_IGP permit 10
 match as-path 1

R3:
router eigrp 100
 network 150.1.3.3 0.0.0.0
!
router bgp 100
 synchronization

R4:
router eigrp 100
 redistribute bgp 100 metric 100000 1000 255 1 1500 route-map
BGP_TO_IGP
 network 150.1.4.4 0.0.0.0
!
router bgp 100
 synchronization
!
ip as-path access-list 1 permit ^54_
!
route-map BGP_TO_IGP permit 10
 match as-path 1
```

Copyright © 2010 Internetwork Expert

```
R5:
router eigrp 100
 network 150.1.5.5 0.0.0.0
!
router bgp 100
 synchronization

R6:
router eigrp 100
 redistribute bgp 100 metric 100000 1000 255 1 1500 route-map
BGP_TO_IGP
 network 150.1.6.6 0.0.0.0
!
router bgp 100
 synchronization
!
ip as-path access-list 1 permit ^54_
!
route-map BGP_TO_IGP permit 10
 match as-path 1

SW1:
router eigrp 100
 network 150.1.7.7 0.0.0.0
!
router bgp 100
 synchronization

SW2:
router eigrp 100
 network 150.1.8.8 0.0.0.0
!
router bgp 100
 synchronization

SW3:
router eigrp 100
 network 150.1.9.9 0.0.0.0
!
router bgp 100
 synchronization

SW4:
router eigrp 100
 network 150.1.10.10 0.0.0.0
!
router bgp 100
 synchronization
```

Copyright © 2010 Internetwork Expert

Verification

Pitfall

Redistribution between IGP and BGP can be dangerous, causing traffic loops, black holes, and network instability due to memory and CPU resource exhaustion. In the previous configuration only necessary routes are redistributed on R2, R4, and R6 based on the AS-Path attribute. Route-map filtering should always be used when performing this type of redistribution to strictly control which prefixes are candidate for redistribution.

Copyright © 2010 Internetwork Expert

In the below output, BGP synchronization is enabled with the **synchronization** command under the BGP process. Note that none of the iBGP learned routes in the table (denoted by the "i" in the status code) are best routes. This is because the synchronization rule has not been met.

```
Rack1R1#show ip bgp
BGP table version is 2, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i28.119.16.0/24   155.1.146.6              0    100      0 54 i
* i                 155.1.146.4              0    100      0 54 i
* i28.119.17.0/24   155.1.146.6              0    100      0 54 i
* i                 155.1.146.4              0    100      0 54 i
* i112.0.0.0        155.1.146.6              0    100      0 54 50 60 i
* i                 155.1.146.4              0    100      0 54 50 60 i
* i113.0.0.0        155.1.146.6              0    100      0 54 50 60 i
* i                 155.1.146.4              0    100      0 54 50 60 i
* i114.0.0.0        155.1.146.6              0    100      0 54 i
* i                 155.1.146.4              0    100      0 54 i
* i115.0.0.0        155.1.146.6              0    100      0 54 i
* i                 155.1.146.4              0    100      0 54 i
* i116.0.0.0        155.1.146.6              0    100      0 54 i
* i                 155.1.146.4              0    100      0 54 i
* i117.0.0.0        155.1.146.6              0    100      0 54 i
* i                 155.1.146.4              0    100      0 54 i
* i118.0.0.0        155.1.146.6              0    100      0 54 i
   Network          Next Hop            Metric LocPrf Weight Path
* i                 155.1.146.4              0    100      0 54 i
* i119.0.0.0        155.1.146.6              0    100      0 54 i
* i                 155.1.146.4              0    100      0 54 i
*> 150.1.1.0/24     0.0.0.0                  0         32768 i
* i150.1.2.0/24     155.1.23.2               0    100      0 200 i
* i150.1.3.0/24     155.1.0.3                0    100      0 i
* i150.1.4.0/24     155.1.146.4              0    100      0 i
* i150.1.5.0/24     155.1.0.5                0    100      0 i
* i150.1.6.0/24     155.1.146.6              0    100      0 i
* i150.1.7.0/24     155.1.67.7               0    100      0 i
* i150.1.8.0/24     155.1.58.8               0    100      0 i
* i150.1.9.0/24     155.1.79.9               0    100      0 i
* i150.1.10.0/24    155.1.108.10             0    100      0 i
* i205.90.31.0      155.1.23.2               0    100      0 200 254 ?
* i220.20.3.0       155.1.23.2               0    100      0 200 254 ?
* i222.22.2.0       155.1.23.2               0    100      0 200 254 ?
```

Copyright © 2010 Internetwork Expert

This can be further verified per the *not synchronized* output seen below. This means that synchronization is on, the route is learned from an iBGP neighbor, and there is no matching IGP route already in the routing table.

```
Rack1R1#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 0
Paths: (2 available, no best path)
  Not advertised to any peer
  54 50 60, (Received from a RR-client)
    155.1.146.6 from 155.1.146.6 (150.1.6.6)
      Origin IGP, metric 0, localpref 100, valid, internal, not
synchronized
  54 50 60, (Received from a RR-client)
    155.1.146.4 from 155.1.146.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, internal, not
synchronized
```

Once BGP is redistributed into IGP, the synchronization rule is met, and the routes are installed as best paths, but the "r" in the status code denotes that RIB-failure now occurs.

```
Rack1R1#show ip bgp
BGP table version is 60, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
r i28.119.16.0/24   155.1.146.6              0    100      0 54 i
r>i                 155.1.146.4              0    100      0 54 i
r i28.119.17.0/24   155.1.146.6              0    100      0 54 i
r>i                 155.1.146.4              0    100      0 54 i
r i112.0.0.0        155.1.146.6              0    100      0 54 50 60 i
r>i                 155.1.146.4              0    100      0 54 50 60 i
r i113.0.0.0        155.1.146.6              0    100      0 54 50 60 i
r>i                 155.1.146.4              0    100      0 54 50 60 i
r i114.0.0.0        155.1.146.6              0    100      0 54 i
r>i                 155.1.146.4              0    100      0 54 i
r i115.0.0.0        155.1.146.6              0    100      0 54 i
r>i                 155.1.146.4              0    100      0 54 i
r i116.0.0.0        155.1.146.6              0    100      0 54 i
r>i                 155.1.146.4              0    100      0 54 i
r i117.0.0.0        155.1.146.6              0    100      0 54 i
r>i                 155.1.146.4              0    100      0 54 i
<output omitted>
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip bgp 222.22.2.0
BGP routing table entry for 222.22.2.0/24, version 37
Paths: (1 available, best #1, table Default-IP-Routing-Table, RIB-
failure(17))
  Advertised to update-groups:
     1
  200 254, (Received from a RR-client)
    155.1.23.2 (metric 3193856) from 155.1.0.3 (150.1.3.3)
      Origin incomplete, metric 0, localpref 100, valid, internal,
synchronized, best
```

Per the above output on R1 222.22.2.0 is now "synchronized", because there is a matching IGP route in the routing table. This means that the route can be used for Bestpath selection, and can be advertised to other BGP neighbors.

The RIB-failure output is an informational message to let us know that although the BGP route is valid, it is not being installed in the routing table. This usually occurs when there is an identical match to a BGP route via an IGP route with a lower administrative distance. In the below output we can see that the external EIGRP route with a distance of 170 prevents the iBGP route from being installed, which would normally have a distance of 200.

```
Rack1R1#show ip route 222.22.2.0
Routing entry for 222.22.2.0/24
  Known via "eigrp 100", distance 170, metric 2937856
  Tag 254, type external
  Redistributing via eigrp 100
  Last update from 155.1.0.5 on Serial0/0.1, 00:05:23 ago
  Routing Descriptor Blocks:
  * 155.1.0.5, from 155.1.0.5, 00:05:23 ago, via Serial0/0.1
      Route metric is 2937856, traffic share count is 1
      Total delay is 50000 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
      Route tag 254
```

RIB-failure by itself is not necessarily bad, but there are certain cases where this disconnect between the BGP table and the routing table can cause traffic loops. By default BGP routes that have RIB-failure *can* be advertised to other neighbors, since the command **`no bgp suppress-inactive`** is the default option under the routing process. To stop RIB-failure routes from being advertised, issue the **`bgp suppress-inactive`** command under the process.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Note

Revert all devices to the *Initial BGP* initial configurations prior to continuing.

7.13 BGP over GRE

- Configure R4 in AS 100, with an EBGP peering to BB3 in AS 54.
- Configure R2 in AS 200, with an EBGP peering to BB2 in AS 254 using the password CISCO.
- Configure an EBGP peering between R2 and R4.
- Advertise the Loopback0 networks of R2 and R4 into BGP.
- Ensure that R2 can reach prefixes learned from AS 54, and R4 can reach prefixes learned from AS 254 when sourcing traffic from their Loopback0 networks.
- Do not redistribute between BGP and IGP to accomplish this.

Configuration

Note

Using automatic tunneling techniques along with BGP is the core of MPLS VPNs. While those are not yet covered within the score of CCIE R&S blueprint, it worth seeing the effect of using simple manual tunnels along with BGP.

Two devices peer BGP (this could be eBGP or iBGP session) across non-BGP capable router cloud. This configuration would mean that any attempt to reach a BGP prefix across the non-BGP cloud would result in prefix black-holing. However, if we establish a direct tunnel between the BGP peers and force all packets go across the tunnel, the non-BGP devices will never ever notice those packets. Thus, the “unknown” addresses will be hidden from the “core” network, only appearing at the edges routers that know about them.

Notice the trick used in the solution. While the “core” IP addresses are used for BGP peering, next-hops in BGP prefixes are modified to point to the tunnel endpoints. Alternatively, you could have peer directly off the tunnel endpoints or even used policy routing to divert packets to the tunnel interfaces.

Copyright © 2010 Internetwork Expert

```
R2:
interface Tunnel0
 ip address 10.0.0.2 255.255.255.0
 tunnel source 155.1.23.2
 tunnel destination 155.1.146.4
!
router bgp 200
 neighbor 155.1.146.4 remote-as 100
 neighbor 155.1.146.4 ebgp-multihop 255
 neighbor 192.10.1.254 remote-as 254
 neighbor 192.10.1.254 password CISCO
 network 150.1.2.0 mask 255.255.255.0

R4:
interface Tunnel0
 ip address 10.0.0.4 255.255.255.0
 tunnel source 155.1.146.4
 tunnel destination 155.1.23.2
!
route-map SET_NEXT_HOP_TO_TUNNEL_OUT permit 10
 set ip next-hop 10.0.0.4
!
route-map SET_NEXT_HOP_TO_TUNNEL_IN permit 10
 set ip next-hop 10.0.0.2
!
router bgp 100
 neighbor 155.1.23.2 remote-as 200
 neighbor 155.1.23.2 ebgp-multihop 255
 neighbor 155.1.23.2 route-map SET_NEXT_HOP_TO_TUNNEL_OUT out
 neighbor 155.1.23.2 route-map SET_NEXT_HOP_TO_TUNNEL_IN in
 neighbor 204.12.1.254 remote-as 54
 network 150.1.4.0 mask 255.255.255.0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Look at the BGP table in R2. Notice that prefixes learned from R4 have their nexhops pointing to the tunnel endpoint.

```
Rack1R2#show ip bgp
BGP table version is 16, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   10.0.0.4                               0 100 54 i
*> 28.119.17.0/24   10.0.0.4                               0 100 54 i
*> 112.0.0.0        10.0.0.4                               0 100 54 50
60 i
*> 113.0.0.0        10.0.0.4                               0 100 54 50
60 i
*> 114.0.0.0        10.0.0.4                               0 100 54 i
*> 115.0.0.0        10.0.0.4                               0 100 54 i
*> 116.0.0.0        10.0.0.4                               0 100 54 i
*> 117.0.0.0        10.0.0.4                               0 100 54 i
*> 118.0.0.0        10.0.0.4                               0 100 54 i
*> 119.0.0.0        10.0.0.4                               0 100 54 i
*> 150.1.2.0/24     0.0.0.0                  0         32768 i
*> 150.1.4.0/24     10.0.0.4                 0             0 100 i
*> 205.90.31.0      192.10.1.254             0             0 254 ?
*> 220.20.3.0       192.10.1.254             0             0 254 ?
*> 222.22.2.0       192.10.1.254             0             0 254 ?
```

Now ping any prefix learned from R4, for example 112.0.0.0/24. Source the packets off R2's Loopback0 interface, as this is the only R2 network known to the external systems:

```
Rack1R2#ping 112.0.0.1 source lo0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 112.0.0.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.2.2
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 100/101/104
ms
```

Copyright © 2010 Internetwork Expert

Next, trace the route to the same prefix off R2's Loopback0 interface. Notice that the path taken by the packets goes across the tunnel interface and any intermediate nodes between R2 and R4 do not show up in the output.

```
Rack1R2#traceroute 112.0.0.1 source loopback 0

Type escape sequence to abort.
Tracing the route to 112.0.0.1

  1 10.0.0.4 56 msec 56 msec 60 msec
  2 204.12.1.254 61 msec 60 msec 60 msec
  3 172.16.4.1 80 msec *  80 msec
```

Now take R4's BGP table and select any prefix learned from R2. Notice that all prefixes received from R2 have their next-hop pointing to 10.0.0.2 (the tunnel endpoint):

```
Rack1R4#show ip bgp
BGP table version is 16, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   204.12.1.254             0             0 54 i
*> 28.119.17.0/24   204.12.1.254             0             0 54 i
*> 112.0.0.0        204.12.1.254                           0 54 50 60 i
*> 113.0.0.0        204.12.1.254                           0 54 50 60 i
*> 114.0.0.0        204.12.1.254                           0 54 i
*> 115.0.0.0        204.12.1.254                           0 54 i
*> 116.0.0.0        204.12.1.254                           0 54 i
*> 117.0.0.0        204.12.1.254                           0 54 i
*> 118.0.0.0        204.12.1.254                           0 54 i
*> 119.0.0.0        204.12.1.254                           0 54 i
*> 150.1.2.0/24     10.0.0.2                 0             0 200 i
*> 150.1.4.0/24     0.0.0.0                  0         32768 i
*> 205.90.31.0      10.0.0.2                               0 200 254 ?
*> 220.20.3.0       10.0.0.2                               0 200 254 ?
*> 222.22.2.0       10.0.0.2                               0 200 254 ?
```

Copyright © 2010 Internetwork Expert

Repeat the traceroute test once again, now targeting prefixes behind BB3. Notice that the very next hop after R4 is the tunnel endpoint again, and transit nodes are unware of the encapsulated packet's destination IP address.

```
Rack1R4#traceroute 222.22.2.1 source loopback 0

Type escape sequence to abort.
Tracing the route to 222.22.2.1

  1 10.0.0.2 48 msec 48 msec 48 msec
  2 192.10.1.254 48 msec *  48 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.14 BGP Redistribute Internal

- Remove all BGP configuration on all devices.
- Configure R1, R3, R4, and R6 in AS 100.
- R4 and R6 should peer with BB3 and BB1 respectively, who are in AS 54.
- R1 should peer with R3, R4, and R6 as a route reflector.
- Configure R4 and R6 to advertise the network 155.X.0.0/16 to AS 54.
- Configure BGP to IGP redistribution on R3 so that all internal devices have reachability to the prefixes learned from AS 54.

Configuration

Note

As discussed previously, enabling BGP on all transit devices in the AS is the best way to ensure that all routers have full external routing information. This solution is scalable, as it avoids feeding large BGP tables into IGP.

In some situations, you cannot enable BGP on all routers in your network. This may be the case of an enterprise network, where only border routers peer eBGP with the ISP. In such situations, it is common to advertise a default route from the border routers towards the rest of the network.

There could be even more complicated scenarios, like when you migrating your network or gradually enabling BGP on all devices. In situations like these, you may find that iBGP peers are separated by non-BGP cloud or that non-BGP speakers need BGP routes from the devices that leaned them via iBGP (no eBGP!). So what's wrong with redistributing iBGP prefixes into IGP? As you remember, BGP uses AS_PATH attributes to detect routing loops. When exchanging iBGP routes, AS_PATH attributes are not prepended and thus the route loop prevention technique does not work. Because of that, feeding iBGP prefixes into an IGP may result in routing loops, as the "split-horizon" rules for BGP prefixes may be broken. To make this situation even worse, iBGP has the AD value that makes it less preferred than any IGP. Thus, iBGP prefixes redistributed into an IGP may preempt iBGP learned prefixes on other iBGP speakers.

Copyright © 2010 Internetwork Expert

To prevent the above mentioned issues, iBGP learned prefixes are not automatically redistributed into IGP when you issue the statement **redistribute bgp** under any IGP process on the router – only eBGP prefixes are propagated. In order to make iBGP redistribution possible, you need an additional statement configured under the BGP process: **bgp redistribute internal**. Be very careful when enabling this feature, as you may quickly end up with routing loops, and try avoiding multiple points of iBGP to IGP redistribution.

In our task, we have to change R1's EIGRP External AD to avoid the effect of iBGP prefixes being preempted by IGP routes.

```
R1:
router eigrp 100
 distance eigrp 90 201
!
router bgp 100
 neighbor 155.1.146.4 remote-as 100
 neighbor 155.1.146.6 remote-as 100
 neighbor 155.1.13.3 remote-as 100
 neighbor 155.1.146.4 route-reflector-client
 neighbor 155.1.146.6 route-reflector-client
 neighbor 155.1.13.3 route-reflector-client

R2:
no router bgp 200

R3:
router eigrp 100
 redistribute bgp 100 metric 100000 1000 255 1 1500
!
router bgp 100
 neighbor 155.1.13.1 remote-as 100
 neighbor 155.1.13.1 route-map SET_NEXT_HOP_FROM_R1 in
 bgp redistribute-internal
!
route-map SET_NEXT_HOP_FROM_R1 permit 10
 set ip next-hop 155.1.13.1

R4:
no router bgp 100
router bgp 100
 neighbor 155.1.146.1 remote-as 100
 neighbor 155.1.146.1 next-hop-self
 neighbor 204.12.1.254 remote-as 54
 network 155.1.146.0 mask 255.255.255.0
 aggregate-address 155.1.0.0 255.255.0.0
```

Copyright © 2010 Internetwork Expert

```
R6:
router bgp 100
 neighbor 155.1.146.1 remote-as 100
 neighbor 155.1.146.1 next-hop-self
 neighbor 54.1.1.254 remote-as 54
 network 155.1.146.0 mask 255.255.255.0
 aggregate-address 155.1.0.0 255.255.0.0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> If you remove the AD fixup in R1 and try tracing the route toward 112.0.0.1 from SW4, you will notice that a routing loop occurs between R1 and R3. This is because the EIGRP route in R1 preempted the iBGP learned route. However, R1 is the route-reflector to R3, and thus R3 thinks it should route back to R1 for prefix 112.0.0.1.

```
Rack1SW4#traceroute 112.0.0.1

Type escape sequence to abort.
Tracing the route to 112.0.0.1

  1 155.1.108.8 4 msec 4 msec 4 msec
  2 155.1.58.5 4 msec 0 msec 4 msec
  3 155.1.0.3 28 msec 32 msec 28 msec
  4 155.1.13.1 36 msec 32 msec 36 msec
  5 155.1.13.3 40 msec 44 msec 40 msec
  6 155.1.13.1 44 msec 44 msec 44 msec
  7 155.1.13.3 52 msec 52 msec 52 msec
  8 155.1.13.1 56 msec 52 msec 56 msec
  9 155.1.13.3 64 msec 60 msec 64 msec
 10 155.1.13.1 68 msec 64 msec 68 msec
 11 155.1.13.3 72 msec 72 msec 72 msec
 12 155.1.13.1 76 msec 76 msec 76 msec
 13 155.1.13.3 84 msec 84 msec 84 msec
 14 155.1.13.1 88 msec 88 msec 88 msec
 15 155.1.13.3 92 msec 96 msec 92 msec
 16 155.1.13.1 96 msec 100 msec 96 msec
 17 155.1.13.3 104 msec 104 msec 104 msec
 18 155.1.13.1 104 msec 112 msec 108 msec
 19 155.1.13.3 116 msec 116 msec 116 msec
 20 155.1.13.1 152 msec 120 msec 120 msec
 21 155.1.13.3 128 msec 128 msec 128 msec
 22 155.1.13.1 128 msec 132 msec 128 msec
 23 155.1.13.3 136 msec 136 msec 136 msec
 24 155.1.13.1 140 msec 140 msec 140 msec
 25 155.1.13.3 148 msec 144 msec 144 msec
 26 155.1.13.1 152 msec 156 msec 152 msec
 27 155.1.13.3 156 msec 156 msec 160 msec
 28 155.1.13.1 160 msec 160 msec 160 msec
 29 155.1.13.3 176 msec 168 msec 164 msec
 30 155.1.13.1 172 msec 172 msec 172 msec
```

Copyright © 2010 Internetwork Expert

Now check the BGP and the routing table of R1. Notice that the best route is via EIGRP toward R3, while BGP prefixes learned point to R4 and R6. But since the BGP prefixes were learned via iBGP, they are preempted by the IGP route in the RIB.

```
Rack1R1#show ip route 112.0.0.0
Routing entry for 112.0.0.0/8
  Known via "eigrp 100", distance 170, metric 2425856
  Tag 54, type external
  Redistributing via eigrp 100
  Last update from 155.1.13.3 on Serial0/1, 00:00:45 ago
  Routing Descriptor Blocks:
  * 155.1.13.3, from 155.1.13.3, 00:00:45 ago, via Serial0/1
      Route metric is 2425856, traffic share count is 1
      Total delay is 30000 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
      Route tag 54

Rack1R1#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 29
Paths: (2 available, best #1, table Default-IP-Routing-Table, RIB-
failure(17))
  Advertised to update-groups:
     1
  54 50 60, (Received from a RR-client)
    155.1.146.4 from 155.1.146.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, internal, best
  54 50 60, (Received from a RR-client)
    155.1.146.6 from 155.1.146.6 (150.1.6.6)
      Origin IGP, metric 0, localpref 100, valid, internal
```

Now apply the Administrative Distance fix in R1 and see how that affects the routing table. Now R1 selects the BGP route as the best one and inserts it into the routing table. At the same time, tracing the route from SW4 reveals that the routing loop has gone and packets can reach the final destination.

```
Rack1R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#router eigrp 100
Rack1R1(config-router)#distance eigrp 90 201
Rack1R1(config-router)#end
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip route 112.0.0.0
Routing entry for 112.0.0.0/8
  Known via "bgp 100", distance 200, metric 0
  Tag 54, type internal
  Last update from 155.1.146.4 00:00:10 ago
  Routing Descriptor Blocks:
  * 155.1.146.4, from 155.1.146.4, 00:00:10 ago
      Route metric is 0, traffic share count is 1
      AS Hops 3
      Route tag 54
```

```
Rack1SW4#traceroute 112.0.0.1

Type escape sequence to abort.
Tracing the route to 112.0.0.1

  1 155.1.108.8 4 msec 0 msec 4 msec
  2 155.1.58.5 0 msec 4 msec 0 msec
  3 155.1.0.3 32 msec 32 msec 32 msec
  4 155.1.13.1 36 msec 36 msec 32 msec
  5 155.1.146.4 24 msec 28 msec 24 msec
  6 204.12.1.254 40 msec 36 msec 36 msec
  7 172.16.4.1 52 msec *  52 msec
```

Copyright © 2010 Internetwork Expert

7.15 BGP Peer Groups

- Remove all BGP the configuration on the devices in AS 100.
- Configure BGP on all internal devices, with the exception of R2 and all switches, using AS 100.
- Configure iBGP peerings from R1 to all other devices in AS 100 using the peer group named IBGP_PEERS, be sure to use the Loopback 0 of these devices to form the peerings.
- Configure EBGP peerings between R4 & BB3 and R6 & BB1 using their directly connected links; BB1 and BB3 are in AS 54.
- Advertise the Loopback 0 interfaces of all devices into both IGP and BGP.
- Ensure full reachability to the Loopback0 interfaces from all internal devices, and to all prefixes learned from AS 54 and AS 254 from all internal devices when sourcing traffic from the Loopback0 interfaces.

Configuration

Note

Large-scale BGP deployments often implement hundreds of BGP peers for a single router. The high amount of peers poses certain scalability issues, to name a few:

1) Configuration burden becomes to intense to deal with and is error-prone.
2) Numerous BGP peers require the router CPU to prepare, batch and replicate BGP updates (often hundred of thousands prefixes) individually for each peer. This poses great stress on router's CPUs.

BGP Peer groups try to overcome the above two issues by utilizing the fact that often many peers have similar BGP policy configuration. For example they all have the same remote AS, outgoing route filters, route-reflection-properties and so on. Based on this fact, many BGP peers sharing the same outgoing policy could be aggregated into a single BGP peer group. The peer group is essentially a template that specifies a particular BGP policy. You create a peer group using the command **neighbor <PEER_GROUP_TAG> peer-group** and then applying all BGP settings to the named tag like if it was a regular BGP peer. When you're done with the peer-group configuration, you assign BGP peers to the group using the command **neighbor <IP_Address> peer-group <PEER_GROUP_TAG>**.

Copyright © 2010 Internetwork Expert

The router's CPU usage is highly optimized when using the peer-groups. All members of the peer group share the same policy, thus allowing a batch of BGP prefixes to be prepared just once, before relayed to all peer-group member. However, this optimization process means that you cannot tune outgoing BGP settings for every member of a peer group individually. For example, you cannot apply a different outgoing route-map to the neighbor that is a member of a BGP peer group. However, you are free to change any incoming policy settings, such as inbound filters.

```
R1:
router bgp 100
 neighbor IBGP_PEERS peer-group
 neighbor IBGP_PEERS remote-as 100
 neighbor IBGP_PEERS update-source Loopback0
 neighbor IBGP_PEERS route-reflector-client
 neighbor 150.1.4.4 peer-group IBGP_PEERS
 neighbor 150.1.3.3 peer-group IBGP_PEERS
 neighbor 150.1.5.5 peer-group IBGP_PEERS
 neighbor 150.1.6.6 peer-group IBGP_PEERS
 network 150.1.1.0 mask 255.255.255.0
!
router eigrp 100
 network 150.1.1.1 0.0.0.0

R3:
router bgp 100
 neighbor 150.1.1.1 remote-as 100
 neighbor 150.1.1.1 update-source Loopback0
 network 150.1.3.0 mask 255.255.255.0
!
router eigrp 100
 network 150.1.3.3 0.0.0.0

R4:
router bgp 100
 neighbor 150.1.1.1 remote-as 100
 neighbor 150.1.1.1 update-source Loopback0
 neighbor 150.1.1.1 next-hop-self
 neighbor 204.12.1.254 remote-as 54
 network 150.1.4.0 mask 255.255.255.0
!
router eigrp 100
 network 150.1.4.4 0.0.0.0

R5:
router bgp 100
 neighbor 150.1.1.1 remote-as 100
 neighbor 150.1.1.1 update-source Loopback0
 network 150.1.5.0 mask 255.255.255.0
!
router eigrp 100
 network 150.1.5.5 0.0.0.0
```

Copyright © 2010 Internetwork Expert

```
R6:
router bgp 100
 neighbor 150.1.1.1 remote-as 100
 neighbor 150.1.1.1 update-source Loopback0
 Neighbor 150.1.1.1 next-hop-self
 neighbor 54.1.1.254 remote-as 54
 network 150.1.6.0 mask 255.255.255.0
!
router eigrp 100
 network 150.1.6.6 0.0.0.0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> First, check the peer-group configured in R1. You can see the peer-group parameters using the command **`show ip bgp peer-group`**. This command shows the group members in addition to displaying the group parameters.

```
Rack1R1#show ip bgp peer-group
BGP peer-group is IBGP_PEERS,  remote AS 100
  BGP version 4
  Default minimum time between advertisement runs is 0 seconds

 For address family: IPv4 Unicast
  BGP neighbor is IBGP_PEERS, peer-group internal, members:
  150.1.3.3 150.1.4.4 150.1.5.5 150.1.6.6
  Index 0, Offset 0, Mask 0x0
  Route-Reflector Client
  Update messages formatted 0, replicated 0
  Number of NLRIs in the update sent: max 0, min 0
```

> **Note**
>
> Now check the BGP table of R1. Make sure you received prefixes from both upstream peers

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip bgp
BGP table version is 21, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i28.119.16.0/24   150.1.6.6                0    100      0 54 i
*>i                 150.1.4.4                0    100      0 54 i
* i28.119.17.0/24   150.1.6.6                0    100      0 54 i
*>i                 150.1.4.4                0    100      0 54 i
* i112.0.0.0        150.1.6.6                0    100      0 54 50 60 i
*>i                 150.1.4.4                0    100      0 54 50 60 i
* i113.0.0.0        150.1.6.6                0    100      0 54 50 60 i
*>i                 150.1.4.4                0    100      0 54 50 60 i
* i114.0.0.0        150.1.6.6                0    100      0 54 i
*>i                 150.1.4.4                0    100      0 54 i
* i115.0.0.0        150.1.6.6                0    100      0 54 i
*>i                 150.1.4.4                0    100      0 54 i
* i116.0.0.0        150.1.6.6                0    100      0 54 i
*>i                 150.1.4.4                0    100      0 54 i
* i117.0.0.0        150.1.6.6                0    100      0 54 i
*>i                 150.1.4.4                0    100      0 54 i
* i118.0.0.0        150.1.6.6                0    100      0 54 i
   Network          Next Hop            Metric LocPrf Weight Path
*>i                 150.1.4.4                0    100      0 54 i
* i119.0.0.0        150.1.6.6                0    100      0 54 i
*>i                 150.1.4.4                0    100      0 54 i
*> 150.1.1.0/24     0.0.0.0                  0         32768 i
*>i150.1.2.0/24     155.1.0.2                0    100      0 200 i
* i                 155.1.23.2               0    100      0 200 i
r>i150.1.3.0/24     150.1.3.3                0    100      0 i
r>i150.1.4.0/24     150.1.4.4                0    100      0 i
r>i150.1.5.0/24     150.1.5.5                0    100      0 i
r>i150.1.6.0/24     150.1.6.6                0    100      0 i
```

Note

Now you can test the connectivity, by ping the external prefixes off all routers' Loopback0 interfaces:

```
Rack1R1#ping 112.0.0.1 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 112.0.0.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/34/36 ms

<The rest of verification omitted>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.16 BGP Network Statement

- Remove the **next-hop-self** statement used in the previous task to peer with BB1 and BB3
- Ensure full reachability from your internal network to all routes learned from AS 54 but do not advertise any prefixes into IGP

Configuration

Note

Unlike the network statement used in IGP protocols configuration, the BGP version of the command is different. The basic command syntax is simple: **network <subnet> mask <netmask>**. It doest *not* define a group of interfaces to enable the protocol, it only specifies the prefix in the IGP table (RIB) to be imported into BGP LocRIB. The LocRIB term stands for Local RIB and is another name for BGP table, which is separate from the routing table (RIB). In order for prefix to be imported, it must exactly match the specification – i.e. it should have the same subnet number and network mask. For example, if you have interface Loopback0 with the IP address 150.1.1.1/24, then the command would be **network 150.1.1.0 mask 255.255.255.0**, not **network 150.1.1.1 mask 255.255.255.0**. The second statement will not match any route in the IGP and thus won't import any prefix. Notice that you may omit the mask specification if it matches the subnet class (e.g. 255.255.255.0 for class C).

When originating prefixes into BGP, it is common to use summarization, in order to minimize the amount of information advertised. One of the ways to achieve this is by creating a special static route that points to Null0 interface and yet encompasses all subnets of the particular AS. The static route is then advertised into BGP table using the network command. For example:

```
ip route 150.0.0.0 255.252.0.0 Null0
!
router bgp 100
 network 150.0.0.0 mask 255.252.0.0
```

One of the special uses for the network command is demonstrated in this task. When peering with another AS, the common question is how to deal with the external next-hop. Two known ways are using the **next-hop-self** parameter when peering via iBGP or advertising the external link subnet into IGP. The last option is advertising the link subnet into BGP and thus propagating it to all iBGP peers. All BGP routers will install it into their RIBs, and perform a recursive lookup to find the actual next hop for every BGP prefix learned from the external AS.

Copyright © 2010 Internetwork Expert

```
R4:
router bgp 100
 no neighbor 150.1.1.1 next-hop-self
 network 204.12.1.0 mask 255.255.255.0

R6:
router bgp 100
 no neighbor 150.1.1.1 next-hop-self
 network 54.1.1.0 mask 255.255.255.0
```

Verification

> **Note**
>
> Look at the BGP table of R1 and notice that both link subnets are there.

```
Rack1R1#show ip bgp
BGP table version is 43, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i28.119.16.0/24   54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i28.119.17.0/24   54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
*>i54.1.1.0/24      150.1.6.6                0    100      0 i
* i112.0.0.0        54.1.1.254               0    100      0 54 50 60 i
*>i                 204.12.1.254             0    100      0 54 50 60 i
* i113.0.0.0        54.1.1.254               0    100      0 54 50 60 i
*>i                 204.12.1.254             0    100      0 54 50 60 i
* i114.0.0.0        54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i115.0.0.0        54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i116.0.0.0        54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i117.0.0.0        54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i118.0.0.0        54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i119.0.0.0        54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
*> 150.1.1.0/24     0.0.0.0                  0         32768 i
*>i150.1.2.0/24     155.1.0.2                0    100      0 200 i
* i                 155.1.23.2               0    100      0 200 i
r>i150.1.3.0/24     150.1.3.3                0    100      0 i
r>i150.1.4.0/24     150.1.4.4                0    100      0 i
r>i150.1.5.0/24     150.1.5.5                0    100      0 i
r>i150.1.6.0/24     150.1.6.6                0    100      0 i
*>i204.12.1.0       150.1.4.4                0    100      0 i
```

Copyright © 2010 Internetwork Expert

Note

Check the routing table entries for the link subnets, and confirm that they point to R4 and R6 respectively.

```
Rack1R1#show ip route 204.12.1.0
Routing entry for 204.12.1.0/24
  Known via "bgp 100", distance 200, metric 0, type internal
  Last update from 150.1.4.4 00:06:28 ago
  Routing Descriptor Blocks:
  * 150.1.4.4, from 150.1.4.4, 00:06:28 ago
      Route metric is 0, traffic share count is 1
      AS Hops 0

Rack1R1#show ip route 54.1.1.0
Routing entry for 54.1.1.0/24
  Known via "bgp 100", distance 200, metric 0, type internal
  Last update from 150.1.6.6 00:06:21 ago
  Routing Descriptor Blocks:
  * 150.1.6.6, from 150.1.6.6, 00:06:21 ago
      Route metric is 0, traffic share count is 1
      AS Hops 0
```

Note

Now check connectivity by pinging the remote AS prefixes. Repeat the below command on all BGP enabled routers:

```
Rack1R1#ping 112.0.0.1 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 112.0.0.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/36/48 ms
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.17 BGP Auto-Summary

- Configure R4 and R6 to originate *classful* auto-summaries for all of your internally assigned address space.
- BB1 and BB3 should not see any of the subnet advertisements that make up this summary.
- Ensure full reachability from your internal network to all routes learned from AS 54.
- Do not use the **`aggregate-address`** command to accomplish this and use different methods to originate routes at R4 and R6.

Configuration

Note

BGP auto-summarization is the legacy feature that automatically summarizes network prefixes to their classful boundaries when the prefixes are advertised into BGP. The automatic summarization starts working when you enable it using the command auto-summary under BGP process configuration. It only applies in the following two cases:

1) A **`network`** command is configured with a *classful* subnet, e.g. **`network 54.0.0.0`** or **`network 155.1.0.0`** or **`network 192.168.1.0`**. In this case, the classful aggregate is installed into BGP table if there is a prefix in the IGP table that is a *subnet* to the classful network. For example, if you advertise network 150.1.0.0 than it would work if any of the prefixes 150.1.2.0/24 or 150.1.3.0/24 etc is in the IGP table. This is in contrary with the regular exact match requirement imposed by the BGP network statements.

2) Prefixes are advertised into BGP using route *redistribution*. All redistributed networks are subject to auto-summarization, i.e. only the major classful subnets are installed in the BGP table.

Since the feature is legacy, you wont seem much use of it nowadays. However, it may become handy in some tricky CCIE scenario that verifies your knowledge of BGP advertisement methods. This scenario uses both methods of route origination with prefix auto-summarization: classful network statement and route redistribution.

Copyright © 2010 Internetwork Expert

```
R1:
router bgp 100
 no network 150.1.1.0 mask 255.255.255.0

R3:
router bgp 100
 no network 150.1.3.0 mask 255.255.255.0
R4:
router bgp 100
 no network 150.1.4.0 mask 255.255.255.0
 auto-summary
 no network 150.1.0.0
 redistribute connected route-map CONNECTED_TO_BGP
!
route-map CONNECTED_TO_BGP
 match interface Loopback0
 match interface FastEthernet 0/1

R5:
router bgp 100
 no network 150.1.5.0 mask 255.255.255.0

R6:
router bgp 100
 no network 150.1.6.0 mask 255.255.255.0
 auto-summary
 network 150.1.0.0
 network 155.1.0.0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> First, start by checking the BGP tables of R4 and R6 for auto-summarized prefixes. Notice that the **regex ^$** filter is used to select the routes locally advertised in this AS. Notice the origin of "?" which means the prefix has been redistributed into BGP. The prefix 155.1.0.0 even has BGP metric assigned based on the IGP metric (EIGRP metric).

```
Rack1R4#show ip bgp regexp ^$
BGP table version is 42, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i54.1.1.0/24      150.1.6.6                0    100      0 i
*> 150.1.0.0        0.0.0.0                  0         32768 ?
* i                 150.1.6.6                0    100      0 i
*> 155.1.0.0        0.0.0.0            2172416         32768 ?
* i                 150.1.6.6          2172416    100      0 i
*> 204.12.1.0       0.0.0.0                  0         32768 i
```

> **Note**
>
> Now look up those prefixes in the BGP table. Notice that both prefixes appear as if they were NOT summarized in classic BGP sense. That is, prefixes don't have any information about the aggregator or the atomic aggregate attribute. This is due to the fact that summarization was performed on the IGP prefixes, not the BGP networks.

```
Rack1R4#show ip bgp 150.1.0.0
BGP routing table entry for 150.1.0.0/16, version 42
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2
  Local
    0.0.0.0 from 0.0.0.0 (150.1.4.4)
      Origin incomplete, metric 0, localpref 100, weight 32768, valid,
sourced, best
  Local
    150.1.6.6 (metric 156160) from 150.1.1.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal
      Originator: 150.1.6.6, Cluster list: 150.1.1.1
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ip bgp 155.1.0.0
BGP routing table entry for 155.1.0.0/16, version 41
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2
  Local
    0.0.0.0 from 0.0.0.0 (150.1.4.4)
      Origin incomplete, metric 2172416, localpref 100, weight 32768,
valid, sourced, best
  Local
    150.1.6.6 (metric 156160) from 150.1.1.1 (150.1.1.1)
      Origin IGP, metric 2172416, localpref 100, valid, internal
      Originator: 150.1.6.6, Cluster list: 150.1.1.1
```

Note

Now check the BGP table in R6 and confirm that auto-summarized prefixes are now in there. In this router, both prefixes has the origin of "i" which mean IGP.

```
Rack1R6#show ip bgp regexp ^$
BGP table version is 34, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 54.1.1.0/24      0.0.0.0                  0         32768 i
*> 150.1.0.0        0.0.0.0                  0         32768 i
*> 155.1.0.0        0.0.0.0            2172416         32768 i
*>i204.12.1.0       150.1.4.4                0    100      0 i
```

Note

Now confirm that only the classful summaries for the internal subnets are advertised to the external BGP peers:

```
Rack1R6#show ip bgp neighbors 54.1.1.254 advertised-routes
BGP table version is 34, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 54.1.1.0/24      0.0.0.0                  0         32768 i
*> 150.1.0.0        0.0.0.0                  0         32768 i
*> 155.1.0.0        0.0.0.0            2172416         32768 i
*>i204.12.1.0       150.1.4.4                0    100      0 i

Total number of prefixes 4
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ip bgp neighbors 204.12.1.254 advertised-routes
BGP table version is 42, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i54.1.1.0/24      150.1.6.6                0    100      0 i
*> 150.1.0.0        0.0.0.0                  0         32768 ?
*> 155.1.0.0        0.0.0.0            2172416         32768 ?
*> 204.12.1.0       0.0.0.0                  0         32768 i

Total number of prefixes 4
```

> **Note**
>
> Finally, perform the following connectivity check on all IBGP speakers to verify full reachability. You should ping any BGP prefix learned from the external BGP peers.

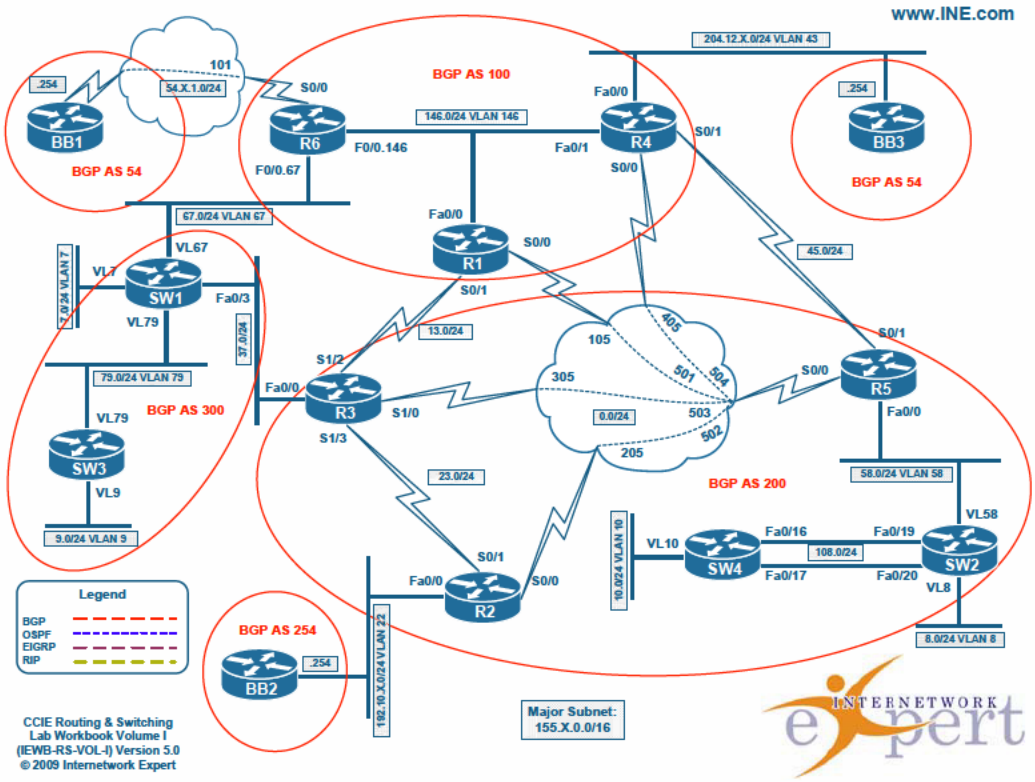
```
Rack1R1#ping 112.0.0.1 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 112.0.0.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/35/37 ms
Rack1R1#

<Rest of the verification omitted>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Note

Load the the *Basic BGP Routing* initial configurations prior to continuing. Use the diagram below as your reference for the tasks that follows.



Copyright © 2010 Internetwork Expert

7.18 BGP Bestpath Selection - Weight

- Using the most influential attribute, configure SW1 so that traffic from AS 300 going to AS 54 prefixes originated in AS 54 exits towards R3, and that traffic from AS 300 going to networks originated in AS 254 networks exits towards R6.

Configuration

Note

BGP Bestpath selection is the core of BGP routing process. This process replaces the shortest-path selection procedure found in traditional IGPs. The reason to use such complicated procedure instead of the shortest path is that BGP prefixes cannot have a classic (additive) metric associated with them. The purpose of the BGP bestpath procedure is to select optimal paths based on *administrative* preferences while maintaining the following properties:

1) Routing Loop detection. The best paths selected should form a loop-free topology. BGP implements this by filtering prefixes with the AS number matching the local AS in the AS_PATH attributes.
2) Deterministic path selection. All BGP routers under the same conditions (e.g. all IBGP speakers configured similarly) must select the same best paths.
2) Routing table stability. The best path selection procedure should not result in constant oscillating route insertion and removals.
3) Information flooding minimization. A BGP speaker only sends the best paths to its neighbors. This significantly reduces the amount of update flooding, saving bandwidth and CPU cycles.

Before we start with the best-path process description, recall that every BGP prefix has a set of attributes associated with it. The procedure uses those attributes when looking for optimal paths. Some of the attributes have more influence on the result than others, as we'll see later. Before the procedure runs, the bestpath process excludes some prefixes based on the following criteria:

1) No valid next hop. This is the most common cause for the prefix being ignored by the selection process. BGP prefixes carry their next-hop is a separate attribute (NEXT_HOP attribute). If the next-hop address is NOT reachable via IGP, the prefix is marked as invalid and is not considered. Most often this happens with eBGP learned prefixes when you forget to enter the command **`nex-hop-self`** or advertise the link subnet into IGP/BGP.
2) BGP Synchronization enabled and the prefix is not in the IGP table. The bestpath process will ignore this prefix. This is a legacy restriction, but you may occasionally run into it, when using BGP synchronization.

Copyright © 2010 Internetwork Expert

3) Prefixes from the neighbor that has the local AS number in the AS_PATH attribute are dropped. This is the well-known BGP loop detection mechanism.

All eligible paths are then sorted and prefixes for the same destination (subnet/mask) are grouped together (the actual implementation may differ though, due to various optimizations). For every group, BGP must elect the best path. Here is a short outline of the steps performed by the selection process. Every step is tried if the previous one cannot reveal the best path:

1) Ignore invalid paths (no valid next hop, not synchronized, looped).
2) Prefer path with the highest locally assigned weight value.
3) Prefer path with the highest Local Preference attribute value.
4) Prefer locally originated prefixes (i.e. originated via the network, aggregate-address or redistribution commands).
5) Prefer path with the shortest AS_PATH attribute length
6) Prefer path with the lowest numerical value of the Origin code (IGP < EGP < Incomplete)
7) Prefer path with the lowest MED attribute value (provided that the first AS in the list is the same).
8) Prefer external BGP paths over Internal
9) Prefer path with the smallest IGP metric to reach the NEXT_HOP IP address
10) Prefer path originated from the router with the lowest BGP Router ID

Ahead of everything else, BGP prefers paths with the highest *weight* attribute. The weight attribute is Cisco-specific, and is not transported along with BGP prefixes. This attribute is configured locally in the router, using the command **neighbor <IP_Address> weight 1-65535**. As mentioned, higher weight values are preferred and the default weight is zero. Thus, among two equal prefixes with different weights the one with the highest weight is preferred. This attribute is commonly used in scenarios where the local router has multiple uplinks and you want to prefer one uplink over another. In addition to the **neighbor** command, weight attribute could be set using inbound route-map associated with the neighbor, for example:

```
route-map SET_WEIGHT
 math ip address ACCCESS_LIST
 set weight 100
!
router bgp 100
 neighbor 204.12.1.254 route-map SET_WEIGHT in
```

This method could be used to change specific prefix preference without affecting any other subnets learned from the same peer. Remember that weight manipulations only affect the way that the traffic leaves the local router.

Copyright © 2010 Internetwork Expert

Notice that IOS routers assign the weight value of 32768 to all locally advertised prefixes – i.e. prefixes advertised using the **network** or **aggregate-address** commands or via route redistribution. This feature ensures that all locally originated prefixes are always preferred again the same subnets learned from the peers.

The solution for this task uses AS-PATH access-lists to match all prefixes from AS 54 and 254. The regular expressions to match all networks originated from AS 54 and 254 respectively are **"54$"** and **"254$"**. SW1 peers with R6 and R3 and we manipulate weight values as follows:

1) For prefixes originated from AS 254 and received from R6 we set the weight value to 1000. This makes SW1 prefer paths to AS254 via R6, as opposed to R3.
2) For prefixes originate from AS 54 and received from R3 we set the weight value to 1000 as well. In result SW1 prefers paths to AS 54 via R3, while it should prefer paths via R6 by default.

Always remember to create a "permit" entry at the end of your route-maps, or you may end up filtering non-matching networks unintentionally.

```
SW1:
no ip as-path access-list 1
no ip as-path access-list 2
ip as-path access-list 1 permit _54$
ip as-path access-list 2 permit _254$
!
route-map FROM_R6 permit 10
 match as-path 2
 set weight 1000
!
route-map FROM_R6 permit 100

!
route-map FROM_R3 permit 10
 match as-path 1
 set weight 1000
!
route-map FROM_R3 permit 100
!
router bgp 300
 neighbor 155.1.67.6 route-map FROM_R6 in
 neighbor 155.1.37.3 route-map FROM_R3 in
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Before you start any verification, use the command **`clear ip bgp * soft`** to force SW1 to refresh the information learned from its peers. By default, when you apply any new policy it does not take any effect unless the new updates are received/sent. Since BGP does not use periodic updates, you may need to force an update manually. The command performs a "soft-reset" - it does not tear down BGP sessions but simply sends out BGP updates and requests route-refresh from the peers. Compared to clear ip bgp * it causes much less load on router's CPU and does not disrupt traffic routing.

```
Rack1SW1#clear ip bgp * soft
```

> **Note**
>
> Now let's look through the BGP table in SW1. Notice the use of regex-based filter _54$ to select only the prefixes originated in AS 54. Look at the prefixes received from R3 – they are selected as "best" and marked with the ">" sign, and the weight assigned is 1000. The "losing" prefixes have the weight value of 0 (the default).

```
Rack1SW1#sh ip bgp regexp _54$
BGP table version is 18, local router ID is 150.1.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  28.119.16.0/24   155.1.67.6                             0 100 54 i
*>                  155.1.37.3                          1000 200 100 54 i
*  28.119.17.0/24   155.1.67.6                             0 100 54 i
*>                  155.1.37.3                          1000 200 100 54 i
*  114.0.0.0        155.1.67.6                             0 100 54 i
*>                  155.1.37.3                          1000 200 100 54 i
*  115.0.0.0        155.1.67.6                             0 100 54 i
*>                  155.1.37.3                          1000 200 100 54 i
*  116.0.0.0        155.1.67.6                             0 100 54 i
*>                  155.1.37.3                          1000 200 100 54 i
*  117.0.0.0        155.1.67.6                             0 100 54 i
*>                  155.1.37.3                          1000 200 100 54 i
*  118.0.0.0        155.1.67.6                             0 100 54 i
*>                  155.1.37.3                          1000 200 100 54 i
*  119.0.0.0        155.1.67.6                             0 100 54 i
*>                  155.1.37.3                          1000 200 100 54 i
```

Copyright © 2010 Internetwork Expert

Note

Repeat the same check with the prefixes originated in AS 254. Notice that now R6 is selected as the upstream peer to route to these subnets.

```
Rack1SW1#sh ip bgp regexp 254$
BGP table version is 18, local router ID is 150.1.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      155.1.67.6                             1000 100 200 254 ?
*                   155.1.37.3                                0 200 254 ?
*> 220.20.3.0       155.1.67.6                             1000 100 200 254 ?
*                   155.1.37.3                                0 200 254 ?
*> 222.22.2.0       155.1.67.6                             1000 100 200 254 ?
*                   155.1.37.3                                0 200 254 ?
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.19 BGP Bestpath Selection - Local Preference

- Remove the BGP configuration for the previous task from SW1.
- Use the local-preference in R6 so that traffic from AS 100 going to AS 254 transits through AS 300.

Configuration

Note

The next attribute that is compared among equal prefixes (assuming the weight for all prefixes is the same) is Local Preference (LP). This is a well-known discretionary attribute that defines how much is the prefix preferred within the scope of a single AS. Numerically higher value of the LP attribute makes BGP prefer the prefix over other with lower LP. Local Preference attribute is carried along with the prefix through the single AS (i.e. relayed across iBGP sessions) but does not leave the AS boundaries. This AS wide propagation ensures that all routers within the single AS perform deterministic path selection.

When you manipulate the local preference, you affect the way that traffic leaves the local AS. Local Preference is typically modified at the bordering BGP speakers, at the point of the particular external connection. To set the local preference for prefixes received from a peer you need to construct a route-map that matches the prefixes you want to manipulate and use the respective set command.

```
route-map SET_LP
 match ip address| match ip as-path ...
 set local-preference 1000
```

The maximum value for local preference is 2^32-1, since the attribute value is 32 bits in size. Keep in mind that by default all iBGP learned prefixes have the Local Preference value of 100 assigned to them. This is needed to give you some space for maneuver when lowering the local preference for some prefixes. If you want to change the default local preference value, use the command **`bgp default local-preference <value>`**. Naturally, this will only affect the default local preference in the router where it's configured.

In this scenario, the default path from AS 100 to AS 254 is across AS 200, based on the AS_PATH length comparison (the step discussed in the following scenario). In order to affect the best-path selection, we configure R6 to assign local preference of 200 to AS 254 prefixes learned from SW1. This will make all routers in AS 100 prefer the exit point via SW1 to reach AS 254 prefixes. Similar to the previous scenarios, we use AS-PATH access-list to match the prefixes originated in AS 254.

Copyright © 2010 Internetwork Expert

```
R6:
no ip as-path access-list 1
ip as-path access-list 1 permit _254$
!
route-map FROM_SW1 permit 10
 match as-path 1
 set local-preference 200
!
route-map FROM_SW1 permit 100
!
router bgp 100
 neighbor 155.1.67.7 route-map FROM_SW1 in

SW1:
router bgp 300
 no neighbor 155.1.67.6 route-map FROM_R6 in
 no neighbor 155.1.37.3 route-map FROM_R3 in
```

Verification

Note

Refresh the routing information and check the BGP tables of the routers in AS 100. Look for paths originated from AS 254, using the regexp "254$".

```
Rack1R6#clear ip bgp * soft
Rack1SW1#clear ip bgp * soft
```

All BGP routers in AS 100 prefer to reach AS 254 prefixes via R6 (look at the next-hop value). Notice that R1 and R4 learned alternative paths to AS 254 via R5 and R3 respectively but they all select the paths with LP 200.

```
Rack1R6#show ip bgp regexp _254$
BGP table version is 18, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      155.1.67.7                        200      0 300 200 254 ?
*> 220.20.3.0       155.1.67.7                        200      0 300 200 254 ?
*> 222.22.2.0       155.1.67.7                        200      0 300 200 254 ?
Rack1R6#

Rack1R1#show ip bgp regexp _254$
BGP table version is 19, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i205.90.31.0      155.1.67.7               0    200      0 300 200 254 ?
*                   155.1.13.3                             0 200 254 ?
*>i220.20.3.0       155.1.67.7               0    200      0 300 200 254 ?
*                   155.1.13.3                             0 200 254 ?
```

Copyright © 2010 Internetwork Expert

```
*>i222.22.2.0       155.1.67.7               0    200      0 300 200 254 ?
*                   155.1.13.3                             0 200 254 ?
Rack1R1#

Rack1R4#show ip bgp regexp _254$
BGP table version is 41, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  205.90.31.0      155.1.45.5                             0 200 254 ?
*>i                 155.1.67.7               0    200      0 300 200 254 ?
*  220.20.3.0       155.1.45.5                             0 200 254 ?
*>i                 155.1.67.7               0    200      0 300 200 254 ?
*  222.22.2.0       155.1.45.5                             0 200 254 ?
*>i                 155.1.67.7               0    200      0 300 200 254 ?
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.20 BGP Bestpath Selection - AS-Path Prepending

- Remove the BGP configuration for the previous task from R6.
- Using AS-Path Prepending, configure AS 200 so that traffic from AS 100 going to AS 254 enters via the link to AS 300.

Configuration

Note

When two prefixes for the same destination have equal weights and local preference values, BGP process will prefer the prefix originated locally – i.e. the prefix advertised via the **`network`** command, **`aggregate-address`** or **`redistribution`**. In IOS routers this step is usually superfluous, as the locally originated prefixes have weight value of 32768, overriding all other steps of BGP bestpath selection process.

The next step in the process is selecting the prefix with the shortest AS_PATH. The length of this attribute is probably the best approximation of the classic IGP metric when mapping this concept to BGP. This could be directly compared to the hop count concept used in RIP. Here is the procedure for computing the AS_PATH length:

1) For every AS_SEQUENCE element, every AS in the list counts as one and the resulting sum is the number of entries in the path.
2) If the prefix was a result of summarization, you may encounter the AS_SET elements in the AS_PATH. The length of AS_SET is assumed to be "1". This is because summarization conceals topology information, and the summarized prefixes may have had different AS_PATH lengths.
3) If the prefix has been originated within the BGP Confederation, then its AS_PATH attribute may contain any of BGP_CONFED_SEQUENCE or BGP_CONFED_SET sub-attributes. The second sub-attribute appears when you summarize prefixes inside confederation. Each of these elements does not count when computing the AS PATH length.

In order to use this step to influence best path selection, you should use the AS_PATH prepending procedure. This procedure applies only to eBGP sessions, i.e. when advertising prefixes to another AS and the local AS number is prepended in front of the AS_PATH attribute the number of times specified. The syntax to perform AS_PATH prending is as follows, assuming that the local AS number is 100:

Copyright © 2010 Internetwork Expert

```
route-map PREPEND
 match ...
 set as-path prepend 100 100 100
!
router bgp 100
 neighbor 54.1.1.254 route-map PREPEND out
```

The above example applies local AS number 3 times, instead of just one, to the prefixes matching the route-map criterion. Notice that even though you apply attribute manipulation in outbound direction, it affects the way that the external systems send traffic to your AS. Manipulating the AS_PATH length is the common way to influence the incoming traffic paths to the local AS and is widely used on the Internet. Usually, the prefixes advertised on least preferred inbound link have the local AS path number prepended 3 or more times. This ensures that any further manipulations will not make those prefixes preferred over the subnets advertise across the "primary" entry point. Keep in mind that the remote AS may change your policy by applying the local preference attribute manipulations. However, that process will only affect path selection within the single AS, not globally on the Internet.

The AS_PATH comparison step could be disable by issuing the command **bgp bestpath as-path ignore**. This command is hidden under BGP configuration CLI, so you have to type it without using the "?" help sign.

```
R6:
router bgp 100
 no neighbor 155.1.67.7 route-map FROM_SW1 in

R3:
ip as-path access-list 1 permit _254$
!
route-map TO_R1 permit 10
 match as-path 1
 set as-path prepend 200 200 200
!
router bgp 200
 neighbor 155.1.13.1 route-map TO_R1 out

R5:
ip as-path access-list 1 permit _254$
!
route-map TO_R1 permit 10
 match as-path 1
 set as-path prepend 200 200 200
!
router bgp 200
 neighbor 155.1.45.4 route-map TO_R1 out
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Clear the BGP session state in R3 and R5 and check the BGP tables in AS 100.

```
Rack1R5#clear ip bgp * soft
Rack1R3#clear ip bgp * soft
```

Notice that R6 has only one set of prefixes toward AS 254 – via AS 300. This is because R4 and R1 received the paths across AS 200 prepended and only advertised the best paths to R6. You can see the prepended paths in the BGP tables of R1 and R4.

```
Rack1R6#show ip bgp regexp _254$
BGP table version is 36, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      155.1.67.7                             0 300 200 254 ?
*> 220.20.3.0       155.1.67.7                             0 300 200 254 ?
*> 222.22.2.0       155.1.67.7                             0 300 200 254 ?
Rack1R6#

Rack1R4#show ip bgp regexp _254$
BGP table version is 53, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i205.90.31.0      155.1.67.7               0    100      0 300 200 254 ?
*                   155.1.45.5                             0 200 200 200 200
254 ?
*>i220.20.3.0       155.1.67.7               0    100      0 300 200 254 ?
*                   155.1.45.5                             0 200 200 200 200
254 ?
*>i222.22.2.0       155.1.67.7               0    100      0 300 200 254 ?
*                   155.1.45.5                             0 200 200 200 200
254 ?
Rack1R4#
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip bgp regexp _254$
BGP table version is 40, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i205.90.31.0      155.1.67.7               0    100      0 300 200 254 ?
*                   155.1.13.3                             0 200 200 200 200
254 ?
*>i220.20.3.0       155.1.67.7               0    100      0 300 200 254 ?
*                   155.1.13.3                             0 200 200 200 200
254 ?
*>i222.22.2.0       155.1.67.7               0    100      0 300 200 254 ?
*                   155.1.13.3                             0 200 200 200 200
254 ?
```

Copyright © 2010 Internetwork Expert

7.21 BGP Bestpath Selection - Origin

- Remove the BGP configuration made for the previous task.
- Using the Origin attribute configure AS 200 so that traffic from AS 100 going to AS 254 enters via the link between R4 and R5.

Configuration

Note

If BGP prefixes have equal weight, local preference and AS_PATH length, then the bestpath selection process compares routes Origin. This attribute is well-known and mandatory and is set by the prefix originator. The allowed values are the following:
1) "IGP", meaning the route was originated using the network or aggregate-address command. It appears as "i" in BGP table output.
2) "EGP", meaning the prefix was received from an EGP peer (legacy). You won't probably see this Origin value in any modern router.
3) "Incomplete" meaning the source could not be determined. This Origin is assigned to the prefixes redistributed into BGP.

This attribute is rarely used to influence the path selection, due to its inflexibility. Nevertheless BGP always account for this step. Origin "IGP" is preferred over "EGP" and the latter is preferred over "Incomplete". This represents the level of "trust" that BGP assigns to various information sources.

In this task we configure the borders routers in AS 200 (R3 and R5) to impose different Origin attribute on the links between R1, R3 and R4, R5. The prefixes advertised to R1 have the Origin value or "Incomplete" while the prefixes advertised to R4 have the Origin value of "IGP".

```
R3:
router bgp 200
 no neighbor 155.1.13.1 route-map TO_R1 out
!
no route-map TO_R1

R5:

router bgp 200
 no neighbor 155.1.45.4 route-map TO_R4 out
!
no route-map TO_R4
```

Copyright © 2010 Internetwork Expert

```
R3:
no ip as-path access-list 1
ip as-path access-list 1 permit _254$
!
route-map TO_R1 permit 10
 match as-path 1
 set origin incomplete
!
route-map TO_R1 permit 100
!
router bgp 200
 neighbor 155.1.13.1 route-map TO_R1 out

R5:
no ip as-path access-list 1
ip as-path access-list 1 permit _254$
!
route-map TO_R4 permit 10
 match as-path 1
 set origin igp
!
route-map TO_R4 permit 100
!
router bgp 200
 neighbor 155.1.45.4 route-map TO_R4 out
```

Verification

> **Note**
>
> Clear BGP sessions and check the BGP tables of R1 and R4. R1 prefers the paths learned from R4, as they have the origin code of "IGP". R4 does not see any AS 254 paths learned from R1, as R1 suppresses their advertisement.

```
Rack1R5#clear ip bgp * soft
Rack1R3#clear ip bgp * soft

Rack1R1#show ip bgp regexp _254$
BGP table version is 18, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i205.90.31.0      155.1.45.5               0    100      0 200 254 i
*                   155.1.13.3                             0 200 254 ?
*>i220.20.3.0       155.1.45.5               0    100      0 200 254 i
*                   155.1.13.3                             0 200 254 ?
*>i222.22.2.0       155.1.45.5               0    100      0 200 254 i
*                   155.1.13.3                             0 200 254 ?
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ip bgp regexp _254$
BGP table version is 20, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      155.1.45.5                             0 200 254 i
*> 220.20.3.0       155.1.45.5                             0 200 254 i
*> 222.22.2.0       155.1.45.5                             0 200 254 i
```

> **Note**
>
> Situation at R6 is a bit more complex. It marks the paths via R4 as being optimal, since the paths received from SW1 has longer AS_PATH length. Thus, this decision does not involve

```
Rack1R6#show ip bgp regexp _254$
BGP table version is 31, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  205.90.31.0      155.1.67.7                             0 300 200 254 ?
*>i                 155.1.45.5               0    100      0 200 254 i
*  220.20.3.0       155.1.67.7                             0 300 200 254 ?
*>i                 155.1.45.5               0    100      0 200 254 i
*  222.22.2.0       155.1.67.7                             0 300 200 254 ?
*>i                 155.1.45.5               0    100      0 200 254 i
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.23 BGP Bestpath Selection - MED

- Remove the BGP configuration made for the previous task.
- Using MED configure AS 100 so that traffic from AS 200 going to AS 54 enters via the link between R4 and R5.

Configuration

Note

If all prefixes have the same Origin, BGP will compare the next less influential attribute – MED or Multi Exit Discriminator. This is attribute is optional non-transitive and its value is an unsigned 32 integer. It is often called “metric” and used on eBGP peering links to select the entry point to the neighboring AS. The AS that originates the prefixes may attach different metric values to them on different exit points from the AS. This hints the neighboring AS to select the best path based on the smallest metric value, since all other attributes usually match (i.e. weight, LP, AS_PATH length).

The MED attribute is non-transitive, and thus the receiving AS will not propagate it across the borders. However, the receiving AS may reset the metric value, if it wants. Cisco BGP implementation automatically assign the value of the MED attribute based on the IGP metric value for the locally originate prefixes. Here is the reasoning behind that.

Most often, BGP prefixes are advertised by the border eBGP speakers. If there are redundant connections between two ASes, multiple BGP speakers may originate the same prefix, attaching the MED attribute copied from the IGP metric of the advertised prefix. The neighboring AS may then apply the bestpath selection for the peer’s prefixes based on the smallest metric value. This makes sense, if the following holds true:

1) The peer (originator) AS uses single IGP across its network, i.e. metric values are numerically comparable.
2) MED values are only compared for the prefixes coming from the same AS.
3) Prefixes were originated from the peer’s AS.

Copyright © 2010 Internetwork Expert

Based on these assumptions, MED attribute simply hints the neighboring AS to select the prefixes with the shortest IGP metric value in the peer's AS. This routing behavior is called "cold potato routing" in contrary to the "hot potato" routing model. The "hot potato" model is effectively in use when MED values are *not* considered during the bestpath selection. When MED values are filtered or are the same among all prefixes, the router will select the prefix with the lowest IGP cost for its next-hop (if the prefixes are of the same type – e.g. both learned via iBGP, then comparing IGP costs is the next step in BGP bespath selection process). This means selecting the path through the closest exit point, based on the IGP metrics of the local AS. This is allegorically analogous to a group of persons trying to get rid of a "hot potato" as fast as possible. The "cold potato" routing model considers the metric hint information from the adjacent AS to make the final routing decision.

Often you may see MED attribute used to influence the exit point selection for prefixes not originated in the local AS. This is perfectly legal, but you need to assign some artificial metric values to these prefixes, as they don't belong to the local AS IGP. Keep in mind that this sort of path attributes manipulation only affects the paths chosen by the directly connected neighboring AS. Lastly, MED attribute is optional and thus may not be present in all prefixes. By default, BGP process will assume the MED value of zero for such prefixes, which will make them more preferred during the selection based on metric. If you want to change this behavior, you need to enter the command **`bgp bestpath med missing-as-worst`**. This will instruct the local router to impose the maximum MED value on the prefixes that does not carry the MED attribute, making these prefixes least preferable. The use of this logic was specified in earlier drafts of the BGP specification, while the most recent instructs to assign the MED value of zero to the prefixes lacking the metric attribute.

In our scenario, we configure AS 100 border routers (R1 and R4) peering with AS 200 to assign metric values for prefixes learned from AS 54. The metrics are assigned so that exit point through R1 is less preferred (higher metric). Notice that we configure both R1 and R4, whereas it is possible to configure just R1, since routes advertised by R4 will have the default MED value of 0.

Copyright © 2010 Internetwork Expert

```
R3:
router bgp 200
 no neighbor 155.1.13.1 route-map TO_R1 out
!
no route-map TO_R1

R5:
router bgp 200
 no neighbor 155.1.45.4 route-map TO_R4 out
!
no route-map TO_R4

R1:
no ip as-path access-list 1
ip as-path access-list 1 permit _54$
!
no route-map TO_R3
route-map TO_R3 permit 10
 match as-path 1
 set metric 1000
!
route-map TO_R3 permit 100
!
router bgp 100
 neighbor 155.1.13.3 route-map TO_R3 out

R4:
no ip as-path access-list 1
ip as-path access-list 1 permit _54$
!
no route-map TO_R5
route-map TO_R5 permit 10
 match as-path 1
 set metric 100
!
route-map TO_R5 permit 100
!
router bgp 100
 neighbor 155.1.45.5 route-map TO_R5 out
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Clear the BGP peering sessions softly and check the BGP tables of AS 200 routers. Notice that R5 only has paths to AS 54 prefixes via R4, as R1 never sends it the prefixes with worse metric. R2 learned all paths with the next-hop toward R4, since both R1 and R5 sent those prefixes to R2. The metric is 100, as set by R4.

```
Rack1R4#clear ip bgp * soft
Rack1R1#clear ip bgp * soft

Rack1R2#show ip bgp regex _54$
BGP table version is 63, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   155.1.45.4             100    100      0 100 54 i
* i                 155.1.45.4             100    100      0 100 54 i
*>i28.119.17.0/24   155.1.45.4             100    100      0 100 54 i
* i                 155.1.45.4             100    100      0 100 54 i
*>i114.0.0.0        155.1.45.4             100    100      0 100 54 i
* i                 155.1.45.4             100    100      0 100 54 i
<snip>

Rack1R5#show ip bgp regexp _54$
BGP table version is 51, local router ID is 150.1.5.5
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   155.1.45.4             100             0 100 54 i
*> 28.119.17.0/24   155.1.45.4             100             0 100 54 i
*> 114.0.0.0        155.1.45.4             100             0 100 54 i
*> 115.0.0.0        155.1.45.4             100             0 100 54 i
*> 116.0.0.0        155.1.45.4             100             0 100 54 i
*> 117.0.0.0        155.1.45.4             100             0 100 54 i
*> 118.0.0.0        155.1.45.4             100             0 100 54 i
*> 119.0.0.0        155.1.45.4             100             0 100 54 i
```

> **Note**
>
> The BGP table of R3 is a bit more cluttered. There are AS 54 prefixes learned from AS 300 (longer AS_PATH, losing) and from both R1 and R5. The prefixes received from R5 won the bestpath process, as they have smaller metric value.

Copyright © 2010 Internetwork Expert

```
Rack1R3#show ip bgp regexp _54$
BGP table version is 31, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   155.1.45.4               100    100      0 100 54 i
*                   155.1.37.7                              0 300 100 54 i
*                   155.1.13.1              1000             0 100 54 i
*>i28.119.17.0/24   155.1.45.4               100    100      0 100 54 i
*                   155.1.37.7                              0 300 100 54 i
*                   155.1.13.1              1000             0 100 54 i
*>i114.0.0.0        155.1.45.4               100    100      0 100 54 i
*                   155.1.37.7                              0 300 100 54 i
*                   155.1.13.1              1000             0 100 54 i
*>i115.0.0.0        155.1.45.4               100    100      0 100 54 i
*                   155.1.37.7                              0 300 100 54 i
*                   155.1.13.1              1000             0 100 54 i
<snip>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.24 BGP Bestpath Selection - Always Compare MED

- Remove the BGP configuration made for the previous task.
- Create a new Loopback 1 interface on both R6 and SW3 with the IP address 1.2.3.4/32 and advertise it into BGP on both R6 and SW3.
- Using just the MED attribute configure the network so that traffic from AS 200 going to this prefix is always received by SW3.

Configuration

Note

As discussed previously, comparing automatically generated MED attributes makes sense only if all prefixes are originated from directly connected AS. However, setting the artificial MED values is often used by adjacent AS to hint the exit point to its peers. Now imagine a situation when the local AS (A) peers with two other ASes (B and C), and the peers have some sort of "backdoor" link between then, e.g. running an IGP on this link. Upon an agreement, the peers may decide to "share" their entry points, so that "A" may send traffic to a subset of "B"'s prefixes across "C" and vice versa. This could be done using the proper manipulation of MED attribute as the prefixes appear to be internal to both ASes, but classic BGP procedure does not compare MEDs for prefixes that were received from different ASes.

This limitation could be disabled by entering the command **`bgp always-compare-med`** under BGP configuration mode. This will instruct BGP code to ignore the first AS# in the AS_PATH attributes when comparing MED values. Of course, this is only possible if the AS_PATH lengths are the same. Using MED for exit point selection to multiple adjacent ASes is getting common, though its against the idea of using MED as a reflection of the internal IGP cost for a prefix. Essentially, this procedure assumes that the adjacent systems have an agreement about MED values assignment, and the local AS accepts their policy.

Copyright © 2010 Internetwork Expert

Through the history of BGP implementation, it has become apparent that using MED in bestpath selection may sometimes result in unstable routing tables or routing loops. There are many different scenarios, where such issues may arise. Particularly, Cisco's BGP implementation suffered from non-deterministic MED-based path selection. What that means is that the result of best-path selection may depend on the order that the local speaker receives BGP prefixes. Specifically, IOS code was ordering BGP prefixes based on their age and compare then in pairs, starting with the newer prefixes. This temporal dependency was intended to make routing tables more stable, by giving more preference for older information. However, when the MED issues become apparent, Cisco implemented a workaround, that removed temporal dependency in MED-based computations. The resulting procedure became deterministic, and no longer depends on the order the prefixes are received. To enable this deterministic mode use the command **bgp bestpath deterministic-med**. The reason this feature is not enabled by default is that many older IOS still use the previous selection procedure and combining those two in the same AS may result in routing loop.

In this task, both AS 300 and AS 100 advertise the same prefix (artificially created). Border routers in AS 300 (SW1) and AS 100 (R1 and R4) are configured to set metrics so that the exit point between R3 and SW1 is used to reach this subnet. In order to make AS 200 account for metrics from different ASes we enable the always-compared-med feature in all router of AS 200.

```
R1:
router bgp 100
 no neighbor 155.1.13.3 route-map TO_R3 out
!
no route-map TO_R3

R4:
router bgp 100
 no neighbor 155.1.45.5 route-map TO_R5 out
!
no route-map TO_R5
```

Copyright © 2010 Internetwork Expert

Note

Here goes the actual configuration.

```
R1:
ip prefix-list LOOPBACK1 permit 1.2.3.4/32
!
route-map TO_R3 permit 10
 match ip address prefix-list LOOPBACK1
 set metric 1000
!
route-map TO_R3 permit 100
!
router bgp 100
 neighbor 155.1.13.3 route-map TO_R3 out

R2:
router bgp 200
 bgp always-compare-med

R3:
router bgp 200
 bgp always-compare-med

R4:
ip prefix-list LOOPBACK1 permit 1.2.3.4/32
!
route-map TO_R5 permit 10
 match ip address prefix-list LOOPBACK1
 set metric 1000
!
route-map TO_R5 permit 100
!
router bgp 100
 neighbor 155.1.45.5 route-map TO_R5 out

R5:
router bgp 200
 bgp always-compare-med

R6:
interface Loopback1
 ip address 1.2.3.4 255.255.255.255
!
router bgp 100
 network 1.2.3.4 mask 255.255.255.255

SW1:
ip prefix-list LOOPBACK1 permit 1.2.3.4/32
!
route-map TO_R3 permit 10
 match ip address prefix-list LOOPBACK1
 set metric 100
```

Copyright © 2010 Internetwork Expert

```
route-map TO_R3 permit 100
!
router bgp 300
 neighbor 155.1.37.3 route-map TO_R3 out

SW2:
router bgp 200
 bgp always-compare-med

SW3:
interface Loopback1
 ip address 1.2.3.4 255.255.255.255
!
router bgp 300
 network 1.2.3.4 mask 255.255.255.255

SW4:
router bgp 200
 bgp always-compare-med
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Clear BGP session on all routers using the command **clear ip bgp * soft**. You may use the **send *** method on the access-server of your rack to accomplish that faster. Now check the BGP table for prefix 1.2.3.4 in R3. The prefix with MED value of 100 is selected as best, even though paths were received from different ASes.

```
Rack1R3#show ip bgp 1.2.3.4
BGP routing table entry for 1.2.3.4/32, version 42
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
     1           2           3
  100
    155.1.13.1 from 155.1.13.1 (150.1.1.1)
      Origin IGP, metric 1000, localpref 100, valid, external
  300
    155.1.37.7 from 155.1.37.7 (150.1.7.7)
      Origin IGP, metric 100, localpref 100, valid, external, best
```

> **Note**
>
> Check the same prefix in R5. The BGP table output here is similar to R3, and again the path through SW1 is selected as the best. Notice the **(metric 2172416)** field on the output. It has nothing to do with the MED value; it's the IGP cost to reach the next-hop for this prefix.

```
Rack1R5#show ip bgp 1.2.3.4
BGP routing table entry for 1.2.3.4/32, version 71
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
     1           3
  100
    155.1.45.4 from 155.1.45.4 (150.1.4.4)
      Origin IGP, metric 1000, localpref 100, valid, external
  300
    155.1.37.7 (metric 2172416) from 155.1.23.3 (150.1.3.3)
      Origin IGP, metric 100, localpref 100, valid, internal, best
```

Copyright © 2010 Internetwork Expert

Note

R2 received two paths from its route reflectors. Since both reflector elected the path via SW1 as the best one, both prefixes in R2's BGP table use SW1 as the exit point out of the AS. R2 selected the second path based on the lowest originating router ID (we will discuss this selection step in a separate task).

```
Rack1R2#show ip bgp 1.2.3.4
BGP routing table entry for 1.2.3.4/32, version 83
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
     1
  300
    155.1.37.7 (metric 2172416) from 155.1.0.5 (150.1.5.5)
      Origin IGP, metric 100, localpref 100, valid, internal
      Originator: 150.1.3.3, Cluster list: 150.1.5.5
  300
    155.1.37.7 (metric 2172416) from 155.1.23.3 (150.1.3.3)
      Origin IGP, metric 100, localpref 100, valid, internal, best
```

Copyright © 2010 Internetwork Expert

7.25 BGP Bestpath Selection - AS-Path Ignore

- Remove the BGP configuration made for the previous task.
- Ensure that traffic from AS 200 to AS 54 prefixes takes path across AS 300.
- Do not use AS-PATH prepending to accomplish this.

Configuration

Note

Ignoring AS_PATH length comparison is optional and could be enabled using a special hidden command. Using this feature in production is highly not recommended as it may severely affect routing table's stability. However, if you need it just for some non-standard tweak of you BGP path selection, use the command **`bgp bestpath as-path ignore`** to activate this feature. BGP will automatically skip AS_PATH length comparison and proceed to comparing the Origin codes, MED attribute and the IGP costs for NEXT_HOPs.

One particular case where you may actually *need* this feature is BGP confederations. As you remember, BGP_CONFED_SEQUENCE and BGP_CONFED_SET do not count when computing the AS_PATH length. Therefore, in BGP confederations you may see suboptimal paths being elected simply based on the AS_PATH attribute carried from the external ASes (internal path lengths are ignored). By disabling the AS_PATH comparison you may force the local speakers to use the "hot patato" routing model, taking in account the IGP cost to reach the prefix NEXT_HOP. That is, only provided if all prefixes have the same Origin and MED attribute values, which could be enforced by border BGP speakers.

In this scenario we modify the Origin attribute for paths injected into AS 200 through the peering connection SW1-R3. All routes in AS 200 has AS_PATH length comparison disabled, and thus prefer the paths to AS 54 learned from SW1 even though those have longer AS_PATHs.

Copyright © 2010 Internetwork Expert

```
R1:
router bgp 100
 no neighbor 155.1.13.3 route-map TO_R3 out
!
no route-map TO_R3

R4:
router bgp 100
 no neighbor 155.1.45.5 route-map TO_R5 out
!
no route-map TO_R5
R6:
no interface Loopback1
!
router bgp 100
 no network 1.2.3.4 mask 255.255.255.255

SW1:
router bgp 300
 no neighbor 155.1.37.3 route-map TO_R3 out
!
no route-map TO_R3

SW3:
no interface Loopback1
!
router bgp 300
 no network 1.2.3.4 mask 255.255.255.255
```

Note

Now the actual configuration for this task

```
R1:
no ip as-path access-list 1
ip as-path access-list 1 permit _54$
!
route-map TO_R3 permit 10
 match as-path 1
 set origin incomplete
!
route-map TO_R3 permit 100
!
router bgp 100
 neighbor 155.1.13.3 route-map TO_R3 out

R2:
router bgp 200
 bgp bestpath as-path ignore

R3:
router bgp 200
 bgp bestpath as-path ignore
```

Copyright © 2010 Internetwork Expert

```
R4:
no ip as-path access-list 1
ip as-path access-list 1 permit _54$
!
route-map TO_R5 permit 10
 match as-path 1
 set origin incomplete
!
route-map TO_R5 permit 100
!
router bgp 100
 neighbor 155.1.45.5 route-map TO_R5 out

R5:
router bgp 200
 bgp bestpath as-path ignore

SW1:
no ip as-path access-list 1
ip as-path access-list 1 permit _54$
!
route-map TO_R3 permit 10
 match as-path 1
 set origin igp
!
route-map TO_R3 permit 100
!
router bgp 300
 neighbor 155.1.37.3 route-map TO_R3 out

SW2:
router bgp 200
 bgp bestpath as-path ignore

SW4:
router bgp 200
 bgp bestpath as-path ignore
```

Verification

```
Rack1SW1#clear ip bgp * soft
Rack1R1#clear ip bgp * soft
Rack1R4#clear ip bgp * soft
```

Note

After clearing the BGP sessions, inspect the BGP tables of R3 and R4. Notice that both route-reflectors select the paths with the longer AS_PATHs, based on their Origin code.

Copyright © 2010 Internetwork Expert

```
Rack1R3#show ip bgp regexp _54$
BGP table version is 23, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   155.1.37.7                             0 300 100 54 i
*                   155.1.13.1                             0 100 54 ?
*> 28.119.17.0/24   155.1.37.7                             0 300 100 54 i
*                   155.1.13.1                             0 100 54 ?
*> 114.0.0.0        155.1.37.7                             0 300 100 54 i
*                   155.1.13.1                             0 100 54 ?
*> 115.0.0.0        155.1.37.7                             0 300 100 54 i
*                   155.1.13.1                             0 100 54 ?
*> 116.0.0.0        155.1.37.7                             0 300 100 54 i
*                   155.1.13.1                             0 100 54 ?
*> 117.0.0.0        155.1.37.7                             0 300 100 54 i
*                   155.1.13.1                             0 100 54 ?
*> 118.0.0.0        155.1.37.7                             0 300 100 54 i
*                   155.1.13.1                             0 100 54 ?
*> 119.0.0.0        155.1.37.7                             0 300 100 54 i
*                   155.1.13.1                             0 100 54 ?
Rack1R3#
```

```
Rack1R5#show ip bgp regexp _54$
BGP table version is 25, local router ID is 150.1.5.5
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  28.119.16.0/24   155.1.45.4                             0 100 54 ?
*>i                 155.1.37.7               0    100      0 300 100 54 i
*  28.119.17.0/24   155.1.45.4                             0 100 54 ?
*>i                 155.1.37.7               0    100      0 300 100 54 i
*  114.0.0.0        155.1.45.4                             0 100 54 ?
*>i                 155.1.37.7               0    100      0 300 100 54 i
*  115.0.0.0        155.1.45.4                             0 100 54 ?
*>i                 155.1.37.7               0    100      0 300 100 54 i
*  116.0.0.0        155.1.45.4                             0 100 54 ?
*>i                 155.1.37.7               0    100      0 300 100 54 i
*  117.0.0.0        155.1.45.4                             0 100 54 ?
*>i                 155.1.37.7               0    100      0 300 100 54 i
*  118.0.0.0        155.1.45.4                             0 100 54 ?
*>i                 155.1.37.7               0    100      0 300 100 54 i
*  119.0.0.0        155.1.45.4                             0 100 54 ?
*>i                 155.1.37.7               0    100      0 300 100 54 i
```

7.26 BGP Bestpath Selection - Router-IDs

- Modify the BGP router-ids in AS 100 as necessary so that traffic from R1 to AS 54 exits via R6.

Configuration

Note

The next few steps after the MED attribute processing include:

a) Selecting external paths over internal, as the information should be more actual. Here external paths are prefixes learned via eBGP sessions on the router, while internal paths are learned via iBGP sessions.
b) Prefer the path with the minimum IGP metric to reach the NEXT_HOP IP address. This is a natural selection step, as it attempts to pick up to closes exit point based on the local AS IGP metrics. Keep in mind that exit point selection based on MED (adjacent AS metrics) is preferred over this step. Routing based on the closes exit point in terms of local IGP metric is called "hot potato" routing. On contrary, routing based on the MED values, i.e. based on the metric values advertised by the adjacent AS is called "cold potato" routing.

If all prefixes have the same IGP cost to reach their NEXT_HOPs, the BGP process may consider inserting all of them into RIB, implementing equal cost multipath load-balancing. This feature is enabled by using the command **maximum-paths [ibgp]** under BGP configuration mode. Specify the **ibgp** keyword if you want to load balance among the paths learned via iBGP.
c) Among the prefixes learned from different eBGP peers, prefer the older one, to minimize route flapping.
d) Use BGP Router IDs of the advertising (peering) routers as tie-breakers for the best-path selection process. The path advertised by the peer with the lowest router ID is preferred.

In this scenario, the default BGP Router IDs for R4 and R6 are based on their Loopback0 IP address value. This makes R1 prefer R4 over R6 as the exit point, as all other criteria are the same. In order to change this, we configure R6 with an artificially lowers Router ID value. Remember that changing a router's BGP router ID will hard reset all active BGP sessions.

```
R6:
router bgp 100
 bgp router-id 6.6.6.6
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the BGP table of R1 before you have changed R6's router ID:

```
Rack1R1#show ip bgp regexp _54$
BGP table version is 15, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i28.119.16.0/24   54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i28.119.17.0/24   54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i114.0.0.0        54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
<snip>
```

> **Note**
>
> The paths learned via R4 are preferred. This is done based on the Router ID of R4, as all other attributes are equal (weight, LP, AP_PATH, Origin, MED, iBGP prefixes) including the IGP cost to reach the next-hops:

```
Rack1R1#show ip route 54.1.1.254
Routing entry for 54.1.1.0/24
  Known via "eigrp 1", distance 170, metric 2560002816, type external
  Redistributing via eigrp 1
  Last update from 155.1.146.6 on FastEthernet0/0, 00:42:50 ago
  Routing Descriptor Blocks:
  * 155.1.146.6, from 155.1.146.6, 00:42:50 ago, via FastEthernet0/0
      Route metric is 2560002816, traffic share count is 1
      Total delay is 110 microseconds, minimum bandwidth is 1 Kbit
      Reliability 1/255, minimum MTU 1 bytes
      Loading 1/255, Hops 1

Rack1R1#show ip route 204.12.1.254
Routing entry for 204.12.1.0/24
  Known via "eigrp 1", distance 170, metric 2560002816, type external
  Redistributing via eigrp 1
  Last update from 155.1.146.4 on FastEthernet0/0, 00:43:01 ago
  Routing Descriptor Blocks:
  * 155.1.146.4, from 155.1.146.4, 00:43:01 ago, via FastEthernet0/0
      Route metric is 2560002816, traffic share count is 1
      Total delay is 110 microseconds, minimum bandwidth is 1 Kbit
      Reliability 1/255, minimum MTU 1 bytes
      Loading 1/255, Hops 1
```

Copyright © 2010 Internetwork Expert

Note

Now look at R1's BGP table once more, after you have changed the router ID in R6. Now all paths are preferred via R6, as it has the lowest Router ID.

```
Rack1R1#show ip bgp regexp _54$
BGP table version is 26, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   54.1.1.254               0    100      0 54 i
* i                 204.12.1.254             0    100      0 54 i
*>i28.119.17.0/24   54.1.1.254               0    100      0 54 i
* i                 204.12.1.254             0    100      0 54 i
*>i114.0.0.0        54.1.1.254               0    100      0 54 i
* i                 204.12.1.254             0    100      0 54 i
*>i115.0.0.0        54.1.1.254               0    100      0 54 i
* i                 204.12.1.254             0    100      0 54 i
*>i116.0.0.0        54.1.1.254               0    100      0 54 i
* i                 204.12.1.254             0    100      0 54 i
*>i117.0.0.0        54.1.1.254               0    100      0 54 i
* i                 204.12.1.254             0    100      0 54 i
*>i118.0.0.0        54.1.1.254               0    100      0 54 i
* i                 204.12.1.254             0    100      0 54 i
*>i119.0.0.0        54.1.1.254               0    100      0 54 i
* i                 204.12.1.254             0    100      0 54 i
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.27 BGP Bestpath Selection - DMZ Link Bandwidth

- Remove the Route-ID configurated in the previous section.
- Modify the configuration of AS 100 routers so that R1 load-balances to the paths in AS 54 proportional to the bandwidth of the links connecting R4 and R6 to AS 54 routers.

Configuration

Note

As mentioned in the previous tasks, local BGP process may implement equal-cost load-balancing to the paths that have:

1) The same set of path attributes up to the MED, i.e. weight, Local Preference, Origin, MED
2) Are of the same type, i.e. both learned via iBGP or eBGP
3) Have the same IGP cost to reach their NEXT_HOP IP address.

If the above conditions are met and **`maximum-paths [ibgp]`** is configured under the BGP process, then BGP will install multiple equal-cost routes into the local RIB and use them for load-balancing. We call the above condition as load-balancing conditions for BGP.

BGP also implements unique unequal-cost load balancing feature. As you remember, unequal-cost load balancing could not be implemented easily with any IGP. The protocol needs a way to ensure that all alternative paths are loop-free. So far only EIGRP support this feature, since all alternate unequal cost paths are guaranteed to be loop free by the virtue of feasible successor property. As for BGP, it ensures loop-free property for any routes learned via eBGP, based on the duplicate AS number detection. Thus, it is possible to implement unequal cost load-balancing in BGP toward the prefixes learned from other ASes.

This feature is called DMZ Link Bandwidth in IOS. The rationale behind this name is that load-balancing is based on the bandwidth of the links connecting the border BGP peers to their neighbors. Here is how it works for a single router with multiple eBGP peering links:

1) You enable the feature in a border BGP router using the command **`bgp dmzlink-bw`**. With this command enabled, the BGP process will instruct the data plane to load-balance based on the bandwidth of the links used to connect to the external BGP peers.In order to select the links that are to be used for load-balancing, you configure the respective BGP peers using the command **`neighbor <IP> dmzlink-bw`**.

Copyright © 2010 Internetwork Expert

The BGP process will take the bandwidth on the links connecting to those peers in consideration when doing the unequal cost load-balancing. Those links are called the DMZ Links in Cisco terminology. The bandwidth is computed based on the **bandwidth** command configured on the respective interfaces.
2) You enable the classic BGP equal-cost load-balancing using the command **maximum-paths** under the local BGP process. Now, assuming that you received the same prefix from multiple peers and all prefixes satisfy the BGP load-balancing conditions defined above, the BGP process will insert them into RIB and assign load-balancing weights proportional to the interface bandwidth values (DMZ link bandwidth).

OK, this seems to be simple enough. Now what if you have multiple BGP border peers in your AS, each having just one uplink? Is it possible to implement an AS-wide load-balancing scheme based on the bandwidth of the upstream links? That is, it would be beneficial if every every router learning multiple paths to the same prefix across iBGP links would load-balance toward them based on the bandwidth of the link where they were received. Cisco IOS allows for such implementation, using the following algorithm:

1) When DMZ Link bandwidth feature is enabled in the border BGP routers for the specific peers, the interface bandwidth value is copied into a new extended community attribute associated with the prefixes received from those eBGP peers. Thus, every prefix received on eBGP peering link will carry the links bandwidth is a special extended community attribute, if the link is enabled for DMZ Link bandwidth feature. Keep in mind that you need two commands in the border peers: **bgp dmzlink-bw** and **neighbor <IP> dmzlink-bw**.
2) All BGP speakers in the AS should be configured to exchange extended communities across the iBGP peering links. This allows all internal BGP speakers to learn the bandwidth of the external link used to reach the prefixes. Use the command **neighbor <IP> send-community extended** to accomplish this.
3) Provided that an internal BGP speaker has both **bgp maximum-path ibgp** and **bgp dmzlink-bw** commands enabled and receives multiple paths to reach the same prefix it performs load-balancing if the paths meet the BGP load-balancing conditions.
4) If all paths received carry the DMZ Link bandwidth extended community, the BGP process will perform unequal cost load-balancing proportional to the extended community attribute values.

Copyright © 2010 Internetwork Expert

In our scenario, we have two border BGP routers – R4 and R6 peering with the same AS 54. Our goal is to make R1 learn about the bandwidth of the links connectiong R4 and R6 to AS 54 and enable load-balancing. We achive this by configuring R4 and R6 to send extended communities to R1 and enabling dmzlink bandwidth feature in R4 and R6. At the same time, R1 is configured for iBGP multipathing, and inserts both both sets of paths into the local RIB.

Copyright © 2010 Internetwork Expert

```
R1:
router bgp 100
 maximum-path ibgp 2
 bgp dmzlink-bw

R4:
router bgp 100
 bgp dmzlink-bw
 neighbor 155.1.146.1 send-community extended
 neighbor 204.12.1.254 dmzlink-bw

R6:
router bgp 100
 bgp dmzlink-bw
 neighbor 155.1.146.1 send-community extended
 neighbor 54.1.1.254 dmzlink-bw
!
interface Serial 0/0/0
 bandwidth 2000
```

Verification

> **Note**
>
> Take any prefix learned from AS 54 in R1 and look it up in the BGP table. Notice that there are two paths, both marked as "multipath". That means BGP is using them both, even though only the second path is elected as "best" by the BGP process. Notice the DMZ-Link Bw attribute values for both paths. Their ratio is 12500/250=50.

```
Rack1R1#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 4
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Multipath: iBGP
  Advertised to update-groups:
     1           2
  54 50 60, (Received from a RR-client)
    54.1.1.254 (metric 2560002816) from 155.1.146.6 (150.1.6.6)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      DMZ-Link Bw 250 kbytes
  54 50 60, (Received from a RR-client)
    204.12.1.254 (metric 2560002816) from 155.1.146.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath,
best
      DMZ-Link Bw 12500 kbytes
```

Copyright © 2010 Internetwork Expert

Note

Now look at the routing table entry for the same prefix. Notice that the share counters are 48:1 which is close to 50:1 but is rounded to match the CEF hashing algorithm. That means that approximately for every 48 packets sent via R4, one packet is routed across R6 (though this proportion may be different with per-flow load balancing).

```
Rack1R1#show ip route 112.0.0.0
Routing entry for 112.0.0.0/8
  Known via "bgp 100", distance 200, metric 0
  Tag 54, type internal
  Last update from 204.12.1.254 00:07:21 ago
  Routing Descriptor Blocks:
    204.12.1.254, from 155.1.146.4, 00:07:21 ago
      Route metric is 0, traffic share count is 48
      AS Hops 3
      Route tag 54
  * 54.1.1.254, from 155.1.146.6, 00:07:21 ago
      Route metric is 0, traffic share count is 1
      AS Hops 3
      Route tag 54
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.28 BGP Bestpath Selection - Maximum AS Limit

- Remove the previous section's configurations.
- Configure the routers in AS 100 to accept only the prefixes originated from directly connected ASes.
- Do not use filtering based on AP-PATH access-lists to accomplish this.

Configuration

Note

BGP implementation in Cisco IOS has one special feature that looks similar to TTL scoping in IP networks. It is called BGP maximum AS limit and enabled using the BGP process command **`bgp maxas-limit <N>`**. This feature sets the maximum number of AS elements allowed in the AS_PATH attribute. The regular counting rules apply, i.e. AS_SET element counts as one, and AS_CONFED_* elements are ignored when counting. The default value for this limit is 75 AS elements, and this may be exceeded if AS_PATH preneding is used extensively.

BGP process will generate log message for prefixes that exceed the configured limit, similar to the message below:

```
%BGP-6-ASPATH: Long AS path 200 254 received from 155.1.146.1: More than
configured MAXAS-LIMIT
```

In our scenario we configure R1, R4 and R6 to accept only the prefixes with only one AS element in the AS_PATH attribute. This limits prefixes accept to just directly attached systems – AS 54, AS 100 and AS 200. For example, due to this filtering, A 100 will no be able to receive prefixes from AS 254.

```
R1, R4 & R6:
router bgp 100
 bgp maxas-limit 1
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Remember to clear the BGP sessions after you have applied this feature, as it applies only to the incoming prefixes. After that, check the BGP tables of all routers in AS 100 and confirm that they only contain prefixes with the AS_PATH length of one or less.

```
Rack1R1#clear ip bgp * soft
Rack1R4#clear ip bgp * soft
Rack1R6#clear ip bgp * soft

Rack1R1#show ip bgp
BGP table version is 20, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i28.119.16.0/24   54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i28.119.17.0/24   54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i114.0.0.0        54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i115.0.0.0        54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i116.0.0.0        54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i117.0.0.0        54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i118.0.0.0        54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i119.0.0.0        54.1.1.254               0    100      0 54 i
*>i                 204.12.1.254             0    100      0 54 i
* i155.1.0.0        155.1.146.6              0    100      0 i
*>i                 155.1.146.4              0    100      0 i
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ip bgp
BGP table version is 32, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   204.12.1.254             0             0 54 i
*> 28.119.17.0/24   204.12.1.254             0             0 54 i
*> 114.0.0.0        204.12.1.254                           0 54 i
*> 115.0.0.0        204.12.1.254                           0 54 i
*> 116.0.0.0        204.12.1.254                           0 54 i
*> 117.0.0.0        204.12.1.254                           0 54 i
*> 118.0.0.0        204.12.1.254                           0 54 i
*> 119.0.0.0        204.12.1.254                           0 54 i
*> 155.1.0.0        0.0.0.0                           32768 i
s> 155.1.146.0/24   0.0.0.0                  0        32768 i

Rack1R6#show ip bgp
BGP table version is 84, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i28.119.16.0/24   204.12.1.254             0    100      0 54 i
*>                  54.1.1.254                            0 54 i
* i28.119.17.0/24   204.12.1.254             0    100      0 54 i
*>                  54.1.1.254                            0 54 i
* i114.0.0.0        204.12.1.254             0    100      0 54 i
*>                  54.1.1.254               0             0 54 i
* i115.0.0.0        204.12.1.254             0    100      0 54 i
*>                  54.1.1.254               0             0 54 i
* i116.0.0.0        204.12.1.254             0    100      0 54 i
*>                  54.1.1.254               0             0 54 i
* i117.0.0.0        204.12.1.254             0    100      0 54 i
*>                  54.1.1.254               0             0 54 i
* i118.0.0.0        204.12.1.254             0    100      0 54 i
*>                  54.1.1.254               0             0 54 i
* i119.0.0.0        204.12.1.254             0    100      0 54 i
*>                  54.1.1.254               0             0 54 i
* i155.1.0.0        155.1.146.4              0    100      0 i
*>                  0.0.0.0                           32768 i
s> 155.1.146.0/24   0.0.0.0                  0        32768 i
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.29 BGP Backdoor

- Remove the BGP configuration applied in the previous task.
- Shutdown the BGP peering link between AS 100 and AS 300
- Create a new Loopback1 interface in SW1 with the IP address 150.1.77.77/24 and advertise it into BGP.
- Configure R1 and R4 so that they prefer reaching the new subnet via EIGRP as opposed to eBGP.

Configuration

Note

BGP prefixes learned from eBGP peers have the AD value of 20 – the lowest among all dynamic routing protocols. This was done intentionally to prevent possible routing loops cause by redistribution of BGP routes into IGP. The local router always trusts the prefixes learned from external peers as they have passed the loop detection test.

In some cases, two autonomous systems may be connected by a "backdoor" link used to exchange routes between the ASes dynamically, by means of an IGP. The may be the result of a split or merger. In any case, the issue is that the backdoor link is usually preferred to exchange traffic between the two ASes. However, if both systems peer with another external AS, the border BGP routers will choose the prefix advetised via eBGP over the same prefix received across the backdoor link via IGP. For example, in our scenario AS 300 and AS 100 run the EIGRP on the "backdoor" link connecting SW1 and R6. When we shutdown the BGP peering session between mentioned routers, the only way they can exchange routing information is by means of IGP. When the new prefix is advertised into BGP from SW1, R1 and R4 will learn it via eBGP and EIGRP simultaneously. Based on the preferred AD for eBGP prefixes, R1 and R4 will choose suboptimal paths across AS 200, instead of the backdoor link.

To resolve this issue you may change the AD of eBGP routes in R1 and R4, but this may increase the risk of routing loops. There is a special command in the BGP configuration mode used to explicitly change the distance of an eBGP prefix: **`network <subnet> mask <netmask> backdoor`**. Keep in mind that the purpose of this command is to change the AD of a particaulr eBGP prefix from 20 to 200, not to advertise a new network. Thus, the command applies to non-local prefixes as well. When the command is entered, the eBGP speakers will prefer paths learned via IGP, and utilize the backdoor link.

Copyright © 2010 Internetwork Expert

```
R1, R4, & R6:
router bgp 100
 no bgp maxas-limit 1

SW1:
router bgp 300
 neighbor 155.1.67.6 shutdown
 network 150.1.77.0 mask 255.255.255.0
!
interface Loopback1
 ip address 150.1.77.77 255.255.255.0

R1, R4::
router bgp 100
 network 150.1.77.0 mask 255.255.255.0 backdoor
```

Verification

> **Note**
>
> Check the route for the new prefix in R1's RIB before you applied the backdoor configuration. Based on eBGP AD, the prefix learned via BGP is preferred:

```
Rack1R1#show ip route 150.1.77.0
Routing entry for 150.1.77.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 200, type external
  Last update from 155.1.13.3 00:01:05 ago
  Routing Descriptor Blocks:
  * 155.1.13.3, from 155.1.13.3, 00:01:05 ago
      Route metric is 0, traffic share count is 1
      AS Hops 2
      Route tag 200
```

> **Note**
>
> Now apply the solution and see how the routing information has changed for the prefix. R1 now prefers the route learned via EIGRP over the eBGP path:

```
Rack1R1#sh ip route 150.1.77.0
Routing entry for 150.1.77.0/24
  Known via "eigrp 1", distance 90, metric 158720, type internal
  Redistributing via eigrp 1
  Last update from 155.1.146.6 on FastEthernet0/0, 00:00:05 ago
  Routing Descriptor Blocks:
  * 155.1.146.6, from 155.1.146.6, 00:00:05 ago, via FastEthernet0/0
      Route metric is 158720, traffic share count is 1
      Total delay is 5200 microseconds, minimum bandwidth is 100000
Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
```

Copyright © 2010 Internetwork Expert

Note

Look at the BGP table entry for the same prefix. As you can see, the prefix is marked with “RIB Failure” state. This means that BGP was unable to insert the best path into RIB, as there is another prefix with better AD.

```
Rack1R1#show ip bgp 150.1.77.0
BGP routing table entry for 150.1.77.0/24, version 22
Paths: (2 available, best #2, table Default-IP-Routing-Table, RIB-
failure(17))
Multipath: iBGP
  Advertised to update-groups:
     2
  200 300, (Received from a RR-client)
    155.1.45.5 (metric 27260160) from 155.1.146.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, internal
  200 300
    155.1.13.3 from 155.1.13.3 (150.1.3.3)
      Origin IGP, localpref 100, valid, external, best
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.30 BGP Aggregation

- Remove the BGP configuration made for the previous task.
- Configure R2 with four new Loopback interfaces with the IP addresses 10.0.0.1/24, 10.0.1.1/24, 10.0.2.1/24, and 10.0.3.1/24.
- Advertise an aggregate route for these networks that does not overlap any address space.

Configuration

Note

Route aggregation is the key for information hiding. It is critical to BGP due to the tremendous amount of routing information passed on the Internet. There are three basic ways to do summarization in BGP:

1) Create a summary prefix in IGP and advertise it into BGP using the **`network`** command. Most often this is accomplished by creating a static route to Null0 in the routing table of the advertising router. This is a common way to advertise local prefixes into BGP. However, you cannot summarize external BGP prefixes using the method.
2) Auto-summarization. As discussed in the separate task, summarizes networks to their classful boundaries. Only applies to redistributed prefixes or when using the classful **`network`** command. Not used in modern networks.
3) Summarization of prefixes found in BGP tables using the **`aggregate-address command`**. This is the most flexible way to do summarization, as it may be applied to any paths learned by the BGP speaker.

Through the next few tasks we are going to work with the last command. It has many options, and its unique feature is that is allows manipulating BGP attributes when aggregating prefixes. In the simplest form, the syntax for the command is **`aggregate-address <prefix> <mask>`**. In order for the command to work, there mask be a subnet in the BGP table that is encompassed by the summarized prefix. For example, if you issue the command **`aggregate-address 192.168.0.0 255.255.0.0`** then at least one subnet like 192.168.1.0/24 must be in the BGP table (Loc-RIB) and not the router's routing table (RIB).

Copyright © 2010 Internetwork Expert

If you didn't specify any additional options to the command, it will create a new prefix in the BGP table, with an empty AS_PATH. It would look like the new prefix was originated in the local AS. The new prefix will automatically have the weight value of 32768 and get a special attribute called ATOMIC_AGGREGATE assigned. The new attribute is informational, and tells the other BGP speakers that this prefix is a result of route aggregation and some information (like AS_PATH or other attributes) from the original prefixes may be missing. In addition to the ATOMIC_AGGREGATE attribute, BGP attaches another attribute called AGGREGATOR to the summarized prefix. This attribute specifies the AS number and the BGP router-ID of the aggregating router. Just like the ATOMIC_AGGREGATE, the new attribute is also informational.

For every aggregate, the BGP process will install an automatic static route to Null0 for the new prefix, in order to prevent routing loops. Keep in mind that the original (specific) prefixes are *still advertised*, unlike in IGP, where summarization automatically suppresses more specific prefixes.

In our scenario we need to come with a most effective summary prefix for 3 subnets. Using the classic summarization rules we write all prefixes in binary format:

```
10.0.0.1=00000110.00000000.000000|00.00000001
10.0.1.1=00000110.00000000.000000|01.00000001
10.0.2.1=00000110.00000000.000000|10.00000001
10.0.3.1=00000110.00000000.000000|11.00000001
```

Based on that output we select the maximum common part for all four prefixes, and shift the subnet prefix length by the number of bits stripped. The result is 10.0.0.0/22 or 10.0.0.0 255.255.252.0.

Copyright © 2010 Internetwork Expert

```
SW1:
router bgp 300
 no neighbor 155.1.67.6 shutdown
 no network 150.1.77.0 mask 255.255.255.0

R2:
interface Loopback 100
 ip address 10.0.0.1 255.255.255.0
!
interface Loopback 101
 ip address 10.0.1.1 255.255.255.0
!
interface Loopback 102
 ip address 10.0.2.1 255.255.255.0
!
interface Loopback 103
 ip address 10.0.3.1 255.255.255.0
!
router bgp 200
 network 10.0.0.0 mask 255.255.255.0
 network 10.0.1.0 mask 255.255.255.0
 network 10.0.2.0 mask 255.255.255.0
 network 10.0.3.0 mask 255.255.255.0
 aggregate-address 10.0.0.0 255.255.252.0
```

Copyright © 2010 Internetwork Expert

Verification

Note

Check the summary prefix in the BGP table of R2. Notice the atomic-aggregate attribute on this prefix, and the aggregator attribute value (aggregated by 200 150.1.2.2).

```
Rack1R2#show ip bgp 10.0.0.0/22
BGP routing table entry for 10.0.0.0/22, version 30
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2
  Local, (aggregated by 200 150.1.2.2)
    0.0.0.0 from 0.0.0.0 (150.1.2.2)
      Origin IGP, localpref 100, weight 32768, valid, aggregated,
local, atomic-aggregate, best
```

Note

Now check the summary route to Null0 installed in the routing table of R2:

```
Rack1R2#sh ip route 10.0.0.0 255.255.252.0
Routing entry for 10.0.0.0/22
  Known via "bgp 200", distance 200, metric 0, type locally generated
  Routing Descriptor Blocks:
  * directly connected, via Null0
      Route metric is 0, traffic share count is 1
      AS Hops 0
```

Note

Confirm that the summary prefix did not suppress the more specific routes. Take R4 for instance:

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ip bgp regexp _200$
BGP table version is 56, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 10.0.0.0/24      155.1.45.5                             0 200 i
* i                 155.1.13.3               0    100      0 200 i
*> 10.0.0.0/22      155.1.45.5                             0 200 i
* i                 155.1.13.3               0    100      0 200 i
*> 10.0.1.0/24      155.1.45.5                             0 200 i
* i                 155.1.13.3               0    100      0 200 i
*> 10.0.2.0/24      155.1.45.5                             0 200 i
* i                 155.1.13.3               0    100      0 200 i
*> 10.0.3.0/24      155.1.45.5                             0 200 i
* i                 155.1.13.3               0    100      0 200 i
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.31 BGP Aggregation - Summary Only

- Modify the configuration for the previous task so that no other devices but R2 could see the specific prefixes that make up the summary.

Configuration

> **Note**
>
> As you learned in the previous scenario, BGP summarization using the **aggregate-address** command creates new prefix in BGP table but does not suppress the advertisement of the specific prefixes that make up the summary. In order to generate just the summary prefix, use the option **summary-only** after the **aggregate-address** command. The BGP process will automatically suppress advertisement of the prefixes in the BGP table encompassed by the new summary address. This is probably the most common use of the aggregation command, as usually only the summarized prefix should be advertised.

```
R2:
router bgp 200
 aggregate-address 10.0.0.0 255.255.252.0 summary-only
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Look at R2's BGP table now. Notice that all specific prefixes are marked with the "s" sign, meaning "suppressed". They are not advertised to any peers, only the summary prefix is advertised.

```
Rack1R2#show ip bgp | include 10.0.
s> 10.0.0.0/24      0.0.0.0                  0         32768 i
*> 10.0.0.0/22      0.0.0.0                            32768 i
s> 10.0.1.0/24      0.0.0.0                  0         32768 i
s> 10.0.2.0/24      0.0.0.0                  0         32768 i
s> 10.0.3.0/24      0.0.0.0                  0         32768 i

Rack1R2#show ip bgp neighbors 155.1.0.5 advertised-routes
BGP table version is 35, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 10.0.0.0/22      0.0.0.0                            32768 i
*> 205.90.31.0      192.10.1.254             0             0 254 ?
*> 220.20.3.0       192.10.1.254             0             0 254 ?
*> 222.22.2.0       192.10.1.254             0             0 254 ?

Total number of prefixes 4
```

> **Note**
>
> Connectivity is preserved nonetheless, as the summary prefix is enough to reach back to the new interfaces of R2.

```
Rack1R2#ping 220.20.3.1 source loopback 101

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 220.20.3.1, timeout is 2 seconds:
Packet sent with a source address of 10.0.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/8 ms
```

Copyright © 2010 Internetwork Expert

7.32 BGP Aggregation - Suppress Map

- Modify the solution for the previous task so that R2 advertises 10.0.2.0/24 prefix in addition to the summary route.

Configuration

Note

When you specify the summary-only keyword, all specific prefixes are suppressed. It is possible to suppress prefixes selectively, using a route-map associated via the parameter suppress-map. The prefixes permitted by this route-map are suppressed; prefixes denied by this route-map are NOT suppressed when performing summarization. For example:

```
ip prefix-list SUPPRESS_PREFIX 150.1.1.0/24
!
route-map SUPPRESS_MAP permit 10
 match ip address prefix-list SUPPRESS_PREFIX
!
router bgp 200
 aggregate-address 150.1.0.0 mask 255.255.0.0 suppress-map SUPPRESS_MAP
```

Imagine that prefixes 150.1.1.0/24, 150.1.2.0/24 and 150.1.3.0/24 are in the BGP table. Then the command will produce new summary prefix 150.1.0.0/16 and suppress only one prefix: 150.1.1.0/24.

In our scenario, all prefixes are suppressed with except to 10.0.2.0/24. We use a prefix list to match the subnet and special “deny” statement in the route map to exclude this prefix from suppression. Other prefixes match the “permit” entry in the end of the route-map and are suppressed.

```
R2:
ip prefix-list NET_2 permit 10.0.2.0/24
!
route-map SUPPRESS_MAP deny 10
 match ip address prefix-list NET_2
!
route-map SUPPRESS_MAP permit 100
!
router bgp 200
 aggregate-address 10.0.0.0 255.255.252.0 suppress-map SUPPRESS_MAP
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Look at R2's BGP table again. Now you can see that the prefix 10.0.2.0/24 is not suppressed. Furthermore, you can confirm that it is being advertised to R2's peers.

```
Rack1R2#show ip bgp | include 10.0
s> 10.0.0.0/24      0.0.0.0                  0         32768 i
*> 10.0.0.0/22      0.0.0.0                            32768 i
s> 10.0.1.0/24      0.0.0.0                  0         32768 i
*> 10.0.2.0/24      0.0.0.0                  0         32768 i
s> 10.0.3.0/24      0.0.0.0                  0         32768 i

Rack1R2#show ip bgp neighbors 155.1.0.5 advertised-routes
BGP table version is 36, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 10.0.0.0/22      0.0.0.0                            32768 i
*> 10.0.2.0/24      0.0.0.0                  0         32768 i
*> 205.90.31.0      192.10.1.254             0             0 254 ?
*> 220.20.3.0       192.10.1.254             0             0 254 ?
*> 222.22.2.0       192.10.1.254             0             0 254 ?

Total number of prefixes 5
```

Copyright © 2010 Internetwork Expert

Note

You can observe the new aggregate prefix generation using the **debug ip bgp** command. Enable debugging and enter the aggregation command under the BGP process. Notice that the debug output shows the suppressed prefixes and explicitly states that 10.0.2.0/24 is not suppressed.

```
Rack1R2(config)#router bgp 200
Rack1R2(config-router)#do debug ip bgp
BGP debugging is on for address family: IPv4 Unicast
Rack1R2(config-router)#aggregate-address 10.0.0.0 255.255.252.0
suppress-map SUPPRESS_MAP
Rack1R2(config-router)#
BGP(0): Aggregate processing for IPv4 Unicast
BGP(0): For aggregate 10.0.0.0/22
BGP(0): 10.0.0.0/22 subtree has an entry 10.0.0.0/24
BGP(0): sub-prefix : 10.0.0.0/24
BGP(0): Needs to be re-aggregated
BGP(0): 10.0.0.0/22 subtree has an entry 10.0.0.0/24
BGP(0): 10.0.0.0/22 aggregate has 10.0.0.0/24 more-specific
BGP(0): 10.0.0.0/22 aggregate created, attributes updated
BGP(0): 10.0.0.0/22 subtree has an entry 10.0.0.0/24
BGP(0): Found sub-prefix 10.0.0.0/24: suppressed
BGP(0): Found sub-prefix 10.0.0.0/22:
BGP(0): Found sub-prefix 10.0.1.0/24: suppressed
BGP(0): Found sub-prefix 10.0.2.0/24: Not matched
BGP(0): Found sub-prefix 10.0.3.0/24: suppressed
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.33 BGP Aggregation - Unsuppress Map

- Remove the summarization configured in R2.
- Using the summary-only feature, configure R3 and R5 to originate an aggregate route for these networks that does not overlap any address space.
- Using the unsuppress-map feature configure the network in such a way that traffic from AS 100 and 54 going to the prefix 10.0.1.0/24 always transits AS 300 unless the link between R3 and SW1 is down.
- Traffic from these ASes going to other subnets of the aggregate should use the direct path through the network.

Configuration

Note

This scenario demonstrates one of the common uses for the **aggregate-address** command. Local networks are advertised into BGP and aggregated by the border BGP speakers.

It is often desirable to load-balancing traffic ingress to the local AS, so that traffic to some subnets enters via one BGP peer and the other peer is used as the entry point for other subnets. Generally, to accomplish this, you need to advertise all specific prefixes on both uplinks and use AS_PATH prepending to modify prefixes preference. This scheme implements load balancing and provides backup in case of any uplink failures.

However, it is possible to achieve the same goal using the different technique. It is based on the fact that classless routing always prefers the most specific prefix to reach the destinations. If there is a specific prefix in the routing table (Say 10.0.1.0/24) and the summary one (say 10.0.0.0/22) then the local router will prefer /24 and will use /22 only if the specific prefix vanishes. Thus, by configuring the border BGP peers for advertising both the summary and selected specific prefixes you may achieve the same load-balancing with the necessary level of redundancy.

To implement this technique, you may use the **unsuppress-map** BGP feature. This feature could be only configured on the router that performs prefix aggregation using the command **aggregate-address ... summary-only**. The feature uses a special route-map that matches and permits the prefixes need to be unsuppressed. The feature is applied only on per-neighbor basis as follows:

Copyright © 2010 Internetwork Expert

```
router bgp 100
 neighbor 150.1.37.7 unsuppress-map UNSUPPRESS
```

When the aggregate route is advertised to the selected peer, all the suppressed prefixes found in the local BGP table are matched against the configured **unsuppress-map**. The matching prefixes are advertised in addition to the summary prefix. Other peers or the local BGP table are not affected by this configuration.

In this scenario, R3 and R5 perform prefix aggregation. R3 is configured to unsuppress and advertise one specific prefix - 10.0.1.0/24 to SW1. When AS 54 receives all prefixes, it prefers to reach 10.0.1.0/24 via AS 300, as this is the way that explicit prefix has travelled to reach AS54.

```
R2:
router bgp 200
 no aggregate-address 10.0.0.0 255.255.252.0

R3:
ip prefix-list NET_1 permit 10.0.1.0/24
!
route-map UNSUPPRESS_MAP permit 10
 match ip address prefix-list NET_1
!
router bgp 200
 aggregate-address 10.0.0.0 255.255.252.0 summary-only
 neighbor 155.1.37.7 unsuppress-map UNSUPPRESS_MAP

R5:
router bgp 200
 aggregate-address 10.0.0.0 255.255.252.0 summary-only
```

Copyright © 2010 Internetwork Expert

Verification

Note

Start your verifications by looks at the BGP tables of R3 and R5. Notice that all specific prefixes are suppressed.

```
Rack1R3#show ip bgp | include 10.0
s i10.0.0.0/24      155.1.0.2                0    100      0 i
*> 10.0.0.0/22      0.0.0.0                               32768 i
s i10.0.1.0/24      155.1.0.2                0    100      0 i
s i10.0.2.0/24      155.1.0.2                0    100      0 i
s i10.0.3.0/24      155.1.0.2                0    100      0 i

Rack1R5#show ip bgp | include 10.0
s>i10.0.0.0/24      155.1.0.2                0    100      0 i
*> 10.0.0.0/22      0.0.0.0                               32768 i
s>i10.0.1.0/24      155.1.0.2                0    100      0 i
s>i10.0.2.0/24      155.1.0.2                0    100      0 i
s>i10.0.3.0/24      155.1.0.2                0    100      0 i
```

Note

Now look at the routes that R3 advertises to SW1. Notice that the prefix 10.0.1.0/24 is included, even though it is marked as “suppressed”. That means that the prefix was unsuppressed by the configured feature.

```
Rack1R3#show ip bgp neighbors 155.1.37.7 advertised-routes | include
10.0
*> 10.0.0.0/22      0.0.0.0                               32768 i
s>i10.0.1.0/24      155.1.23.2               0    100      0 i
```

Note

Now take any router in AS 100. Look up the route 10.0.1.0 in the local routing table. Notice that it points to SW1 and the number of AS hops is 2.

```
Rack1R6#show ip route 10.0.1.0
Routing entry for 10.0.1.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 300, type external
  Last update from 155.1.67.7 00:06:08 ago
  Routing Descriptor Blocks:
  * 155.1.67.7, from 155.1.67.7, 00:06:08 ago
      Route metric is 0, traffic share count is 1
      AS Hops 2
      Route tag 300
```

Copyright © 2010 Internetwork Expert

> **Note**
>
> At the same time, all unsuppressed prefixes are preferred via R1, as they are covered by the summary route.

```
Rack1R6#show ip route 10.0.2.0
Routing entry for 10.0.0.0/22
  Known via "bgp 100", distance 200, metric 0
  Tag 200, type internal
  Last update from 155.1.13.3 00:06:34 ago
  Routing Descriptor Blocks:
  * 155.1.13.3, from 155.1.146.1, 00:06:34 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
      Route tag 200
```

> **Note**
>
> If you look at R6's BGP table, you will notice that the path to 10.0.0.0/22 is via R3, while the path to 10.0.1.0/24 is via SW1.

```
Rack1R6#show ip bgp
BGP table version is 88, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  10.0.0.0/22      155.1.67.7                             0 300 200 i
*>i                 155.1.13.3               0    100      0 200 i
*> 10.0.1.0/24      155.1.67.7                             0 300 200 i
```

Copyright © 2010 Internetwork Expert

7.34 BGP Aggregation - AS-Set

- Configure R1 to aggregate the subnets 112.0.0.0/24-119.0.0.0/24 into one prefix using the optimal prefix length.
- Ensure that the new summary prefix is not accepted by AS 54 peers.
- Do not use any filtering technique to accomplish this.

Configuration

Note

It is important to remember that aggregation hides information previous found in the specific prefixes. This includes all attributes, such as NEXT_HOP, AS_PATH and so on. The new prefix appears to be originated from within the local AS. This causes no problems if all specific prefixes belong to the local AS. However, when you summarize prefixes learned from other ASes, information hiding may result in the following dangerous consequences:

1) Suboptimal routing, due to the loss of path information, such as AS_PATH, MED and so on.
2) Routing loops, as removal of AS_PATH attribute and replacement it with an empty list will prevent BGP loop-detection mechanism from working properly.

The second issue is most dangerous. In order to overcome it, it is possible to insert a special new member into the AS_PATH of the newly created summary prefix. This element is called AS_SET and contains the AS numbers found in all AS_PATHs of the specific prefixes. This list of AS numbers is unordered, unlike the regular AS_SEQUENCE element. It's only use is for routing loop prevention – when BGP receives a prefix it scans the AS_PATH attribute. If the local AS number is found in any of the AS_SET or AS_SEQUENCE elements, the prefix is dropped.

By default, the aggregated address in BGP will not include the AS-Set information. In order to force the use of this information, specify the as-set option as follows: aggregate-address <subnet> <mask> as-set.

In our scenario, AS100 speaker aggregates the prefixes learned from another ASes. If not the as-set feature, both BB1 and BB3 would have accepted the new summary prefix. If that would have happened, that in case when BB1 or BB3 lose route to say prefix 113.0.0.0 they would route to AS100 for this prefix, while AS 100 never had it. The solution aggregates the prefixes using the following bitmap breakdown:

Copyright © 2010 Internetwork Expert

```
1110|000.00000000.00000000.00000000
1110|001.00000000.00000000.00000000
...
1110|111.00000000.00000000.00000000
```

Which results in the summary prefix of 112.0.0.0/5 or the same as 112.0.0.0 248.0.0.0

```
R1:
router bgp 100
 aggregate-address 112.0.0.0 248.0.0.0 summary-only as-set
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> If you look at R1's BGP table, you will see that prefix 112.0.0.0/5 have the AS_PATH attribute in format {54,50,60} listing all AS numbers found in prefixes received from BB1 and BB3.

```
Rack1R1#show ip bgp
BGP table version is 85, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i10.0.0.0/22     155.1.45.5               0    100      0 200 i
*>                 155.1.13.3               0             0 200 i
*>i10.0.1.0/24     155.1.67.7               0    100      0 300 200 i
* i28.119.16.0/24  54.1.1.254               0    100      0 54 i
*>i                204.12.1.254             0    100      0 54 i
* i28.119.17.0/24  54.1.1.254               0    100      0 54 i
*>i                204.12.1.254             0    100      0 54 i
s i112.0.0.0       54.1.1.254               0    100      0 54 50 60 i
s>i                204.12.1.254             0    100      0 54 50 60 i
*> 112.0.0.0/5     0.0.0.0                       100  32768 {54,50,60} i
```

> **Note**
>
> Look at the detailed information for the new BGP prefix and notice the aggregator and the AS_SET attributes:

```
Rack1R1#show ip bgp 112.0.0.0 248.0.0.0
BGP routing table entry for 112.0.0.0/5, version 77
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2
  {54,50,60}, (aggregated by 100 150.1.1.1)
    0.0.0.0 from 0.0.0.0 (150.1.1.1)
      Origin IGP, localpref 100, weight 32768, valid, aggregated,
local, best
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.35 BGP Aggregation - Attribute-Map

- Configure R4 to mark the prefix 112.0.0.0/24 received from BB3 with the community value of “no-export”.
- Ensure this community value propagates across AS 100.
- Configure R1 so that the summary prefix 112.0.0.0/5 is still advertised to AS 300 and AS 200.

Configuration

Note

When you use the as-set parameter to the aggregate-address command, the resulting prefix will inherit “additive” attributes of the specific prefixes. This includes the AS_PATH attributes, condensed into AS_SET and the community attributes, which are grouped together from all prefixes. We will explore community signaling in details further, but so far, keep in mind that any prefix bearing the community attribute value of “no-export” is not advertised to the adjacent ASes.

In our scenario, we have R4 tagging just one prefix – 112.0.0.0/24 with the community value of “no-export”. However, when R1 aggregates all prefixes into one, the summary prefix inherits the “no-export” community from one of the specific routes. In effect, AS 100 speakers will not be able to advertise the summary prefix to the neighbors.

The solution to this problem is the use of the **`attribute-map`** parameter to the **`aggregate-address`** command. This parameter specifies the route-map that sets BGP attributes for the newly generated prefix. You may set any configuration BGP value, such as metric, origin, local-preference and so on. However, in our case we are interested in setting the community attribute value for the summary. The route-map applies the set community none command and erases all communities for the new prefix. Naturally, all routers are configured to propagate communities across AS 100.

Copyright © 2010 Internetwork Expert

```
R4:
no ip prefix-list NET_112
ip prefix-list NET_112 permit 112.0.0.0/8
!
no route-map SET_COMMUNITY
route-map SET_COMMUNITY permit 10
 match ip address prefix-list NET_112
 set community no-export
!
route-map SET_COMMUNITY permit 100
!
router bgp 100
 neighbor 204.12.1.254 route-map SET_COMMUNITY in
 neighbor 155.1.146.1 send-community

R1:
route-map ATTR_MAP
 set community none
!
router bgp 100
 aggregate-address 112.0.0.0 248.0.0.0 summary-only as-set attribute-
map ATTR_MAP
 neighbor 155.1.146.6 send-community
 neighbor 155.1.146.4 send-community
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the BGP table of R4 and confirm that the prefix 112.0.0.0/8 is marked with the community attribute “no-export”.

```
Rack1R4#show ip bgp 112.0.0.0 255.0.0.0
BGP routing table entry for 112.0.0.0/8, version 18
Paths: (1 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
Flag: 0x820
  Advertised to update-groups:
     2
  54 50 60
    204.12.1.254 from 204.12.1.254 (31.3.0.1)
      Origin IGP, localpref 100, valid, external, best
      Community: no-export
```

> **Note**
>
> Check the BGP tables of R1 for the prefix 112.0.0.0/5 before you applied the solution for this task to R1. Notice that the summary prefix has the “no-export” community attached as well. This prevents the summary prefix from being advertised to AS 100 and AS 300. Confirm that other speakers, such as R6 also have the summary prefix tagged with “no-export” community.

```
Rack1R1#show ip bgp 112.0.0.0 248.0.0.0
BGP routing table entry for 112.0.0.0/5, version 126
Paths: (1 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
Flag: 0x880
  Advertised to update-groups:
     2
  {54,50,60}, (aggregated by 100 150.1.1.1)
    0.0.0.0 from 0.0.0.0 (150.1.1.1)
      Origin IGP, localpref 100, weight 32768, valid, aggregated,
local, best
      Community: 54 no-export

Rack1R6#show ip bgp 112.0.0.0 248.0.0.0
BGP routing table entry for 112.0.0.0/5, version 92
Paths: (1 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
Flag: 0x880
  Not advertised to any peer
  {54,50,60}, (aggregated by 100 150.1.1.1)
    155.1.146.1 from 155.1.146.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Community: 54 no-export
```

Copyright © 2010 Internetwork Expert

Note

Now apply the solution to R1 and check the BGP table entry for 112.0.0.0/5 again. Confirm that the summary prefix does not have any community values now, and R1 advertises it to any eBGP peer.

```
Rack1R1#show ip bgp 112.0.0.0 248.0.0.0
BGP routing table entry for 112.0.0.0/5, version 127
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Flag: 0x880
  Advertised to update-groups:
     1           2           3
  {54,50,60}, (aggregated by 100 150.1.1.1)
    0.0.0.0 from 0.0.0.0 (150.1.1.1)
      Origin IGP, localpref 100, weight 32768, valid, aggregated,
local, best

Rack1R1#show ip bgp neighbors 155.1.13.3 advertised-routes
BGP table version is 127, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i10.0.1.0/24      155.1.67.7               0    100      0 300 200 i
*>i28.119.16.0/24   204.12.1.254             0    100      0 54 i
*>i28.119.17.0/24   204.12.1.254             0    100      0 54 i
*> 112.0.0.0/5      0.0.0.0                       100  32768 {54,50,60}
i
*>i155.1.0.0        155.1.146.4              0    100      0 i
```

Copyright © 2010 Internetwork Expert

7.36 BGP Aggregation - Advertise Map

- Configure R2 with two new Loopback interfaces with the IP addresses 222.22.0.1/24 and 222.22.1.1/24 and advertise them into BGP.
- Configure SW3 with a new Loopback interface with the IP address 222.22.3.1/24 and advertise it into BGP.
- Configure R4 and R6 to advertise the aggregate 222.22.0.0/22 into BGP. Include as much of the original AS-Path information as possible while still allowing devices in AS 300 to install the aggregate in the BGP table.

Configuration

Note

When using the as-set keyword with BGP aggregation, some of the specific prefix attributes got mixed together in the new prefix. Specifically, you should watch out for the resulting AS_SET and list of community attributes. In the previous task you have learned how to modify some of the aggregated prefix attributes. However, you cannot manipulate an important attribute such as AS_SET directly. Instead of this, you may specify the specific prefixes that will be used to make up the attribute list for the aggregate prefix. This is accomplished by using the **`advertise-map`** parameter to the **`aggregate-address`** command. The route-map used as advertise-map should permit specific prefixes to be used to compose the aggregate attributes, such as AS_SET. You can use only access-list, prefix-list or as-path access-lists to match the specific prefixes. Information from the prefixes denied by the route-map is not used when constructing the resulting summary-prefix. You may this method to remove the prefixes with unwanted BGP community attributes as well.

In our scenario, the AS_SET attribute for the summary route should be composed of AS numbers 200,254 and 300. However, by using the advertise-map parameter we filter out prefix originated in AS 300 and thus end up with AS_SET of {200,254}. This tricks AS 300 into accepting back the summary prefix.

Copyright © 2010 Internetwork Expert

R2:

```
interface Loopback220
 ip address 222.22.0.1 255.255.255.0
!
interface Loopback221
 ip address 222.22.1.1 255.255.255.0
!
router bgp 200
 network 222.22.0.0 mask 255.255.255.0
 network 222.22.1.0 mask 255.255.255.0
```

SW3:

```
interface Loopback 223
 ip address 222.22.3.1 255.255.255.0
!
router bgp 300
 network 222.22.3.0 mask 255.255.255.0
```

R4, R6:

```
ip prefix-list AS300_PREFIX permit 222.22.3.0/24
!
route-map ADVERTISE_MAP deny 10
 match ip address prefix-list AS300_PREFIX
!
route-map ADVERTISE_MAP permit 100
!
router bgp 100
 aggregate-address 222.22.0.0 255.255.252.0 summary-only as-set
advertise-map ADVERTISE_MAP
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the specific and the summary prefix at any BGP speaker that performs summarization. Notice the AS_PATH attributes of the specific prefixes, especially for the prefix 222.22.3.0/24. The AS_SET for the summary /22 does not include AS# 300.

```
Rack1R6#show ip bgp
BGP table version is 31, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
<snip>
*>i220.20.3.0       155.1.13.3               0    100      0 200 254 ?
*                   155.1.67.7                             0 300 200 254 ?
s>i222.22.0.0       155.1.13.3               0    100      0 200 i
s                   155.1.67.7                             0 300 200 i
*> 222.22.0.0/22    0.0.0.0                       100  32768 {200,254} ?
* i                 155.1.146.4              0    100      0 {200,254} ?
s>i222.22.1.0       155.1.13.3               0    100      0 200 i
s                   155.1.67.7                             0 300 200 i
s>i222.22.2.0       155.1.13.3               0    100      0 200 254 ?
s                   155.1.67.7                             0 300 200 254 ?
s i222.22.3.0       155.1.13.3               0    100      0 200 300 i
s>                  155.1.67.7                             0 300 i
```

> **Note**
>
> Now check SW1's BGP table and notice that the summary prefix is there:

```
Rack1SW1#show ip bgp
BGP table version is 19, local router ID is 150.1.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
<snip>
*> 222.22.0.0       155.1.37.3                             0 200 i
*> 222.22.0.0/22    155.1.67.6               0             0 100 {200,254} ?
*> 222.22.1.0       155.1.37.3                             0 200 i
*> 222.22.2.0       155.1.37.3                             0 200 254 ?
*>i222.22.3.0       155.1.79.9               0    100      0 i
```

Copyright © 2010 Internetwork Expert

Note

Load the the *Basic BGP Routing* initial configurations prior to continuing..

Copyright © 2010 Internetwork Expert

7.37 BGP Communities

- Configure AS200 to set local-preference attribute to 200 for eBGP prefixes tagged with community value 200:200
- Configure R1 in AS100 to signal AS200 to prefer path to the prefixes originated in AS 60 via R3.

Configuration

Note

BGP Communities are optional transitive attributes used mainly to associate an administrative tag to a route. All prefixes with the same community essentially belong to the same group and share some property. Using communities allows for manipulation of BGP prefixes based on their "meaning" (e.g. customer prefixes, peer prefixes, upstream networks) not the network values. Even though BGP community attribute is transitive, Cisco IOS routers do not pass it across BGP sessions by default. In order to start sending the community values to a particular peer, you have to activate this feature using the command **`neighbor <IP> send-community`**.

There could be many communities associated with a BGP prefix and every community attribute has a 32 bit value. There are two formats to read a community value: raw, as a 32-bit number and structured, as a pair **`AS Number:16-bit value`**. The second format makes it easier to interpret communities, as they naturally points to the AS that originated the tagging. Notice that by default community values are displayed in 32-bit format, and to use the structured notation you need to enter the global configuration command **`ip bgp-community new-format`**. Since there are no ASes numbered 0 or 65535, the BGP community ranges 0x0000:0x0000-0x0000:0xFFFF and 0xFFFF:0x0000-0xFFFF:0xFFFF are reserved and not available for public use. There are three well-known BGP community values from reserved range: NO_EXPORT (0xFFFF:0xFF01), NO_ADVERTISE (0xFFFF:0xFF02) and NO_EXPORT_SUBCONFED (0xFFFF:0xFF03) interpreted by all BGP speakers. We will discuss their use in separate tasks.

Copyright © 2010 Internetwork Expert

Other community values do not trigger any special BGP processing unless you configure your BGP speaker to do so. The most common use of communities is to signal a neighboring AS (since the attribute is transitive) some special sort of treatment to apply to the tagged prefixes. For example, imagine a company with its own AS, being a customer of two ISPs in separate ASes. Every ISP may implement a policy, where routes tagged with a community value say AS#:101 are prepended with ISP's AS# when advertised upstream. This allows the customer to instruct the ISPs for prepending of specific prefixes and thus adjusting BGP bestpath selection in the upstream AS. More advanced uses include implementing community-based filtering, such as using communities to signal "advertise this prefix only to directly connected peers" or using communities to signal QoS policy. As you can see, communities could be used to implement almost any configurable policy. However, the use of community values should be agreed between two ASes, as semantic interpretation is left to the administrator.

In order to set a community value, use the route-map command **`set community <value1> <value2> ... <valueN>`** or **`set community additive <value1> <value2> ... <valueN>`**. The former command will impose a new set of community values on the prefix. The other command will add the specified communities to the list already attached to the path. In order to match a community value, you need to configure a *community-list* and use it in a route-map later. There are two types of lists: standard and expanded. A list could be either numbered (1-99 for standard, 100-500 for expanded) or named. Standard community list entries permit or deny a community value, such as following:

```
ip community-list 1 permit 100:10 100:20
ip community-list 1 deny no-export
```

In order for an entry to match, all mentioned community values must exist in the prefix. For example, the first line in the example above would match only prefixes with two community values: 100:10 and 100:20. Essentially, OR logic is implemented by setting multiple entries, and AND logic is implemented for values in a single line.

Copyright © 2010 Internetwork Expert

Expanded community lists allow the use of regular expressions for community matching. This is helpful when you need to match a range or community values. Before applying an expanded community-list, BGP engine will order the communities for every prefix numerically (since the communities are just 32 bit values) and remove the duplicates. This allows for deterministic construction of regular expressions. We will illustrate the use of most common wildcard characters:

^100:1_200:1 – match communities 100:1 and 200:1 in the beginning (^ anchor) of the community list. Here "_" means the "space" separating different community values. Notice that the prefix may not have any community values ranged between 100:1 and 200:1 due to the community ordering. Also, the prefix may not have any community less than 100:1 as this is the first community in the list.

**300:2$** - match the community 300:2 at the end of the community list ($ anchor). The prefix may not have any community greater than 300:2 as this is the last value in the list.

400:[2-9]_ – match a range of communities ([] specify a rage) such as 300:1, 300:2,..., 300:9. The use of "_" at the end is important; otherwise this pattern would also match 300:22, 300:2333 and so on. This may be one of the most useful wildcard constructs for matching community ranges.

100:1.*_ - illustrates the use of two wildcard characters. The "." matches any digit (you may also use [0-9] in this context) and "*" means "repeat the previous pattern zero or more times". The pattern above would match 100:1, 100:2, 100:22, 100:11 and so on. You may use "+" instead of "*" which means "repeat one or more times". Thus **100:1.+_** will not match against 100:1.

100:([0-9]2)+_ - demonstrates the use of "()" for grouping. Here "[0-9]2" is treated as a single group for the operator "+". Thus the pattern would match 100:1212, 100:2222, 100:0202 and so on.

100:1_|100:2_ – here we use the alternation symbol "|". The pattern would match either 100:1 or 100:2. You may use "|" with grouping such as 100:(12)|(22) that will match 100:12 or 100:22.

Copyright © 2010 Internetwork Expert

In spite of expanded community lists flexibility, most times you need just the standard lists, as the number of destination groups is usually limited. In our scenario, AS 200 advertises to its peers that the community 200:200 is treated as having local preference of 200 inside the AS. In order to implement this policy, we configure inbound route-maps on R3 and R5 matching the community 200:200 and setting the local-preference to 200. R1 matches the prefixes originated in AS 60 and mark them with the community value of 200:200 signaling the preferred path.

R1:

```
no ip as-path access-list 1
ip as-path access-list 1 permit 60$
!
route-map SET_COMMUNITY permit 10
 match as-path 1
 set community 200:200
!
route-map SET_COMMUNITY permit 100

router bgp 100
 neighbor 155.1.13.3 send-community
 neighbor 155.1.13.3 route-map SET_COMMUNITY out
```

R3:

```
ip community-list standard 200:200 permit 200:200
!
route-map SET_LOCAL_PREFERENCE permit 10
 match community 200:200
 set local-preference 200
!
route-map SET_LOCAL_PREFERENCE permit 100
!
router bgp 200
 neighbor 155.1.13.1 route-map SET_LOCAL_PREFERENCE in
```

R5:

```
ip community-list standard 200:200 permit 200:200
!
route-map SET_LOCAL_PREFERENCE permit 10
 match community 200:200
 set local-preference 200
!
route-map SET_LOCAL_PREFERENCE permit 100
!
router bgp 200
 neighbor 155.1.45.4 route-map SET_LOCAL_PREFERENCE in
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the paths toward prefixes originated in AS 60 on R3. Notice the local preference of 200 associated with the path via R1.

```
Rack1R3#sh ip bgp regexp 60$
BGP table version is 21, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  112.0.0.0        155.1.37.7                             0 300 100 54 50 60 i
*>                  155.1.13.1                200          0 100 54 50 60 i
*  113.0.0.0        155.1.37.7                             0 300 100 54 50 60 i
*>                  155.1.13.1                200          0 100 54 50 60 i
```

> **Note**
>
> Look at the BGP attributes for 113.0.0.0/8. Notice the local preference and the community value. The community is presented in unstructured format. Configure the router to display the communities in structured format:

```
Rack1R3#sh ip bgp 113.0.0.0
BGP routing table entry for 113.0.0.0/8, version 21
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2           3
  300 100 54 50 60
    155.1.37.7 from 155.1.37.7 (150.1.7.7)
      Origin IGP, localpref 100, valid, external
  100 54 50 60
    155.1.13.1 from 155.1.13.1 (150.1.1.1)
      Origin IGP, localpref 200, valid, external, best
      Community: 13107400

Rack1R3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#ip bgp-community new-format
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#sh ip bgp 113.0.0.0
BGP routing table entry for 113.0.0.0/8, version 21
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2           3
  300 100 54 50 60
    155.1.37.7 from 155.1.37.7 (150.1.7.7)
      Origin IGP, localpref 100, valid, external
  100 54 50 60
    155.1.13.1 from 155.1.13.1 (150.1.1.1)
      Origin IGP, localpref 200, valid, external, best
      Community: 200:200
Rack1R3#
```

Note

Check that R5 prefers to reach 113.0.0.0/8 via R3 as well.

```
Rack1R5#show ip bgp 113.0.0.0
BGP routing table entry for 113.0.0.0/8, version 22
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           3
  100 54 50 60
    155.1.13.1 (metric 46111744) from 155.1.23.3 (150.1.3.3)
      Origin IGP, metric 0, localpref 200, valid, internal, best
  100 54 50 60
    155.1.45.4 from 155.1.45.4 (150.1.4.4)
      Origin IGP, localpref 100, valid, external
```

Copyright © 2010 Internetwork Expert

7.38 BGP Communities - No-Advertise

- Configure R2 so that it does not advertise prefixes received from AS 254 to any peer.
- Do not use any sort of prefix filtering to accomplish this.

Configuration

Note

The well-know NO_ADVERTISE BGP community signals BGP speaker not to advertise the particular prefix to any BGP peer. This may be useful to limit the scope of routing information to just directly connected neighbors. In our scenario, we set the community attribute inbound on prefixes received from BB2. This makes R2 think that the prefixes should not be advertised to any other peers.

```
R2:
route-map SET_COMMUNITY
 set community no-advertise
!
router bgp 200
 neighbor 192.10.1.254 route-map SET_COMMUNITY in
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the BGP prefixes received from AS 254. Look at the detailed information for any of the prefixes. Confirm that community no-advertise prevent route from being advertised to any peer.

```
Rack1R2#show ip bgp regexp 254
BGP table version is 33, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      192.10.1.254             0             0 254 ?
*> 220.20.3.0       192.10.1.254             0             0 254 ?
*> 222.22.2.0       192.10.1.254             0             0 254 ?

Rack1R2#show ip bgp 205.90.31.0
BGP routing table entry for 205.90.31.0/24, version 31
Paths: (1 available, best #1, table Default-IP-Routing-Table, not
advertised to any peer)
Flag: 0x880
  Not advertised to any peer
  254
    192.10.1.254 from 192.10.1.254 (222.22.2.1)
      Origin incomplete, metric 0, localpref 100, valid, external, best
      Community: no-advertise
```

Copyright © 2010 Internetwork Expert

7.39 BGP Communities - No-Export

- Modify R2'2 configuration so that AS254 prefixes are constrained to stay within AS 200 only.

Configuration

Note

The well-know NO_EXPORT community instructs the BGP speaker to advertise the prefix only across iBGP peering links. This restricts the prefix to remain within the boundaries of the local AS. One good use of this feature is prefix aggregation. Imagine that your local AS has multiple connections to some other. You advertise a summary of all your internal prefixes using the **aggregate-address** command out of all links. Yet you want just the adjacent AS to select the best entry point based on the MED attribute. This could be accomplished by advertising the specific prefixes tagged with NO_EXPORT community along with the aggregates. The neighboring AS would be able to select the best path based on the specific information (e.g. MED) but will not advertise the specifics any further, thus containing the routing information.

In our scenario, we simply tag all prefixes received from BB2 with the community NO_EXPORT. Since the community value must propagate to all peers, we enable sending community in all AS200's BGP speakers. There is no need to send BGP communities to all speakers, just make sure the border speakers always receive the community-tagged routes.

```
R2:
route-map SET_COMMUNITY permit 10
 no set community
 set community no-export
!
router bgp 200
 neighbor 155.1.23.3 send-community
 neighbor 155.1.0.5 send-community
 neighbor 192.10.1.254 route-map SET_COMMUNITY in

R3:
router bgp 200
 neighbor 155.1.0.5 send-community

R5:
router bgp 200
 neighbor 155.1.23.3 send-community
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Find the prefixes learned from AS 254. Check these prefixes in the BGP tables of R3 and R5. Make sure they are tagged with no-export community. For example, we verify one prefix – 205.90.31.0/24.

```
Rack1R2#show ip bgp regexp 254$
BGP table version is 26, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      192.10.1.254             0             0 254 ?
*> 220.20.3.0       192.10.1.254             0             0 254 ?
*> 222.22.2.0       192.10.1.254             0             0 254 ?

Rack1R3#show ip bgp 205.90.31.0
BGP routing table entry for 205.90.31.0/24, version 18
Paths: (2 available, best #2, table Default-IP-Routing-Table, not
advertised to EBGP peer)
  Advertised to update-groups:
     1           2
  254
    192.10.1.254 (metric 2560512256) from 155.1.0.5 (150.1.5.5)
      Origin incomplete, metric 0, localpref 100, valid, internal
      Community: no-export
      Originator: 150.1.2.2, Cluster list: 150.1.5.5
  254, (Received from a RR-client)
    192.10.1.254 (metric 2560512256) from 155.1.23.2 (150.1.2.2)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Community: no-export

Rack1R5#show ip bgp 205.90.31.0
BGP routing table entry for 205.90.31.0/24, version 18
Paths: (2 available, best #2, table Default-IP-Routing-Table, not
advertised to EBGP peer)
Flag: 0x880
  Advertised to update-groups:
     1           2
  254
    192.10.1.254 (metric 2560512256) from 155.1.23.3 (150.1.3.3)
      Origin incomplete, metric 0, localpref 100, valid, internal
      Community: no-export
      Originator: 150.1.2.2, Cluster list: 150.1.3.3
  254, (Received from a RR-client)
    192.10.1.254 (metric 2560512256) from 155.1.0.2 (150.1.2.2)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Community: no-export
```

Copyright © 2010 Internetwork Expert

Note

Confirm that AS254 prefixes are not among the prefixes advertised by AS 200 to other ASes. Check this by inspecting the advertised routes on AS200 border peers.

```
Rack1R3#show ip bgp nei 155.1.13.1 advertised-routes
BGP table version is 18, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   155.1.13.1                             0 100 54 i
*> 28.119.17.0/24   155.1.13.1                             0 100 54 i
*> 112.0.0.0        155.1.13.1                             0 100 54 50
60 i
*> 113.0.0.0        155.1.13.1                             0 100 54 50
60 i
*> 114.0.0.0        155.1.13.1                             0 100 54 i
*> 115.0.0.0        155.1.13.1                             0 100 54 i
*> 116.0.0.0        155.1.13.1                             0 100 54 i
*> 117.0.0.0        155.1.13.1                             0 100 54 i
*> 118.0.0.0        155.1.13.1                             0 100 54 i
*> 119.0.0.0        155.1.13.1                             0 100 54 i
*> 155.1.0.0        155.1.13.1                             0 100 i

Total number of prefixes 11

Rack1R3#show ip bgp nei 155.1.37.7 advertised-routes
BGP table version is 18, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   155.1.13.1                             0 100 54 i
*> 28.119.17.0/24   155.1.13.1                             0 100 54 i
*> 112.0.0.0        155.1.13.1                             0 100 54 50
60 i
*> 113.0.0.0        155.1.13.1                             0 100 54 50
60 i
*> 114.0.0.0        155.1.13.1                             0 100 54 i
*> 115.0.0.0        155.1.13.1                             0 100 54 i
*> 116.0.0.0        155.1.13.1                             0 100 54 i
*> 117.0.0.0        155.1.13.1                             0 100 54 i
*> 118.0.0.0        155.1.13.1                             0 100 54 i
*> 119.0.0.0        155.1.13.1                             0 100 54 i
*> 155.1.0.0        155.1.13.1                             0 100 i

Total number of prefixes 11
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.40 BGP Communities - Local-AS

- Re-configure R1 and R4 in the same BGP sub-confederation, using the AS# 65014. R6 should be in the sub-confederation 65006.
- Advertise R4's Loopback0 network in the BGP, but make sure that inside AS 100 only R1 receives it.

Configuration

Note

The well-know community Local-AS or NO_EXPORT_SUBCONFED in IETF RFC terms serves the same purpose as the NO_EXPORT community, but within a sub-confederation boundaries. That is, prefixes tagged by this community are not advertised to external sub-confederation peers (i.e. peers in other sub-confederations) AND to regular eBGP peers. In effect, the prefix is contained within a single sub-confederation. The use of Local-AS community is the same as of NO_EXPORT community, but only within the single confederation boundaries. For example, you may use it for routing optimization toward prefixes aggregated within a sub-confederation.

In our example, R4 advertises its local Loopback0 subnet into BGP and tags it with the Local-AS community. This prevents the prefix from leaking out of AS 65014 boundaries.

```
R1:
no router bgp 100
router bgp 65014
 bgp confederation identifier 100
 bgp confederation peers 65006
 neighbor 155.1.13.3 remote-as 200
 neighbor 155.1.146.4 remote-as 65014
 neighbor 155.1.146.6 remote-as 65006

R4:
route-map SET_COMMUNITY
 set community local-as
!
no router bgp 100
router bgp 65014
 bgp confederation identifier 100
 network 155.1.146.0 mask 255.255.255.0
 aggregate-address 155.1.0.0 255.255.0.0 summary-only
 neighbor 155.1.45.5 remote-as 200
 neighbor 155.1.146.1 remote-as 65014
 neighbor 155.1.146.1 send-community
 neighbor 204.12.1.254 remote-as 54
 network 150.1.4.0 mask 255.255.255.0 route-map SET_COMMUNITY
```

Copyright © 2010 Internetwork Expert

```
R6:
no router bgp 100
router bgp 65006
 bgp confederation identifier 100
 bgp confederation peers 65014
 network 155.1.146.0 mask 255.255.255.0
 aggregate-address 155.1.0.0 255.255.0.0 summary-only
 neighbor 54.1.1.254 remote-as 54
 neighbor 155.1.67.7 remote-as 300
 neighbor 155.1.146.1 remote-as 65014
```

Verification

> **Note**
>
> Check that the prefix is tagged with the community value of Local-AS and is not advertised to eBGP peers on R4:

```
Rack1R4#show ip bgp 150.1.4.4
BGP routing table entry for 150.1.4.0/24, version 13
Paths: (1 available, best #1, table Default-IP-Routing-Table, not
advertised outside local AS)
  Advertised to update-groups:
     1
  Local
    0.0.0.0 from 0.0.0.0 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, weight 32768, valid,
sourced, local, best
      Community: local-AS

Rack1R4#show ip bgp neigh 204.12.1.254 advertised-routes
BGP table version is 15, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   204.12.1.254             0             0 54 i
*> 28.119.17.0/24   204.12.1.254             0             0 54 i
*> 112.0.0.0        204.12.1.254                           0 54 50 60 i
*> 113.0.0.0        204.12.1.254                           0 54 50 60 i
*> 114.0.0.0        204.12.1.254                           0 54 i
*> 115.0.0.0        204.12.1.254                           0 54 i
*> 116.0.0.0        204.12.1.254                           0 54 i
*> 117.0.0.0        204.12.1.254                           0 54 i
*> 118.0.0.0        204.12.1.254                           0 54 i
*> 119.0.0.0        204.12.1.254                           0 54 i
*> 155.1.0.0        0.0.0.0                            32768 i

Total number of prefixes 11
Rack1R4#
```

Copyright © 2010 Internetwork Expert

Note

Now confirm that the prefix gets into R1's BGP table and is not advertised to any other peer, such as R6:

```
Rack1R1#show ip bgp 150.1.4.0
BGP routing table entry for 150.1.4.0/24, version 14
Paths: (1 available, best #1, table Default-IP-Routing-Table, not
advertised outside local AS, RIB-failure(17))
  Not advertised to any peer
  Local
    155.1.146.4 from 155.1.146.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, confed-internal, best
      Community: local-AS
Rack1R1#

Rack1R6#show ip bgp 150.1.4.0
% Network not in table
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.41 BGP Communities - Deleting

- Configure R2 to tag prefixes received from AS 254 with the community values “254:100”, “200:254” and “200:123”.
- Configure AS 300 to add the community value 300:200 to the list of communities and send them to AS 100.
- Configure AS 300 to remove any communities attached by AS 200, i.e. community starting with “200:” when sending prefixes to AS 100.

Configuration

Note

In complicated community signaling environment it may become necessary to remove a subset of communities associate with a prefix. For example, when passing a transit prefix to another AS, you may want to remove community values attached by downstream AS, yet retain community values attached by the AS of origin. Or you may want to delete the no-export community for a prefix, while leaving other communities intact.

Deleting a subset of community list is possible using special IOS feature. The configuration is as follows. First, you create a community access-list (standard or expanded). This list specifies the communities to be removed. It is flexible to use expanded access-lists to be able to remove community ranges – for example, by matching “200:[0-9]+_” you will erase any community set in AS 200. Then, you create a route-map to delete the communities using the following syntax: **`set comm-list {<NAME>|<NUMBER>} delete`**. You may set your own communities while deleting the other.

In our scenario, SW1 is configured to remove any communities matching the pattern “200:[0-9]+” and attach its own community “300:200”. This is performed using a single route-map entry.

Copyright © 2010 Internetwork Expert

```
R2:
route-map SET_COMMUNITY 10
 no set community
 set community 200:254 254:200 200:123
!
router bgp 200
 neighbor 155.1.23.3 send-community
 neighbor 155.1.0.5 send-community
 neighbor 192.10.1.254 route-map SET_COMMUNITY in
!
ip bgp-community new-format

R3:
router bgp 200
 neighbor 155.1.37.7 send-community
!
ip bgp-community new-format

R6:
ip bgp-community new-format

SW1:
ip community-list expanded AS200 permit 200:[0-9]+_
!
route-map RESET_COMMUNITY permit 10
 set community 300:200 additive
 set comm-list AS200 delete
!
router bgp 300
 neighbor 155.1.67.6 send-community
 neighbor 155.1.37.3 route-map RESET_COMMUNITY in
!
ip bgp-community new-format
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Trace any prefix from AS254 through the BGP tables of R3, SW1 and R6. Notice how the BGP communities associated with the prefix change every time.

```
Rack1R3#show ip bgp 205.90.31.0
BGP routing table entry for 205.90.31.0/24, version 13
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
     1           2           3           4
  254
    192.10.1.254 (metric 2560512256) from 155.1.0.5 (150.1.5.5)
      Origin incomplete, metric 0, localpref 100, valid, internal
      Community: 200:123 200:254 254:200
      Originator: 150.1.2.2, Cluster list: 150.1.5.5
  254, (Received from a RR-client)
    192.10.1.254 (metric 2560512256) from 155.1.23.2 (150.1.2.2)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Community: 200:123 200:254 254:200
Rack1R3#
```

Copyright © 2010 Internetwork Expert

Note

SW1 deletes all community values starting with “200:”

```
Rack1SW1#show ip bgp 205.90.31.0
BGP routing table entry for 205.90.31.0/24, version 49
Paths: (2 available, best #1, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
     2           3
  200 254
    155.1.37.3 from 155.1.37.3 (150.1.3.3)
      Origin incomplete, localpref 100, valid, external, best
      Community: 254:200 300:200
  100 200 254
    155.1.67.6 from 155.1.67.6 (150.1.6.6)
      Origin incomplete, localpref 100, valid, external
```

Note

The same set of communities is associated with the prefix in R6’s BGP table. Notice that only the prefix learned via AS 300 is tagged.

```
Rack1R6# show ip bgp 205.90.31.0
BGP routing table entry for 205.90.31.0/24, version 18
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     2
  300 200 254
    155.1.67.7 from 155.1.67.7 (150.1.7.7)
      Origin incomplete, localpref 100, valid, external
      Community: 254:200 300:200
  (65014) 200 254
    155.1.13.3 (metric 27260160) from 155.1.146.1 (150.1.1.1)
      Origin incomplete, metric 0, localpref 100, valid, confed-
external, best
```

Copyright © 2010 Internetwork Expert

Note

Load the the *Basic BGP Routing* initial configurations prior to continuing.

7.42 BGP Conditional Advertisement

- Configure R3 in such a way that AS 300 uses AS 100 to get to all prefixes learned from AS 254.
- If the link between R1 and R3 goes down traffic from AS 300 to AS 254 should be rerouted directly to AS 200.

Configuration

Note

Conditional advertisement allows BGP speaker to advertise a set of BGP prefixes to a peer only if certain other prefixes are present or not present in the local BGP table. Thus, the existence (or non-existence) of those "trigger" prefixes is used as a condition to advertise selected prefixes to a peer. Conditional advertisements are helpful in situations with multiple uplinks to different ASes. For example, assume that your AS is connected to a pair of ISPs. One ISP charges you more, and thus you only want to use it in case of emergency with the primary ISP. Based on this, you want to advertise your local prefixes to the "expensive" ISP only in case if the primary fails. In order to detect the primary ISP failure, you track a certain prefix learned from the primary ISP. This allows for more sophisticated tracking of the connectivity issue, compared to simply tracking the interface state. Once the tracked prefix is gone, all local prefixes are announced to the "expensive" ISP, until the moment the tracked prefix appears again.

The syntax for conditional advertisement is as follows:

```
neighbor <IP> advertise-map MAP1 {non-exist|exist-map} MAP2
```

Copyright © 2010 Internetwork Expert

The configuration involves defining two route-maps. One route-map (MAP1) selects the prefixes to be advertised to the peer. These prefixes must already exist in the local BGP table. The other route-map (MAP2) selects the prefixes to be tracked in the local BGP table. If this is a “non-exist” map, than condition is triggered when no prefixes in the BGP table match the route-map. If this is an “exist” map, then condition is triggered when there is a prefix in the BGP table matching the route-map. BGP process performs condition verification every time BGP scanner runs (60 seconds by default) so it may take some time after your configuration change before the conditional advertisement occurs.

In our scenario we advertise the link connecting R1 and R3 into BGP. We then create a route-map matching this prefix. This route-map is used as a “non-exist” condition for the advertisement of AS 254 prefixes. The prefixes are selected using an AS_PATH access-list matching the regular expression “254$”

```
R3:
ip as-path access-list 1 permit 254$
!
route-map ADVERTISE_MAP permit 10
 match as-path 1
!
ip prefix-list LINK_R1_R3 permit 155.1.13.0/24
!
route-map NON_EXIST_MAP permit 10
 match ip address prefix-list LINK_R1_R3
!
router bgp 200
 network 155.1.13.0 mask 255.255.255.0
 neighbor 155.1.37.7 advertise-map ADVERTISE_MAP non-exist-map
NON_EXIST_MAP
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> First, we check the "normal" conditions: the link between R1 and R3 is up. The prefix is in R3's BGP table.

```
Rack1R3#show ip bgp 155.1.13.0
BGP routing table entry for 155.1.13.0/24, version 3
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2           3           4
  Local
    0.0.0.0 from 0.0.0.0 (150.1.3.3)
      Origin IGP, metric 0, localpref 100, weight 32768, valid,
sourced, local, best
```

> **Note**
>
> AS 254 prefixes are not advertised to SW1, even though they are in the local BGP table.

```
Rack1R3#show ip bgp regexp 254$
BGP table version is 33, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i205.90.31.0      192.10.1.254             0    100      0 254 ?
*>i                 192.10.1.254             0    100      0 254 ?
* i220.20.3.0       192.10.1.254             0    100      0 254 ?
*>i                 192.10.1.254             0    100      0 254 ?
* i222.22.2.0       192.10.1.254             0    100      0 254 ?
*>i                 192.10.1.254             0    100      0 254 ?
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#show ip bgp neighbors 155.1.37.7 advertised-routes
BGP table version is 16, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   155.1.13.1                             0 100 54 i
*> 28.119.17.0/24   155.1.13.1                             0 100 54 i
*> 112.0.0.0        155.1.13.1                             0 100 54 50
60 i
*> 113.0.0.0        155.1.13.1                             0 100 54 50
60 i
*> 114.0.0.0        155.1.13.1                             0 100 54 i
*> 115.0.0.0        155.1.13.1                             0 100 54 i
*> 116.0.0.0        155.1.13.1                             0 100 54 i
*> 117.0.0.0        155.1.13.1                             0 100 54 i
*> 118.0.0.0        155.1.13.1                             0 100 54 i
*> 119.0.0.0        155.1.13.1                             0 100 54 i
*> 155.1.0.0        155.1.13.1                             0 100 i
*> 155.1.13.0/24    0.0.0.0                  0         32768 i

Total number of prefixes 12
```

Note

Check the state of the advertise map associated with SW1. Notice that status is "Withdraw" meaning that prefixes matching the advertise-map are not advertised to SW1.

```
Rack1R3#show ip bgp neighbors 155.1.37.7
BGP neighbor is 155.1.37.7,  remote AS 300, external link
<snip>
  Condition-map NON_EXIST_MAP, Advertise-map ADVERTISE_MAP, status:
Withdraw
<snip>
```

Note

Now shutdown the link connecting R1 and R3 and check the prefixes advertised to SW1. Notice that AS254 prefixes are now advertised.

Copyright © 2010 Internetwork Expert

```
R3:
interface Serial 1/2
 shutdown

Rack1R3#show ip bgp neighbors 155.1.37.7 advertised-routes
BGP table version is 33, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   155.1.45.4               0    100      0 100 54 i
*>i28.119.17.0/24   155.1.45.4               0    100      0 100 54 i
*>i112.0.0.0        155.1.45.4               0    100      0 100 54 50
60 i
*>i113.0.0.0        155.1.45.4               0    100      0 100 54 50
60 i
*>i114.0.0.0        155.1.45.4               0    100      0 100 54 i
*>i115.0.0.0        155.1.45.4               0    100      0 100 54 i
*>i116.0.0.0        155.1.45.4               0    100      0 100 54 i
*>i117.0.0.0        155.1.45.4               0    100      0 100 54 i
*>i118.0.0.0        155.1.45.4               0    100      0 100 54 i
*>i119.0.0.0        155.1.45.4               0    100      0 100 54 i
*>i155.1.0.0        155.1.45.4               0    100      0 100 i
*>i205.90.31.0      192.10.1.254             0    100      0 254 ?
*>i220.20.3.0       192.10.1.254             0    100      0 254 ?
*>i222.22.2.0       192.10.1.254             0    100      0 254 ?

Total number of prefixes 14
```

Note

Check the advertise-map associated with SW1 and confirm that the status is now "Advertise".

```
Rack1R3#show ip bgp neighbors 155.1.37.7
BGP neighbor is 155.1.37.7,  remote AS 300, external link
<snip>
  Condition-map NON_EXIST_MAP, Advertise-map ADVERTISE_MAP, status:
Advertise
<snip>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.43 BGP Conditional Route Injection

- Configure R2 with four new Loopback interfaces with the IP addresses 10.0.0.1/24, 10.0.1.1/24, 10.0.2.1/24 & 10.0.3.1/24 and advertise them into BGP.
- Configure R2 to originate an aggregate route for these networks that does not overlap any address space. Ensure no other devices in the BGP network see the individual subnet routes of this aggregate.
- Configure BGP Conditional Route Injection on R4 and R6 in such a way that traffic from AS 54 going to the subnet 10.0.1.0/24 enters via R4, while traffic to the subnet 10.0.2.0/24 enters via R6.
- Do not allow the more specific routes to be advertised to R1 from either R6 or R4, and prevent the formation of a routing loop on SW1.

Configuration

Note

Conditional Route Injection (CRI) is special feature that allows a BGP speaker to "de-aggregate" a particular prefix. As you know, aggregation is critical to large-scale routing in order to reduce routing table size and increase stability. CRI operation is opposite to aggregation, and its purpose is to create specific prefixes at administrator's discretion. This may be useful to optimize routing by advertising some specific subnets of the aggregate across a certain path. CRI is similar to BGP unsuppress-map feature, but it will work on any router, not just the one originating the aggregate prefix. However, due to the lack of information about the prefixes that were summarized, you have to explicitly set the prefixes to be injected into BGP table.

Copyright © 2010 Internetwork Expert

In order to configure CRI, you need two route-maps. The first route-map specifies the prefixes to be injected into the BGP table by means of **set ip address prefix-list <MAP1>** command. The **le** and **ge** keywords in the prefix-list entries are ignored. In addition to setting the prefixes, you may also set other BGP attributes, such as Weight, Local Preference, Origin, Metric, Community list and so on. The AS_PATH attribute is reset to an empty list, to reflect the fact that prefixes were originated in the local AS. By default, the new prefixes don't have a Local Preference value assigned and the Weight attribute is reset to zero (unlike 32768 for locally originated prefixes). This could be changed by setting these values manually. The second route-map defines the conditions that must be met for the new prefixes to be injected. This route-map must have two match statements. The first statement is **match ip address prefix-list <MAP2>** and it matches the prefix list defining the aggregated prefix. The second statement is **match ip route-source prefix-list <NAME>**. This prefix-list should match the IP address of the BGP peer that advertised the aggregate to the local router. Keep in mind that this is NOT the NEXT_HOP attribute of the aggregate prefix. It is the IP address used to establish the BGP session with a peer that sent the update to the local system. The two route-maps are then used as follows:

```
route bgp <AS#>
 bgp inject-map <MAP1> exist-map <MAP2>
```

The result is that prefixes matching MAP1 are injected in the local BGP table if the conditions specified by MAP2 have been met.

Copyright © 2010 Internetwork Expert

R2:

```
int loopback 100
ip address 10.0.0.1 255.255.255.0
int loopback 101
ip address 10.0.1.1 255.255.255.0
int loopback 102
ip address 10.0.2.1 255.255.255.0
int loopback 103
ip address 10.0.3.1 255.255.255.0

router bgp 200
network 10.0.0.0 mask 255.255.255.0
network 10.0.1.0 mask 255.255.255.0
network 10.0.2.0 mask 255.255.255.0
network 10.0.3.0 mask 255.255.255.0
aggregate-address 10.0.0.0 255.255.252.0 summary-only
```

R4:

```
ip prefix-list INJECT_PREFIX permit 10.0.1.0/24

ip prefix-list AGGREGATE permit 10.0.0.0/22
ip prefix-list ROUTE_SOURCE permit 155.1.146.1/32

route-map INJECT_MAP permit 10
 set ip address prefix-list INJECT_PREFIX
 set origin igp

route-map EXIST_MAP permit 10
 match ip address prefix-list AGGREGATE
 match ip route-source ROUTE_SOURCE

route-map DENY_INJECT_PREFIX deny 10
 match ip address prefix-list INJECT_PREFIX
 route-map DENY_INJECT_PREFIX permit 100

router bgp 100
 bgp inject-map INJECT_MAP exist-map EXIST_MAP
 neighbor 155.1.146.1 route-map DENY_INJECT_PREFIX out
```

Copyright © 2010 Internetwork Expert

R6:

```
ip prefix-list INJECT_PREFIX permit 10.0.2.0/24

ip prefix-list AGGREGATE permit 10.0.0.0/22
ip prefix-list ROUTE_SOURCE permit 155.1.146.1/32

route-map INJECT_MAP permit 10
 set ip address prefix-list INJECT_PREFIX
 set origin igp

route-map EXIST_MAP permit 10
 match ip address prefix-list AGGREGATE
 match ip route-source ROUTE_SOURCE

route-map DENY_INJECT_PREFIX deny 10
 match ip address prefix-list INJECT_PREFIX
 route-map DENY_INJECT_PREFIX permit 100

router bgp 100
 bgp inject-map INJECT_MAP exist-map EXIST_MAP
 neighbor 155.1.67.7 route-map DENY_INJECT_PREFIX out
 neighbor 155.1.146.1 route-map DENY_INJECT_PREFIX out
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> First check R6 and R4's BGP routing table. Confirm that the aggregate prefix is there. After this, check that paths injected into the BGP table. Notice that the NEXT_HOP attribute for these prefixes it taken from the aggregate prefix.

```
Rack1R6#show ip bgp 10.0.0.0 255.255.252.0
BGP routing table entry for 10.0.0.0/22, version 68
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Flag: 0x800
  Advertised to update-groups:
     1
  300 200, (aggregated by 200 150.1.2.2)
    155.1.67.7 from 155.1.67.7 (150.1.7.7)
      Origin IGP, localpref 100, valid, external, atomic-aggregate
  200, (aggregated by 200 150.1.2.2)
    155.1.13.3 (metric 27260160) from 155.1.146.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal, atomic-
aggregate, best

Rack1R4#sh ip bgp 10.0.0.0 255.255.252.0
BGP routing table entry for 10.0.0.0/22, version 22
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
        1    2
  200, (aggregated by 200 150.4.2.2)
    155.1.45.5 from 155.1.45.5 (150.1.5.5)
      Origin IGP, localpref 100, valid, external, atomic-aggregate,
best
  200, (aggregated by 200 150.1.2.2)
    155.1.13.3 (metric 27260160) from 155.1.146.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal, atomic-
aggregate
Rack4R4#

Rack1R6#show ip bgp injected-paths
BGP table version is 70, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 10.0.2.0/24      155.1.67.7                             0 i
* i                 155.1.13.3                             0 i
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#sh ip bgp injected-paths
BGP table version is 26, local router ID is 150.4.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i10.0.1.0/24     155.1.13.3                             0 i
*>                 155.1.45.5                             0 i
```

Note

Now check the advertised prefixes. In order for AS 54 to select the paths to 10.0.1.0/24 and 10.0.2.0/24 via R4 and R6 respectively:

```
Rack1R6#sh ip bgp nei 54.1.1.254 advertised-routes | include 10.0.
*>i10.0.0.0/22     155.1.13.3               0    100      0 200 i
*> 10.0.2.0/24     155.1.67.7                             0 i
Rack4R6#

Rack1R4#sh ip bgp nei 204.12.1.254 advertised-routes | inc 10.0.
*> 10.0.0.0/22     155.1.45.5                             0 200 i
*> 10.0.1.0/24     155.1.45.5                             0 i
Rack4R4#
```

Copyright © 2010 Internetwork Expert

> **Note**
> And make sure the prefixes are not advertised to R1 (the route reflector).

```
Rack1R6#show ip bgp neighbors 155.1.146.1 advertised-routes
BGP table version is 72, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   54.1.1.254                             0 54 i
*> 28.119.17.0/24   54.1.1.254                             0 54 i
*> 112.0.0.0        54.1.1.254                0            0 54 50 60 i
*> 113.0.0.0        54.1.1.254                0            0 54 50 60 i
*> 114.0.0.0        54.1.1.254                0            0 54 i
*> 115.0.0.0        54.1.1.254                0            0 54 i
*> 116.0.0.0        54.1.1.254                0            0 54 i
*> 117.0.0.0        54.1.1.254                0            0 54 i
*> 118.0.0.0        54.1.1.254                0            0 54 i
*> 119.0.0.0        54.1.1.254                0            0 54 i
*> 155.1.0.0        0.0.0.0                              32768 i

Total number of prefixes 11

Rack1R4#sh ip bgp nei 155.1.146.1 advertised-routes
BGP table version is 26, local router ID is 150.4.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 10.0.0.0/22      155.1.45.5                             0 200 i
*> 28.119.16.0/24   204.12.1.254           0               0 54 i
*> 28.119.17.0/24   204.12.1.254           0               0 54 i
*> 112.0.0.0        204.12.1.254                           0 54 50 60 i
*> 113.0.0.0        204.12.1.254                           0 54 50 60 i
*> 114.0.0.0        204.12.1.254                           0 54 i
*> 115.0.0.0        204.12.1.254                           0 54 i
*> 116.0.0.0        204.12.1.254                           0 54 i
*> 117.0.0.0        204.12.1.254                           0 54 i
*> 118.0.0.0        204.12.1.254                           0 54 i
*> 119.0.0.0        204.12.1.254                           0 54 i
*> 155.4.0.0        0.0.0.0                            32768 i
*> 205.90.31.0      155.1.45.5                             0 200 254 ?
*> 220.20.3.0       155.1.45.5                             0 200 254 ?
*> 222.22.2.0       155.1.45.5                             0 200 254 ?

Total number of prefixes 15
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.44 BGP Filtering with Prefix-Lists

- Configure a prefix-list on R2 so that it does not accept the prefix 222.22.2.0/24 from BB2; this prefix-list should be applied directly to the neighbor.
- Configure a prefix-list on R4 so that it does not accept any prefixes with a subnet mask greater than /22 from BB3; this prefix-list should be applied through a route-map to the neighbor.

Configuration

Note

Prefix lists are the most preferable way to filter subnets in BGP based on their IP addressing information. Prefix list is an ordered sequence of entries, where each entry specifies either a single IP prefix or a range of prefixes. Prefix lists are stored in efficient data structures allowing for very fast lookup and information retrieval. They have certain performance benefits over the standard and extended IOS access-lists when used for prefix filtering. Here is the syntax for a typical prefix-list entry:

```
ip prefix-list <NAME> seq <Num> {permit|deny}
<Subnet>/<Prefix > [ge <Length1>] [le <Length2>]
```

Entries in a prefix list are processed sequentially, until the first match. As soon as the match is found, the processing is stopped and associated action performed. The <Subnet>/<Prefix> pair specifies the major subnet that all prefix matching this entry should belong to. For example this could be 192.168.0.0/16 or 172.16.8.0/24 and so on – any valid classless prefix. The modifiers **ge** and **le** are optional and used to specify a prefix range. Specifically, a prefix matches the entry if:

a) The prefix is a subnet of **<Subnet>/<Prefix>**, i.e. the prefix subnet is a subset of <Subnet> and prefix-length is greater than or equal than <Prefix>.

b) The prefix length is less than or equal to **<Length2>**. That is, if the **le** modifier is used, then the prefix length must be within the **[<Prefix>, <Length2>]** range. For example, with **192.168.0.0/16 le 24** an example of valid prefix is 192.168.2.0/24 or 192.168.0.0/22 as both prefixes are subnets to 192.168.0.0/16 and have prefix-length less than or equal to 24. However, 192.168.2.128/25 will not match the above prefix-list entry.

Copyright © 2010 Internetwork Expert

c) The prefix length is greater than or equal to **<Length1>** but less than 32 is the **ge** modifier is used. That is, the prefix-length should be within the **[<Length1>,32]** range. It's is obvious that **<Length1>** should be greater than or equal than **<Prefix>**. Take for example prefix-list entry **172.16.3.0/24 ge 25**. It would match 172.16.3.128/25, 172.16.3.0/30, 172.16.3.1/32 but not the 172.16.3.0/24.

If both **le** and **ge** modifiers are in use, the resulting prefix-length range is between <Length1> and <Length2> inclusive. For example, 172.16.0.0/16 ge 24 le 30 would match 172.16.0.0/24, 172.16.3.0/24, 172.16.3.252/30 and so on.

Two common questions with prefix-lists is how to match the default route and match all prefixes. The entries are permit **0.0.0.0/0** and permit **0.0.0.0/0 le 32** respectively. The first entry matches the prefix with the prefix-length of zero and the network part of 0.0.0.0. The second entry matches any subnet of 0.0.0.0/0 which encompasses the whole IPv4 address space.

Prefix lists could be applied directly to a BGP peer using the command **neighbor <IP> prefix-list <NAME {in|out}** or using a route-map matching the prefix-list. The latter is a preferable way, as it allows you for more flexible policy editing.

```
R2:
ip prefix-list BLOCK_222 deny 222.22.2.0/24
ip prefix-list BLOCK_222 permit 0.0.0.0/0 le 32
!
router bgp 200
 neighbor 192.10.1.254 prefix-list BLOCK_222 in

R4:
ip prefix-list SHORTER_THAN_22 permit 0.0.0.0/0 le 22
!
route-map FROM_BB3 permit 100
 match ip address prefix-list SHORTER_THAN_22
!
router bgp 100
 neighbor 204.12.1.254 route-map FROM_BB3 in
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Compare the BGP table of R2 before and after you applied the prefix list.

```
Rack1R2#show ip bgp regexp 254$
BGP table version is 29, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      192.10.1.254             0             0 254 ?
*> 220.20.3.0       192.10.1.254             0             0 254 ?
*> 222.22.2.0       192.10.1.254             0             0 254 ?
```

```
Rack1R2#show ip bgp regexp 254$
BGP table version is 30, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      192.10.1.254             0             0 254 ?
*> 220.20.3.0       192.10.1.254             0             0 254 ?
```

> **Note**
>
> Compare the BGP tables of R4 and R6. Notice that /24 prefixes are only learned by R6, while R4 filters then out.

```
Rack1R6#show ip bgp regexp _54$
BGP table version is 32, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   54.1.1.254                             0 54 i
*> 28.119.17.0/24   54.1.1.254                             0 54 i
*> 114.0.0.0        54.1.1.254               0             0 54 i
* i                 204.12.1.254             0    100      0 54 i
*> 115.0.0.0        54.1.1.254               0             0 54 i
* i                 204.12.1.254             0    100      0 54 i
*> 116.0.0.0        54.1.1.254               0             0 54 i
* i                 204.12.1.254             0    100      0 54 i
*> 117.0.0.0        54.1.1.254               0             0 54 i
* i                 204.12.1.254             0    100      0 54 i
```

Copyright © 2010 Internetwork Expert

```
*> 118.0.0.0        54.1.1.254               0             0 54 i
* i                 204.12.1.254             0    100      0 54 i
*> 119.0.0.0        54.1.1.254               0             0 54 i
* i                 204.12.1.254             0    100      0 54 i
Rack1R6#
```

```
Rack1R4#show ip bgp regexp _54$
BGP table version is 26, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   54.1.1.254               0    100      0 54 i
*>i28.119.17.0/24   54.1.1.254               0    100      0 54 i
*> 114.0.0.0        204.12.1.254                           0 54 i
*> 115.0.0.0        204.12.1.254                           0 54 i
*> 116.0.0.0        204.12.1.254                           0 54 i
*> 117.0.0.0        204.12.1.254                           0 54 i
*> 118.0.0.0        204.12.1.254                           0 54 i
*> 119.0.0.0        204.12.1.254                           0 54 i
```

Copyright © 2010 Internetwork Expert

7.45 BGP Filtering with Standard Access-Lists

- Remove the filters applied in the previous task.
- Configure a standard access-list on R2 so that it does not accept any prefix with the address 222.22.2.0 from BB2; this access-list should be applied directly to the neighbor.
- Configure a standard access-list on R4 so that it does not accept any prefixes with an even number in the first octet; this access-list should be applied through a route-map to the neighbor.

Configuration

Note

Using standard access-list for BGP filtering is not as performance-effective as using prefix lists. In addition, a standard access-list does not allow you specifying the prefix-length, only the subnet numbers. Still, this approach offers some unique features not available with prefix-lists, such as selecting a range of subnet numbers.

As you remember, a standard access-list selects a subnet based on the pre-defined number and a wildcard mask. The wildcard mask is a 32-bit value that specifies the bits in the subnet numbers to be ignored. If a bit in the wildcard mask is 0, then the respective subnet bit value is “fixed”. If a bit in the wildcard mask is 1, then the respective subnet bit value could be either 0 or 1. For example, take the combination 192.168.0.0 0.0.255.255. Leading 16 bits of the subnet number are fixed at their value of “192.168”. However, the remaining 16 bits could take any value, as the wildcard mask bits are all ones. Thus the construct would match 192.168.0.1, 192.168.2.0, 192.168.32.128 and so on.

Since the wildcard mask does not represent the prefix subnet mask, you may make it discontinuous. This allows for some “oddball” filtering, such as permitting odd/even prefixes. Due to the binary nature of the wildcard mask, the number of prefixes selected is always a power of 2, i.e. 2^0, 2^1, 2^2 and so on. For example, the following combination “23.0.1.0 14.0.0.255” would match 8x256 subnets (3 bits set to one in the first octet and 8 bits set to 1 in the last octet) such as 23.0.1.64, 21.0.1.128, 17.0.1.32 and so on. The key is to transform the subnet number into binary format and apply the wildcard mask by walking over all possible combinations.

Copyright © 2010 Internetwork Expert

In this scenario we create an access-list that matches all prefixes with the odd first octet. This means the first octet must always have the lowest-significant bit set to one. This results in the corresponding wildcard bit set to 0 all the time. All other bits don't matter, so we can set the remaining wildcard bits to ones. The resulting combination is "1.0.0.0 254.255.255.255".

In order to associate the access-list with a BGP peer use the command **neighbor <IP> distribute-list {in|out}**. Notice that you cannot use prefix-list based and access-list based filtering at the same time, i.e. you cannot apply the **distribute-list/prefix-list** commands at the same time for the same peer. However, you may freely mix those command in a route-map.

```
R2:
ip access-list standard BLOCK_222
 deny 222.22.2.0
 permit any
!
router bgp 200
 no neighbor 192.10.1.254 prefix-list BLOCK_222 in
 neighbor 192.10.1.254 distribute-list BLOCK_222 in

R4:
ip access-list standard ODD_FIRST_OCTET
 permit 1.0.0.0 254.255.255.255
!
no route-map FROM_BB3
!
route-map FROM_BB3 permit 100
 match ip address ODD_FIRST_OCTET
!
router bgp 100
 neighbor 204.12.1.254 route-map FROM_BB3 in
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check R4's BGP table. Notice that prefixes received from BB3 all have odd first octet value.

```
Rack1R4#show ip bgp regexp _54$
BGP table version is 34, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   54.1.1.254               0    100      0 54 i
*>i28.119.17.0/24   54.1.1.254               0    100      0 54 i
*>i114.0.0.0        54.1.1.254               0    100      0 54 i
*> 115.0.0.0        204.12.1.254                           0 54 i
*>i116.0.0.0        54.1.1.254               0    100      0 54 i
*> 117.0.0.0        204.12.1.254                           0 54 i
*>i118.0.0.0        54.1.1.254               0    100      0 54 i
*> 119.0.0.0        204.12.1.254                           0 54 i
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.46 BGP Filtering with Extended Access-Lists

- Modify the filtering configuration in R4 as follows.
- Configure an extended access-list on R4 so that it does not accept any prefixes with even 3rd octet and with a subnet mask greater than /22 from BB3.
- This list should apply directly to the neighbor.

Configuration

Note

Extended access-lists add more functionality to BGP prefixes filtering. In addition to matching the subnet numbers they allows for subnet mask matching as well. A typical extended access-list entry in the format

```
permit {proto} <src-subnet> <src-mask> <dst-subnet> <dst-
mask> [options]
```

is treated as follows. First, the protocol field and other options are ignored. Next, **<src-subnet> <src-mask>** pair is used to build an expression for prefix subnet matching. The pair **<dst-subnet> <dst-mask>** is used as an expression to match prefixes subnet mask. For example the statement

```
permit ip 192.168.0.0 0.0.0.255 255.255.255.0 0.0.0.255
```

would match any prefix with the subnet number in range 192.168.0.0-192.168.0.255 AND having the prefix length of /24 or greater. It is possible to use more sophisticated constructs based on the wildcard bits logic, but this usually makes the configuration hard to read and interpret. Here are more examples:

permit ip 10.0.0.0 0.0.0.0 255.255.0.0 0.0.0.0 - matches 10.0.0.0/16 - Only

permit ip 10.0.0.0 0.0.0.0 255.255.255.0 0.0.0.0 - matches 10.0.0.0/24 - Only

permit ip 10.1.1.0 0.0.0.0 255.255.255.0 0.0.0.0 - matches 10.1.1.0/24 - Only

permit ip 10.0.0.0 0.0.255.0 255.255.255.0 0.0.0.0 - matches 10.0.X.0/24 - Any number in the 3rd octet of the network with a /24 subnet mask.

Copyright © 2010 Internetwork Expert

permit ip 10.0.0.0 0.255.255.0 255.255.255.0 0.0.0.0 - matches 10.X.X.0/24 - Any number in the 2nd & 3rd octet of the network with a /24 subnet mask.

permit ip 10.0.0.0 0.255.255.255 255.255.255.240 0.0.0.0 - matches 10.X.X.X/28 - Any number in the 2nd, 3rd & 4th octet of the network with a /28 subnet mask.

permit ip 10.0.0.0 0.255.255.255 255.255.255.0 0.0.0.255 - Matches 10.X.X.X/24 to 10.X.X.X/32 - Any number in the 2nd, 3rd & 4th octet of the network with a /24 to /32 subnet mask.

permit ip 10.0.0.0 0.255.255.255 255.255.255.128 0.0.0.127 - Matches 10.X.X.X/25 to 10.X.X.X/32 - Any number in the 2nd, 3rd & 4th octet of the network with a /25 to /32 subnet mask

In this scenario, we create a special entry that matches only the prefixes with the even 3rd octet AND have the mask length greater than or equal than 22. The second requirement is accomplished by translating the prefix length of 22 into binary and then into the decimal form: 255.255.252.0. Now we construct the wildcard mask that permits the remaining bit to take any value and end up with 255.255.252.0 0.0.3.255.

```
R4:
no ip access-list extended EVEN_3RD_MASK_GT_22
ip access-list extended EVEN_3RD_MASK_GT_22
 deny ip 0.0.0.0 255.255.254.255 255.255.252.0 0.0.3.255
 permit ip any any
!
no route-map FROM_BB3
!
router bgp 100
 no neighbor 204.12.1.254 route-map FROM_BB3 in
 neighbor 204.12.1.254 distribute-list EVEN_3RD_MASK_GT_22 in
```

Copyright © 2010 Internetwork Expert

Verification

Note

Check the BGP table in R4. Notice that the prefix 28.119.16.0/24 is being received from R6. This is the only prefix matching the access-list (3rd octet even and prefix-length of 24).

```
Rack1R4#show ip bgp regexp ^54
BGP table version is 44, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   54.1.1.254               0    100      0 54 i
*> 28.119.17.0/24   204.12.1.254             0             0 54 i
*> 112.0.0.0        204.12.1.254                           0 54 50 60 i
*> 113.0.0.0        204.12.1.254                           0 54 50 60 i
*> 114.0.0.0        204.12.1.254                           0 54 i
*> 115.0.0.0        204.12.1.254                           0 54 i
*> 116.0.0.0        204.12.1.254                           0 54 i
*> 117.0.0.0        204.12.1.254                           0 54 i
*> 118.0.0.0        204.12.1.254                           0 54 i
*> 119.0.0.0        204.12.1.254                           0 54 i
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.47 BGP Regular Expressions

- Create a new Loopback 1 on each device R1 – R6 with IP addresses in the format Y.Y.Y.Y/32, where Y is your device number, and advertise them into BGP.
- Configure an AS-Path access-list on SW1 so that AS 300 cannot be used as transit for AS 100 to reach AS 200 or vice-versa; this access-list should be applied directly to its neighbors.
- Configure a local-preference modification on R5 such that traffic from AS 200 going to routes originated in AS 54 is always sent to R4, while traffic to routes that transit AS 54 but were not originated in AS 54 is always sent to R3.
- Additionally configure R3 so that routes learned from AS 254 are not advertised to R1

Configuration

Note

Filtering based on AS_PATH attribute is done using BGP regular expressions. Regular expressions are matched against the AS_PATH strings. As you remember, AS_PATH could be constructed of the following elements: AS_SET (unordered list of AS numbers), AS_SEQUENCE (ordered list of AS numbers), AS_CONFED_SET and AS_CONFED_SEQUENCE which are the same elements but consist of the confederation AS numbers. For the purpose of matching, the AS_PATH attribute is viewed as a string starting with the adjacent AS number on the leftmost position, and the originating AS number in the rightmost position. When matching the AS_SET attribute, enclose the AS numbers in curly brackets and separate them by commas, e.g. {100,200,300}. When matching a confederation path, enclose the AS numbers in parenthesis, using backslash to escape the special meaning of the character: "\(100\)".

We are going to discuss the most useful types of regexp patterns suitable for many "real-life" situations. You may read more about BGP regular expressions basics in our blog post Understanding BGP Regular Expressions. First, recall the basic regular expression meta-characters or modifiers: "." – any character, "?" – repeat the previous character one or zero times, "*" – repeat the previous character zero or any times, "+" – repeat the previous character one or more times, "^" – match the beginning of a string, "$" match the end of a string, "[]" means range or elements and "_" is the character to match the "space" separating AS numbers OR the end of the AS_PATH list.

Copyright © 2010 Internetwork Expert

Other important regexp features include grouping and back-referencing. You can use parentheses to group AS numbers, e.g. (123 124 1+) and every group is assigned a number starting from left to right. For example in the string “1 2 (3 4) 5 6 (7 8)” the first group is assigned number “1” and the second group number “2”. You can later “recall” the grouping using the commands \1, \2 and so on for the group numbers. For example the string “(1 2) 3 \1” would match “1 2 3 1 2”. You may use the pipe character “|” in addition to the grouping characters for the concept of alternation. For example (1 2)|(5 6) would match “1 2” or “5 6”.

Now the practical examples:

“^$” - means empty AS_PATH attribute, which identifies the prefixes advertised in the local AS.

“^254_” - means prefixes received from the directly adjacent AS 254. Notice that using “_” is important, as there could be another adjacent AS with the number starting with 254.

“_254_” - prefixes transiting AS 254. The “_” characters are needed to clearly separate the AS number.

“_254$” - means prefixes originated in the AS 254. This expression matches the rightmost position in the string, meaning that the expression could be of arbitrary length.

“^([0-9]+)_254” - routes from the AS 254 when it’s just “one-hop” away.

“^254_([0-9]+)” - prefixes from the clients of the directly connected AS 254.

“^(254_)+([0-9]+)” - prefixes from the clients of the adjacent AS 254, accounting for the fact that AS 254 may do AS_PATH prepending.

“^254_([0-9]+_)+” - prefixes from the clients of the adjacent AS 254, accounting for the fact that the clients may do AS_PATH prepending.

^\(65100\) – prefixes learned from the confederation peer 65100.

Copyright © 2010 Internetwork Expert

You configure BGP regular-expression using the IP AS-PATH access-lists: **ip as-path access-list <N> {permit|deny} <Regexp>.** This access-list might be applied as a filter-list to a peer using the syntax: **neighbor <IP> filter-list <N> {in|out}**. However, the best approach is to match AS_PATH access-lists under a route-map applied to the peer (**match as-path**), as this allows for flexible policy editing. If you are wondering about the order features are applied, it is as follows:

For inbound updates:

1. route-map
2. filter-list
3. prefix-list OR distribute-list

For outbound updates:

1. prefix-list OR distribute-list
2. filter-list
3. route-map

Keep in mind that you may test regular expression on the BGP table using the command **show ip bgp regexp** or **show ip bgp quote-regexp**. The latter command allows using the "|" character to additionally filter the output.

R1:

```
interface Loopback 1
 ip address 1.1.1.1 255.255.255.0
!
router bgp 100
 network 1.1.1.0 mask 255.255.255.0
```

R2:

```
interface Loopback 1
 ip address 2.2.2.2 255.255.255.0
!
router bgp 200
 network 2.2.2.0 mask 255.255.255.0
```

Copyright © 2010 Internetwork Expert

R3:

```
interface Loopback 1
 ip address 3.3.3.3 255.255.255.0
!
no ip as-path access-list 1
ip as-path access-list 1 deny _54$
ip as-path access-list 1 permit _54_
!
ip as-path access-list 2 permit _254$
!
route-map FROM_R1 permit 10
 match as-path 1
 set local-preference 200
!
route-map FROM_R1 permit 100

!
!
!

route-map TO_R1 deny 10
 match as-path 2
!
route-map TO_R1 permit 100

!
!
!

router bgp 200
 network 3.3.3.0 mask 255.255.255.0
 neighbor 155.1.13.1 route-map FROM_R1 in
 neighbor 155.1.13.1 route-map TO_R1 out
```

R4:

```
interface Loopback 1
 ip address 4.4.4.4 255.255.255.0
!
router bgp 100
 network 4.4.4.0 mask 255.255.255.0
```

Copyright © 2010 Internetwork Expert

R5:
```
interface Loopback 1
 ip address 5.5.5.5 255.255.255.0
!
ip as-path access-list 1 permit _54$
!
route-map FROM_R4 permit 10
 match as-path 1
 set local-preference 200
!
route-map FROM_R4 permit 100
!
router bgp 200
 network 5.5.5.0 mask 255.255.255.0
 neighbor 155.1.45.4 route-map FROM_R4 in
```

R6:
```
interface Loopback 1
 ip address 6.6.6.6 255.255.255.0
!
router bgp 100
 network 6.6.6.0 mask 255.255.255.0
```

SW1:
```
ip as-path access-list 1 permit ^$
!
route-map NO_TRANSIT permit 100
 match as-path 1
!
router bgp 300
 neighbor 155.1.67.6 route-map NO_TRANSIT out
 neighbor 155.1.37.3 route-map NO_TRANSIT out
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Look at SW1's BGP table. Even though there are prefixes learned from AS 100 they are not being advertised to

```
Rack1SW1#show ip bgp regexp ^100
BGP table version is 28, local router ID is 150.1.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 1.1.1.0/24       155.1.67.6                             0 100 i
*  2.2.2.0/24       155.1.67.6                             0 100 200 i
*  3.3.3.0/24       155.1.67.6                             0 100 200 i
*> 4.4.4.0/24       155.1.67.6                             0 100 i
*  5.5.5.0/24       155.1.67.6                             0 100 200 i
*> 6.6.6.0/24       155.1.67.6               0             0 100 i
*> 28.119.16.0/24   155.1.67.6                             0 100 54 i
*> 28.119.17.0/24   155.1.67.6                             0 100 54 i
*> 112.0.0.0        155.1.67.6                             0 100 54 50
60 i
*> 113.0.0.0        155.1.67.6                             0 100 54 50
60 i
*> 114.0.0.0        155.1.67.6                             0 100 54 i
*> 115.0.0.0        155.1.67.6                             0 100 54 i
*> 116.0.0.0        155.1.67.6                             0 100 54 i
*> 117.0.0.0        155.1.67.6                             0 100 54 i
*> 118.0.0.0        155.1.67.6                             0 100 54 i
*> 119.0.0.0        155.1.67.6                             0 100 54 i
*> 155.1.0.0        155.1.67.6               0             0 100 i
*  205.90.31.0      155.1.67.6                             0 100 200
254 ?
*  220.20.3.0       155.1.67.6                             0 100 200
254 ?
*  222.22.2.0       155.1.67.6                             0 100 200
254 ?
```

```
Rack1SW1#show ip bgp neighbors 155.1.67.6 advertised-routes

Total number of prefixes 0
```

Copyright © 2010 Internetwork Expert

Note

Check R2's BGP table. Notice that prefixes transit across AS 54 have NEXT_HOP pointing to R1. At the same time, AS 54 originated prefixes have their NEXT_HOP pointing to R4.

```
Rack1R2#show ip bgp quote-regexp _54_
BGP table version is 114, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   155.1.45.4               0    200      0 100 54 i
* i                 155.1.45.4               0    200      0 100 54 i
*>i28.119.17.0/24   155.1.45.4               0    200      0 100 54 i
* i                 155.1.45.4               0    200      0 100 54 i
* i112.0.0.0        155.1.13.1               0    200      0 100 54 50 60 i
*>i                 155.1.13.1               0    200      0 100 54 50 60 i
* i113.0.0.0        155.1.13.1               0    200      0 100 54 50 60 i
*>i                 155.1.13.1               0    200      0 100 54 50 60 i
*>i114.0.0.0        155.1.45.4               0    200      0 100 54 i
* i                 155.1.45.4               0    200      0 100 54 i
*>i115.0.0.0        155.1.45.4               0    200      0 100 54 i
* i                 155.1.45.4               0    200      0 100 54 i
*>i116.0.0.0        155.1.45.4               0    200      0 100 54 i
* i                 155.1.45.4               0    200      0 100 54 i
*>i117.0.0.0        155.1.45.4               0    200      0 100 54 i
* i                 155.1.45.4               0    200      0 100 54 i
*>i118.0.0.0        155.1.45.4               0    200      0 100 54 i
* i                 155.1.45.4               0    200      0 100 54 i
*>i119.0.0.0        155.1.45.4               0    200      0 100 54 i
* i                 155.1.45.4               0    200      0 100 54 i
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.48 BGP Filtering with Maximum Prefix

- Configure R6 so that the peering session to BB1 is torn down if R6 learns more than 20 BGP prefixes from that neighbor.
- Once 16 prefixes are received from BB1, R6 should begin generating warning messages.
- Once down the peering should attempt to be restarted after three minutes.
- Configure SW1 so that if more than 20 prefixes are learned from R3 warning messages should be generated, but the peering session should NOT be terminated.

Configuration

Note

Filtering based on maximum-prefix number is an important BGP security feature. The number of BGP prefixes in the Internet is many hundred of thousands. It's is possible to overwhelm a BGP speaker's table by injecting too many prefixes and putting too much stress on router's CPU or memory. Cisco IOS supports special feature that limits the maximum number of prefix received over a BGP session. The command is **neighbor <IP> maximum-prefix <Number> [<Threshold%>] [warning-only]|[restart <minutes>]**.

The router by default permanently shuts down the session that exceeds the maximum-prefix limit specified by the <Number> parameter. The <Threshold%> value specifies the percent value applied to <Number> to generate a warning syslog message. The default threshold is 75%. For example if the prefix limit is 1000 then warning message is generated once 750 prefixes have been learned from this neighbor.

If you don't want the session to be shut-down once the maximum prefix number has reached, you can specify the **warning-only** keyword. This instruct the router to generate two warning messages: once for 75%*<Maximum> number of prefixes and once the <Maximum> has been crossed. Another option is using the **restart <minutes>** parameter. With this option set, the router will tear down the session once it reaches the maximum threshold, but will restore it after the number of **<minutes>** has passed.

R6:
```
router bgp 100
 neighbor 54.1.1.254 maximum-prefix 20 80 restart 3
```

SW1:
```
router bgp 300
 neighbor 155.1.37.3 maximum-prefix 20 warning-only
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> You may get the following message on SW1 due to excessive number of prefixes received. Also, the commands below will display the current prefix-limit settings.

```
%BGP-4-MAXPFX: No. of prefix received from 155.1.37.3 (afi 0) reaches
20, max 20

Rack1SW1#show ip bgp neighbors 155.1.37.3 | include Maximum|Thresh
  Maximum prefixes allowed 20 (warning-only)
  Threshold for warning message 75%

Rack1R6#show ip bgp neighbors 54.1.1.254 | include Maximum|rest
  Maximum prefixes allowed 20
  Threshold for warning message 80%, restart interval 3 min
```

Copyright © 2010 Internetwork Expert

Note

Load the the *Basic BGP Routing* initial configurations prior to continuing.

7.49 BGP Default Routing

- Configure R2 to originate a default route to R3 and R5 via BGP.
- This default route should be withdrawn if R2's link to BB2 goes down.

Configuration

Note

BGP default routing could be helpful when a stub AS uses just one uplink or uses primary-backup uplinks scenario. The upstream eBGP peers could be configure to advertise just the default route information. You may inject a default route into BGP by using the **`network 0.0.0.0 mask 0.0.0.0`** command, provided that there is a default route in the RIB. However, this will advertise the default route to all BGP neighbors. To selectively generate a default route, use the command **`neighbor <IP> default-originate [route-map <CONDITION>]`**. Without the route-map parameter, this command will generate a default route and send it to the configured peer. It is not required to have a matching default route in the BGP table or the RIB.

You can make the default-route advertisement conditional by associating a route-map with the statement. In this case, the default route is advertised if the route-map match conditions are satisfied. You can match ip addresses with the route-map using either standard, extended access-list or prefix-lists. When using extended ACLs, you can match both the prefix and the subnet mask range in the same way you specify that for BGP filtering. In our scenario, we are required to advertise the default route only if the interface connecting R2 to BB2 is up. This is accomplished by matching the prefix corresponding to the interface subnet 192.10.X.0/24. For more advanced scenarios, you may create reliable static routes, tracking the next hop reachability and matching the reliable prefix within the conditional route-map.

```
R2:
ip prefix-list LINK_TO_BB2 permit 192.10.1.0/24
!
route-map DEFAULT permit 10
 match ip address prefix-list LINK_TO_BB2
!
router bgp 200
 neighbor 155.1.23.3 default-originate route-map DEFAULT
 neighbor 155.1.0.5 default-originate route-map DEFAULT
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Make sure both R3 and R5 receive the default route from R2

```
Rack1R3#show ip bgp 0.0.0.0
BGP routing table entry for 0.0.0.0/0, version 30
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2           3
  Local
    155.1.0.2 from 155.1.0.5 (150.1.5.5)
      Origin IGP, metric 0, localpref 100, valid, internal
      Originator: 150.1.2.2, Cluster list: 150.1.5.5
  Local, (Received from a RR-client)
    155.1.23.2 from 155.1.23.2 (150.1.2.2)
      Origin IGP, metric 0, localpref 100, valid, internal, best

Rack1R5#show ip bgp 0.0.0.0
BGP routing table entry for 0.0.0.0/0, version 41
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2           3
  Local
    155.1.23.2 (metric 2681856) from 155.1.23.3 (150.1.3.3)
      Origin IGP, metric 0, localpref 100, valid, internal
      Originator: 150.1.2.2, Cluster list: 150.1.3.3
  Local, (Received from a RR-client)
    155.1.0.2 from 155.1.0.2 (150.1.2.2)
      Origin IGP, metric 0, localpref 100, valid, internal, best
```

> **Note**
>
> Now configure R2 to debug BGP updates for the prefix 0.0.0.0. Shut down the interface connected to BB2 and observe how R2 sends BGP WITHDRAW messages to R3 and R5.

```
Rack1R2(config)#access-list 99 permit 0.0.0.0
Rack1R2#debug ip bgp updates 99
BGP updates debugging is on for access list 99 for address family: IPv4
Unicast
Rack1R2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R2(config)#interface fastEthernet 0/0
Rack1R2(config-if)#shutdown
Rack1R2#
%BGP-5-ADJCHANGE: neighbor 192.10.1.254 Down Interface flap
BGP(0): 155.1.0.5 send unreachable 0.0.0.0/0
BGP(0): 155.1.0.5 enqueued default-originate update
BGP(0): 155.1.23.3 send unreachable 0.0.0.0/0
BGP(0): 155.1.23.3 enqueued default-originate update
```

Copyright © 2010 Internetwork Expert

7.50 BGP Local AS

- AS 100 is planning transition to the AS number 146. Configure R4 and R6 to use the new AS number while R1 should still use the old AS 100.
- Ensure all BGP peering relationships are still maintained but do not modify the configurations of any routersother than R1, R4 and R6.

Configuration

Note

The Hide Local Autonomous System feature could be useful when migrating an autonomous system to a different AS number. When the AS has multiple eBGP peering links, it may become time consuming to negotiate the AS number change with all peering partners. In this case, you may reconfigure the local BGP speakers to use the new AS number but advertise the old AS in BGP OPEN messages and BGP updates. This could be enforced on per-eBGP peer basis using the command **`neighbor <IP> local-as <OldAS> [no-prepend]`**.

The **`local-as <OldAS>`** command instructs the local router to advertise the **`<OldAS>`** number in BGP OPEN messages instead of the AS number specified with **`router bgp <NewAS>`** command. In addition to that, all BGP prefixes advertised to this eBGP peer would have the AS numbers <OldAS> <NewAS> preprended in front of every BGP update's AS_PATH attribute. Thus, the external system may continue with the local system using the old AS number. In addition to that, the external system will see the updates coming from the <OldAS> looking like they first transited <NewAS>. This is needed to avoid BGP routing loops.

If you specify the **`no-prepend`** keyword, then any routes *received* from the eBGP peer will not have <OldAS> prepended upon reception. By default the AS number specified with the **`local-as`** command (<OldAS>) is prepended to all updates received, to avoid potential routing loops. However, this may cause problems with partial transitions, when part of your AS is using the new AS number, and another part is still using the old AS number. The routers using the old number will reject such updates due to the same AS number present in AS_PATH.

In our scenario, only R4 and R6 have been reconfigured to use the new AS number 146. R1 is still using AS 100 and have been reconfigured to peer eBGP with R4 and R6. In order to make R1 accept AS 54 prefixes, we use the **`no-prepend`** keyword when peering using the local-as feature with BB1 and BB3.

Copyright © 2010 Internetwork Expert

```
R1:
router bgp 100
 no neighbor 155.1.146.4 route-reflector-client
 no neighbor 155.1.146.6 route-reflector-client
 neighbor 155.1.146.4 remote-as 146
 neighbor 155.1.146.6 remote-as 146
 neighbor 155.1.13.3 remote-as 200

R4:
no router bgp 100
router bgp 146
 neighbor 155.1.146.1 remote-as 100
 neighbor 204.12.1.254 remote-as 54
 neighbor 204.12.1.254 local-as 100 no-prepend
 neighbor 155.1.45.5 remote-as 200
 neighbor 155.1.45.5 local-as 100 no-prepend
 network 155.1.146.0 mask 255.255.255.0
 aggregate-address 155.1.0.0 255.255.0.0 summary-only

R6:
no router bgp 100
router bgp 146
 neighbor 155.1.146.1 remote-as 100
 neighbor 54.1.1.254 remote-as 54
 neighbor 54.1.1.254 local-as 100 no-prepend
 neighbor 155.1.67.7 remote-as 300
 neighbor 155.1.67.7 local-as 100 no-prepend
 network 155.1.146.0 mask 255.255.255.0
 aggregate-address 155.1.0.0 255.255.0.0 summary-only
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> First, look how the Local-AS feature show's up in the respective command's output. Next, check R1's BGP table and notice that routes learned from AS 54 appear like they transited through AS 146.

```
Rack1R6#show ip bgp neighbors 155.1.67.7 | include local
BGP neighbor is 155.1.67.7,  remote AS 300,  local AS 100 no-prepend,
external link

Rack1R1#show ip bgp regexp _54$
BGP table version is 42, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  28.119.16.0/24   155.1.146.6                            0 146 54 i
*>                  155.1.146.4                            0 146 54 i
*  28.119.17.0/24   155.1.146.6                            0 146 54 i
*>                  155.1.146.4                            0 146 54 i
*  114.0.0.0        155.1.146.6                            0 146 54 i
*>                  155.1.146.4                            0 146 54 i
*  115.0.0.0        155.1.146.6                            0 146 54 i
*>                  155.1.146.4                            0 146 54 i
*  116.0.0.0        155.1.146.6                            0 146 54 i
*>                  155.1.146.4                            0 146 54 i
*  117.0.0.0        155.1.146.6                            0 146 54 i
*>                  155.1.146.4                            0 146 54 i
*  118.0.0.0        155.1.146.6                            0 146 54 i
*>                  155.1.146.4                            0 146 54 i
*  119.0.0.0        155.1.146.6                            0 146 54 i
*>                  155.1.146.4                            0 146 54 i
```

Copyright © 2010 Internetwork Expert

Note

Prefixes advertised via eBGP to SW1 (**local-as no-prepend** peer) have AS_PATH prepended with "100 146". Keep in mind that **no-prepend** feature applies only to inbound learned routes. All externally advertise routes still have the local-as number prepended.

```
Rack1SW1#show ip bgp regexp 146_54$
BGP table version is 139, local router ID is 150.1.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  28.119.16.0/24   155.1.67.6                             0 100 146 54
i
*  28.119.17.0/24   155.1.67.6                             0 100 146 54
i
*  114.0.0.0        155.1.67.6                             0 100 146 54
i
*  115.0.0.0        155.1.67.6                             0 100 146 54
i
*  116.0.0.0        155.1.67.6                             0 100 146 54
i
*  117.0.0.0        155.1.67.6                             0 100 146 54
i
*  118.0.0.0        155.1.67.6                             0 100 146 54
i
*  119.0.0.0        155.1.67.6                             0 100 146 54
i
```

Note

Now, disable to "no-prepend" feature in R6 and check the BGP routes learned from AS 54. Notice that now they have the AS number 100 prepended in front of their AS_PATH attribute.

```
R6:
router bgp 146
 neighbor 54.1.1.254 local-as 100
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip bgp regexp _54$
BGP table version is 34, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   54.1.1.254                             0 100 54 i
*> 28.119.17.0/24   54.1.1.254                             0 100 54 i
*> 114.0.0.0        54.1.1.254               0             0 100 54 i
*> 115.0.0.0        54.1.1.254               0             0 100 54 i
*> 116.0.0.0        54.1.1.254               0             0 100 54 i
*> 117.0.0.0        54.1.1.254               0             0 100 54 i
*> 118.0.0.0        54.1.1.254               0             0 100 54 i
*> 119.0.0.0        54.1.1.254               0             0 100 54 i
```

Note

The AS 100 attribute in the AS_PATH prevents the AS 54 originated routes learned via R4 from being accepted by R1. R1 shows only the prefixes learned via R4.

```
Rack1R1#show ip bgp regexp _54$
BGP table version is 42, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   155.1.146.4                            0 146 54 i
*> 28.119.17.0/24   155.1.146.4                            0 146 54 i
*> 114.0.0.0        155.1.146.4                            0 146 54 i
*> 115.0.0.0        155.1.146.4                            0 146 54 i
*> 116.0.0.0        155.1.146.4                            0 146 54 i
*> 117.0.0.0        155.1.146.4                            0 146 54 i
*> 118.0.0.0        155.1.146.4                            0 146 54 i
*> 119.0.0.0        155.1.146.4                            0 146 54 i
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.51 BGP Local AS Replace-AS/Dual-AS

- Re-configure R1 to reside in AS 146 as well and act as a routing reflector for R4 and R6.
- All external Autonomous Systems should be unaware of the new AS numbers used by these routers.
- R5 should peer with R4 using the AS number 146.

Configuration

Note

As you remember, when configuring the hide local AS feature, the external peers would see both the local-AS and the real AS number prepended in front of the AS_PATH. Sometimes, it is desirable to completely hide the "real" AS number (the one configured via **`router bgp <RealAS>`** command). To accomplish this, use the **`no-prepend replace-as`** parameters to the **`local-as`** command. This combination will replace the real AS number with the one specified in the **`local-as`** command. The respective neighbor will be completely tricked into thinking that all routers are received from the AS number configured with the **`local-as`** command, as this number will appear in the AS_PATH and BGP OPEN message. Keep in mind that such replacement could lead to routing loops, if the original AS was partitioned using two AS numbers.

With the replace-AS feature is configured, it may happen that the external peer is configured to peer using the real AS number, for example the AS number that would be used *after* migration. In this case, it is possible to configure the "hiding" peer to initiate/accept BGP sessions using both AS numbers (the real number and the local number). The external peer will accept the correct number and negotiate the BGP session. The "hiding" peer will then use the negotiated AS number to prepend the updates sent to the external peer.

```
R1:
no router bgp 100
router bgp 146
 neighbor 155.1.146.4 remote-as 146
 neighbor 155.1.146.6 remote-as 146
 neighbor 155.1.146.4 route-reflector-client
 neighbor 155.1.146.6 route-reflector-client
 neighbor 155.1.13.3 remote-as 200
 neighbor 155.1.13.3 local-as 100 no-prepend replace-as

R4:
router bgp 146
 neighbor 155.1.146.1 remote-as 146
 neighbor 155.1.45.5 local-as 100 no-prepend replace-as dual-as
 neighbor 204.12.1.254 local-as 100 no-prepend replace-as
```

Copyright © 2010 Internetwork Expert

```
R5:
router bgp 200
 neighbor 155.1.45.4 remote-as 146

R6:
router bgp 146
 neighbor 155.1.146.1 remote-as 146
 neighbor 155.1.67.7 local-as 100 no-prepend replace-as
 neighbor 54.1.1.254 local-as 100 no-prepend replace-as
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Look at R3's and SW1's prefixes received from AS 146. Notice that only AS 100 is prepended to those prefixes by AS 146 speakers, even though the real AS is 146.

```
Rack1R3#show ip bgp neighbors 155.1.13.1 routes
BGP table version is 115, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   155.1.13.1                             0 100 54 i
*> 28.119.17.0/24   155.1.13.1                             0 100 54 i
*> 112.0.0.0        155.1.13.1                             0 100 54 50
60 i
*> 113.0.0.0        155.1.13.1                             0 100 54 50
60 i
*> 114.0.0.0        155.1.13.1                             0 100 54 i
*> 115.0.0.0        155.1.13.1                             0 100 54 i
*> 116.0.0.0        155.1.13.1                             0 100 54 i
*> 117.0.0.0        155.1.13.1                             0 100 54 i
*> 118.0.0.0        155.1.13.1                             0 100 54 i
*> 119.0.0.0        155.1.13.1                             0 100 54 i
*> 155.1.0.0        155.1.13.1                             0 100 i

Total number of prefixes 11
```

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show ip bgp neighbors 155.1.67.6 routes
BGP table version is 150, local router ID is 150.1.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   155.1.67.6                             0 100 54 i
*> 28.119.17.0/24   155.1.67.6                             0 100 54 i
*> 112.0.0.0        155.1.67.6                             0 100 54 50
60 i
*> 113.0.0.0        155.1.67.6                             0 100 54 50
60 i
*> 114.0.0.0        155.1.67.6                             0 100 54 i
*> 115.0.0.0        155.1.67.6                             0 100 54 i
*> 116.0.0.0        155.1.67.6                             0 100 54 i
*> 117.0.0.0        155.1.67.6                             0 100 54 i
*> 118.0.0.0        155.1.67.6                             0 100 54 i
*> 119.0.0.0        155.1.67.6                             0 100 54 i
*> 155.1.0.0        155.1.67.6                  0          0 100 i

Total number of prefixes 11
```

Note

At the same time, R5 peers with R4 via the Dual-AS feature. All prefixes received from R4 have AS_PATH prepended with AS 146, not AS 100.

```
Rack1R5#show ip bgp neighbors 155.1.45.4 routes
BGP table version is 137, local router ID is 150.1.5.5
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   155.1.45.4                             0 146 54 i
*> 28.119.17.0/24   155.1.45.4                             0 146 54 i
*> 112.0.0.0        155.1.45.4                             0 146 54 50
60 i
*> 113.0.0.0        155.1.45.4                             0 146 54 50
60 i
*> 114.0.0.0        155.1.45.4                             0 146 54 i
*> 115.0.0.0        155.1.45.4                             0 146 54 i
*> 116.0.0.0        155.1.45.4                             0 146 54 i
*> 117.0.0.0        155.1.45.4                             0 146 54 i
*> 118.0.0.0        155.1.45.4                             0 146 54 i
*> 119.0.0.0        155.1.45.4                             0 146 54 i
*> 155.1.0.0        155.1.45.4                  0          0 146 i

Total number of prefixes 11
```

Copyright © 2010 Internetwork Expert

Note

Now reconfigure R5 for peering using the AS number 100. Check the routes received from R4 after this. Notice that now AS_PATH is prepended with the AS number 100, not 146.

```
Rack1R5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#router bgp 200
Rack1R5(config-router)#neighbor 155.1.45.4 remote-as 100

Rack1R5#show ip bgp neighbors 155.1.45.4 routes
BGP table version is 159, local router ID is 150.1.5.5
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   155.1.45.4                             0 100 54 i
*> 28.119.17.0/24   155.1.45.4                             0 100 54 i
*> 112.0.0.0        155.1.45.4                             0 100 54 50
60 i
*> 113.0.0.0        155.1.45.4                             0 100 54 50
60 i
*> 114.0.0.0        155.1.45.4                             0 100 54 i
*> 115.0.0.0        155.1.45.4                             0 100 54 i
*> 116.0.0.0        155.1.45.4                             0 100 54 i
*> 117.0.0.0        155.1.45.4                             0 100 54 i
*> 118.0.0.0        155.1.45.4                             0 100 54 i
*> 119.0.0.0        155.1.45.4                             0 100 54 i
*> 155.1.0.0        155.1.45.4                    0        0 100 i

Total number of prefixes 11
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.52 BGP Remove Private AS

- Reconfigure SW1 and SW3 in the private AS 65089 and adjust the peering settings accordingly.
- Create and advertise Loopback 1 subnet in SW1 with the IP address 7.7.7.7/24.
- Configure AS 100 and AS 200 speakers to strip the private AS number when advertising the prefixes to AS 254 and AS 54.

Configuration

Note

Private AS numbers in range 64512-65535 are often assigned to small enterprises that use BGP to peer with their ISPs. Private AS numbers are similar to RFC 1918 IP addressing, which allows for consuming AS numbers on the Internet. However, private AS numbers should not appear on the public Internet, as many sites may originate the same number. Thus, the AS that provides upstream connection for the private site should remove the private AS numbers from the AS_PATH attribute.

The command to perform the AS_PATH stripping in IOS **`is neighbor <IP> remove-private-as`**. All BGP updates sent over this session are inspected to have a sequence of private AS numbers in the beginning of the AS_PATH. All private numbers are then removed and the local AS number is prepended. Notice that in situation when the private AS sequence is not located in the beginning of the AS_PATH, the stripping will not work and AS_PATH will remain unmodified.

```
R2:
router bgp 200
 neighbor 192.10.1.254 remove-private-as

R3:
router bgp 200
 neighbor 155.1.37.7 remote-as 65089

R4:
router bgp 146
 neighbor 204.12.1.254 remove-private-as

R6:
!
! BGP AS was modified in the previous task
!
router bgp 146
 neighbor 155.1.67.7 remote-as 65089
 neighbor 54.1.1.254 remove-private-as
```

Copyright © 2010 Internetwork Expert

```
SW1:
no router bgp 300
router bgp 65089
 neighbor 155.1.79.9 remote-as 65089
 neighbor 155.1.67.6 remote-as 100
 neighbor 155.1.37.3 remote-as 200
 network 7.7.7.0 mask 255.255.255.0
!
interface Loopback1
 ip address 7.7.7.7 255.255.255.0

SW3:
no router bgp 300
router bgp 65089
 neighbor 155.1.79.7 remote-as 65089
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> First, check the paths advertised to BB3 on R4. Notice the AS_PATH attribute for the prefix 7.7.7.0/24. This is the output of Adj-RIBs-Out, and the result of the private AS removal and local AS prepending is not yet show.

```
Rack1R4#show ip bgp neighbors 204.12.1.254 advertised-routes
BGP table version is 22, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i7.7.7.0/24       155.1.67.7               0    100      0 65089 i
*> 155.1.0.0        0.0.0.0                            32768 i
*> 205.90.31.0      155.1.45.5                             0 200 254 ?
*> 220.20.3.0       155.1.45.5                             0 200 254 ?
*> 222.22.2.0       155.1.45.5                             0 200 254 ?
```

> **Note**
>
> Now check BB3's BGP table. Notice that the prefixy 7.7.7.0/24 appear with the AS_PATH of 100, i.e. the private number has been removed. Keep in mind that in the real exam you wont have access to the backbone routers.

```
RS.30.1.BB3>show ip bgp
BGP table version is 16, local router ID is 31.3.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i7.7.7.0/24       172.16.4.1               0    100      0 100 i
*>                  204.12.1.4                             0 100 i
*> 28.119.16.0/24   0.0.0.0                  0         32768 i
*> 28.119.17.0/24   0.0.0.0                  0         32768 i
*>i112.0.0.0        172.16.4.1               0    100      0 i
```

Copyright © 2010 Internetwork Expert

Note

Now shut down the BGP peering session between R6 and SW1. This will make AS 100 (146) accept the prefix from AS 200 with the AS_PATH "200 65089". Notice that in real life AS 200 should have removed the private AS when advertising the prefix to AS 100. However, we intentionally left this misconfiguration.

```
R6:
neighbor 155.1.67.7 shutdown

Rack1R4#show ip bgp neighbors 204.12.1.254 advertised-routes
BGP table version is 24, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 7.7.7.0/24       155.1.45.5                             0 200 65089
i
*> 155.1.0.0        0.0.0.0                            32768 i
*> 205.90.31.0      155.1.45.5                             0 200 254 ?
*> 220.20.3.0       155.1.45.5                             0 200 254 ?
*> 222.22.2.0       155.1.45.5                             0 200 254 ?

Total number of prefixes 5
```

Note

Now check BB3's BGP table and notice that the private AS has not been stripped, as it wasn't the first AS number in the AS_PATH.

```
RS.30.1.BB3>show ip bgp
BGP table version is 18, local router ID is 31.3.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i7.7.7.0/24       172.16.4.1               0    100      0 100 200
65089 i
*>                  204.12.1.4                             0 100 200
65089 i
```

Copyright © 2010 Internetwork Expert

7.53 BGP Dampening

- Create a Looback 1 interface in R1 with the IP address 1.1.1.1/24 and advertise it into BGP.
- Configure AS 200 routers to suppress advertisement of oscillating networks.
- Once a prefix flaps two times in a row, the advertisement should resume in 5 minutes.

Configuration

Note

Network instabilities (e.g. unreliable links) may cause BGP prefix flapping. A flap is the even of a network prefix going up and down or vice-versa, i.e. flap is change in reachability state. Prefix flapping is dangerous to network stability, as it causes network withdraws and best-path re-computations. Moreover, a flapping prefix may cause recursive route withdrawn, resulting in massive BGP table changes. Two main methods to reduce the impact of network instabilities are summarization (information hiding) and prefix dampening. Summarization aggregates reachability information and hides flaps of the specific prefixes constituting a summary. Dampening is the process of suppressing a flapping prefix advertisement till the moment it becomes "stable". This introduces some "inertial" mechanism to new prefix advertisement, delaying the changes announcements for oscillating prefixes.

The idea of dampening is to suppress a prefix based on the number of flaps accounted and un-suppress the prefix only after an exponentially decaying timer expired. Here is the description of the process. Every time a prefix flaps, it is assigned an additive penalty value, 1000 by default. If this is just an attribute change, e.g. Local-Preference or AS_PATH then the penalty is halved, i.e. 500 by default. If the prefix becomes unavailable after flapping at least once, BGP process still keeps it in the table, marked in "history" state to account for further flaps and penalty accumulation. If the accumulated penalty value exceeds the *Suppress Limit,* which defaults to 2000, BGP process will mark the route as *damped*. Even though the prefix could be currently "active" (flapped from down to up), BGP will not advertise it to any peers. Every five seconds the BGP process exponentially decreases the penalty value assigned to the prefix. The exponential decay process has one parameter - *Half-Life* time period, which specifies the amount of time needed to decrease the current penalty to the value twice smaller. That is, the decay process follows the equation `P(t) = P(0)/2^(t/Half_Life)` with the default *Half_Life* time of 15 minutes. As soon as the penalty falls below the *Reuse Limit*, the router will unsuppress the route and start advertising it again. The decision to unsuppress a prefix is made every 10 seconds.

Copyright © 2010 Internetwork Expert

When facing tasks similar to the current scenario, you should understand that they assume some "ideal" conditions. That is, when the scenario says "flaps two times in a row" you may assume the flaps immediately once after another. This results in accumulated penalty of 2000 and route being damped. We now need to find the Half_Life value that will make the router reuse the prefix in 5 minutes. We take the penalty evolution equation and write it as follows:

```
P(5) = P(0)/2^(5/Half_Life)
```

And substitute P(5)=750 (the reuse limit) and P(0)=2000 (Suppress Limit). The equation then becomes:

```
200/750=2^(5/Half_Life)
```

From this equation we can find the Half_Life value by taking logarithm of the both sides:

`Half_Life=5*Ln(2)/Ln(200/75)=3.5` (approximately).

We may round the result up to 4 minutes. That is, the task could be accomplished by setting the Half_Life value to 4 minutes. Now, the BGP command to apply the dampening parameters is **`bgp dampening [<Half_Life> <ReuseLimit> <SuppressLimit> <MaximumSuppressTime>]`**. The last parameter, <MaximumSuppressTime>, specifies the time limit to keep the prefix damped if it keeps oscillating. The default value if 4xHalf_Life or 60 minutes. The router sets the maximum penalty value based on this timer using the formula **`Max_Penalty = ReuseLimit *2^(MaximumSuppressTime/Half_Life)`**. You may review the current dampening parameters (if enabled) using the command **`show ip bgp dampening parameters`**.

R2, R3, R5, SW2, SW4:

```
router bgp 200
 bgp dampening 4 750 2000 16
```

R1:

```
!
! We adjust the advertisement interval to minimize prefix batching
! and make R1 advertise prefix changes ASAP
!
router bgp 146
 network 1.1.1.0 mask 255.255.255.0
 neighbor 155.1.13.3 advertisement-interval 0
!
interface Loopback1
 ip address 1.1.1.1 255.255.255.0
```

Copyright © 2010 Internetwork Expert

Verification

Note

Start by checking the BGP dampening parameters in any of AS 200 routers. Next, go to R1 and shutdown/no shutdown Loopback1 interface a few times emulating route flaps, enough to accumulate the suppress-limit penalty in AS 200 routers.

```
Rack1R3#show ip bgp dampening parameters
 dampening 4 750 2000 16
  Half-life time       : 4  mins       Decay Time       : 620 secs
  Max suppress penalty: 12000          Max suppress time: 16 mins
  Suppress penalty     :  2000         Reuse penalty    : 750
```

Note

Inspect dampened path and flap statistics in R3. Notice the character "d" meaning the prefixes have been damped.

```
Rack1R3#show ip bgp dampening dampened-paths
BGP table version is 31, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          From             Reuse    Path
*d 1.1.1.0/24       155.1.13.1       00:09:20 100 i
 d 1.1.1.0/24       155.1.37.7       00:07:00 300 100 i

Rack1R3#show ip bgp dampening flap-statistics
BGP table version is 31, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          From            Flaps Duration Reuse    Path
*d 1.1.1.0/24       155.1.13.1      5     00:05:07 00:09:10 100
 d                  155.1.37.7      5     00:05:07 00:08:50 300 100
```

Copyright © 2010 Internetwork Expert

Note

Check R3's BGP table for the prefix 1.1.1.0/24. Notice that the prefix shows up as damped and not advertised to any peer. If you check R2's BGP table after this, you will notice that the prefix is not there.

```
Rack1R3#show ip bgp
BGP table version is 31, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*d 1.1.1.0/24       155.1.13.1               0             0 100 i
*d                  155.1.37.7                             0 300 100 i

Rack1R3#show ip bgp 1.1.1.0
BGP routing table entry for 1.1.1.0/24, version 31
Paths: (2 available, no best path)
Flag: 0x960
  Not advertised to any peer
  100, (suppressed due to dampening)
    155.1.13.1 from 155.1.13.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, external
      Dampinfo: penalty 3542, flapped 5 times in 00:05:18, reuse in
00:09:00
  300 100, (suppressed due to dampening) (history entry)
    155.1.37.7 from 155.1.37.7 (150.1.7.7)
      Origin IGP, localpref 100, external
      Dampinfo: penalty 3330, flapped 5 times in 00:05:18, reuse in
00:08:40

Rack1R2#show ip bgp 1.1.1.0
% Network not in table
```

7.54 BGP Dampening with Route-Map

- Ensure that dampening process applies only to AS 100 originated routes, and does not affect any other prefixes.

Configuration

> **Note**
>
> Sometimes it might be desirable to apply dampening only to a certain set of routes, e.g. to the prefixes originated from the "problematic" AS. Another good example might be a set of prefixes describing critical network resources that must always be available. To account for such situations it is possible to use a route map with the BGP dampening command to select prefixes eligible for dampening. The command **bgp dampening route-map <MAP_NAME>** where the route-map may match IP addresses using access-lists, prefix-lists and as-path lists. For every route-map entry you may set specific dampening parameters using the command **set dampening <Half_Life> <ReuseLimit> <SuppressLimit> <MaximumSuppressTime>**.
>
> For this task, we create an AS_PATH access-list matching the paths originated in AS 100 and set the dampening parameters using the Half-Life value of 4 minutes.

R2, R3, R5, SW2, SW4:

```
ip as-path access-list 100 permit _100$
!
route-map DAMPENING
 match as-path 100
 set dampening 4 750 2000 16
!
router bgp 200
 no bgp dampening
 bgp dampening route-map DAMPENING
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Configure R3 for BGP dampening debugging. Next, configure SW1 for minimal advertisement interval and do a number of consecutive shutdowns/no shutdowns for Loopback1 interface. After this, go to R1 and perform the same series of operations on Loopback1.

```
Rack1SW1(config)#router bgp 65089
Rack1SW1(config-router)#neighbor 155.1.37.3 advertisement-interval 0

Rack1R3#debug ip bgp dampening
BGP dampening debugging is on for address family: IPv4 Unicast
```

> **Note**
>
> Now check BGP dampening settings in R3. Notice that the dampening settings only apply to the prefixes matching the route map. Next, check the output for the debugging command activated above. Notice that the only prefix tracked by the dampening process is 1.1.1.0/24.

Copyright © 2010 Internetwork Expert

```
Rack1R3#show ip bgp dampening parameters
 dampening 4 750 2000 16 (route-map DAMPENING 10)
  Half-life time       : 4  mins       Decay Time       : 620 secs
  Max suppress penalty: 12000          Max suppress time: 16 mins
  Suppress penalty     :  2000         Reuse penalty    : 750

EvD: charge penalty 1000, new accum. penalty 1000, flap count 1
BGP(0): charge penalty for 1.1.1.0/24 path 65089 100 with halflife-time
4 reuse/suppress 750/2000
BGP(0): flapped 1 times since 00:00:00. New penalty is 1000
EvD: charge penalty 1000, new accum. penalty 1000, flap count 1
BGP(0): charge penalty for 1.1.1.0/24 path 100 with halflife-time 4
reuse/suppress 750/2000
BGP(0): flapped 1 times since 00:00:00. New penalty is 1000
EvD: accum. penalty decayed to 1000 after 2 second(s)
EvD: accum. penalty decayed to 1000 after 2 second(s)
EvD: accum. penalty 853, not suppressed
EvD: accum. penalty 853, not suppressed
EvD: accum. penalty decayed to 853 after 2 second(s)
EvD: accum. penalty decayed to 853 after 2 second(s)
EvD: accum. penalty decayed to 793 after 28 second(s)
EvD: charge penalty 1000, new accum. penalty 1793, flap count 2
BGP(0): charge penalty for 1.1.1.0/24 path 100 with halflife-time 4
reuse/suppress 750/2000
BGP(0): flapped 2 times since 00:01:30. New penalty is 1793
EvD: accum. penalty decayed to 793 after 29 second(s)
EvD: charge penalty 1000, new accum. penalty 1793, flap count 2
BGP(0): charge penalty for 1.1.1.0/24 path 65089 100 with halflife-time
4 reuse/suppress 750/2000
BGP(0): flapped 2 times since 00:01:31. New penalty is 1793
EvD: accum. penalty decayed to 1644 after 32 second(s)
EvD: accum. penalty decayed to 1644 after 31 second(s)
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Note

Load the the *Basic BGP Routing* initial configurations prior to continuing.

7.55 BGP Timers Tuning

- Configure R2's BGP process to process conditional route advertisement every 20 seconds.
- R2 should not batch routing updates to BB2 and advertise them immediately.
- Configure R2 so that session deactivation happens within 15 seconds of no session activity.

Configuration

Note

BGP routing process is complicated and uses many data structures and periodically scheduled tasks. One of the most important BGP processes is "BGP scanner" which performs the following important functions:

1) Runs through all prefixes in BGP table and check the validity and reachability of BGP NEXT_HOP attribute.
2) Performs conditional advertisement and route injection.
3) Imports new routes into BGP table from RIB (via network and redistribute commands).
4) Performs route dampening.

Later you will some that some of this periodic behavior became event-driven in recent IOS releases (BGP next-hop trigger feature). However, for now what's important is that BGP scanner runs every 60 seconds by default. You can change this interval using the command **`bgp scan-time <5-60>`**. The shorter is the interval, the better is routing convergence, but at the same time the more load is put on the router's CPU. You may check the BGP scanner runs using the command **`debug ip bgp events`**.

Copyright © 2010 Internetwork Expert

Another important BGP process is "BGP I/O", which processes BGP UPDATE and KEEPALIVE messages. By default, BGP batches all new prefixes and delays the sending of an update packet to the peer until the next advertisement-interval timer expires. This interval is configured on a per-peer basis using the command **neighbor <IP> advertisement-interval <seconds>**. Decreasing this interval (the minimum value is zero) improves BGP convergence but generates more BGP traffic and puts more stress on router's CPU.

The last important timer is the BGP session keepalive interval. Keepalives are important to validate BGP session health, to avoid routing blackholes. BGP peers advertise the hold time interval when establishing BGP peering session and send KEEPALIVE message to inform each other of their availability. Peers may advertise different hold-time interval – it's only important that the peer receive the KEEPALIVE message until its hold-time interval expires. You can change the keepalive and hold-time periods on per-process basis using the command **timers bgp <keepalive> <holdtime>**. The default values are 60 and 180 for keepalive and holdtime intervals respectively. Setting the keepalive interval too small results in faster peering deactivation detection, but may lead to "false positives" and disruption of BGP session by mistake and excessive route flapping. Keep in mind that you need to completely reset the BGP session in order for the new keepalive timers to take effect. If you want to observe the BGP keepalive exchange process, use the command **debug ip bgp keepalive**.

```
R2:
router bgp 200
 timers bgp 5 15
 neighbor 192.10.1.254 advertisement-interval 0
 bgp scan-time 20
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the timer values using show commands demonstrated below.

```
Rack1R2#show ip bgp summary | include scan
BGP activity 14/0 prefixes, 25/0 paths, scan interval 20 secs

Rack1R2#show ip bgp neighbors 192.10.1.254 | inc advertisement|keepal
  Last read 00:00:01, last write 00:00:01, hold time is 15, keepalive
interval is 5 seconds
  Configured hold time is 15,keepalive interval is 5 seconds, Minimum
holdtime from neighbor is 0 seconds
  Default minimum time between advertisement runs is 30 seconds
  Minimum time between advertisement runs is 0 seconds
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.56 BGP Fast Fallover

- Disable the BGP feature in R3 that allows for eBGP peering session deactivation when a physical interface goes down.
- Configure all R3's BGP peering session for fast peering deactivation.

Configuration

Note

It is common to have eBGP peers directly connected across a physical interface. With point-to-point links this allows for fast external session deactivation based on the interface protocol state. That is, as soon as the interface connecting to an external peer signals protocols down, the BGP process may deactivate the peering session without waiting for the hold-down time to expire. This feature is enabled by default for eBGP sessions using the BGP process command **`bgp fast-external-fallover`**. Notice that this feature is only efficient when the peering session is established across the non-shared link – using it on NBMA and Ethernet interfaces might be inefficient.

A more advanced version of this feature is available in recent IOS release. The new feature is called "fast peering session deactivation support" and could be configured on a per-neighbor basis using the command **`neighbor <IP> fall-over`**. This feature applies to both iBGP and eBGP (mostly multi-hop) sessions and does not depend on interface state. The IGP route used to reach the peer configured for fast session deactivation is monitored by the BGP process. Once the IGP route disappears, the BGP session gets immediately torn down, unless there is a backup IGP route to reach the peer. Thus, the new feature is event driven and allows for fast detection of peering issues even for iBGP (non-direct) sessions. Notice that this feature has the same limitation as the fast-external-fallover – if the peer is connected via a shared interface, the router may not detect the loss of the directly connected IGP network. In situations like this, you may use point-to-point NBMA subinterfaces or reliable static /32 routes on the shared NBMA or Ethernet interfaces.

Keep in mind that convergence improvements result in less stable topology. In order to minimize the impact of IGP instabilities on BGP tables, use event dampening technologies (such as ip dampening) and prefix summarization.

Copyright © 2010 Internetwork Expert

```
R3:
router bgp 200
 no bgp fast-external-fallover
 neighbor 155.1.0.5 fall-over
 neighbor 155.1.13.1 fall-over
 neighbor 155.1.23.2 fall-over
 neighbor 155.1.37.7 fall-over
 neighbor 155.1.58.8 fall-over
 neighbor 155.1.108.10 fall-over
```

Verification

Note

In order to test the fast fall-over feature, configure R4 and R3 to shutdown all BGP peering sessions with external ASes. This is needed because R4 and R6 generate a summary prefix 155.1.0.0/16 which is injected into RIB and could be used as a backup route to reach any BGP next-hop in AS 200.

```
R3:
router bgp 200
 neighbor 155.1.13.1 shutdown
 neighbor 155.1.37.7 shutdown

R4:
router bgp 200
 neighbor 155.1.45.4 shutdown
```

Copyright © 2010 Internetwork Expert

Note

Now check that R3's iBGP peering session is enabled for fast fall-over. After this, activate a BGP debugging command for RIB watching and shutdown the link connecting SW2 and SW4.

```
Rack1R3#show ip bgp neighbors 155.1.108.10 | include [Ff]all
 Fall over configured for session

Rack1R3#debug ip bgp rib-filter
BGP Rib filter debugging is on

Rack1SW2(config)#
Rack1SW2(config)#interface port-channel 1
Rack1SW2(config-if)#shutdown

BGP- ATF: EVENT 155.1.108.0/24 RIB update DOWN
BGP- ATF: T 155.1.108.10/32 (0) c=0x843C4198 Query pending
BGP- ATF: R 155.1.108.0/24 (0) -> Se1/0.1 155.1.0.5 Deleting
BGP- ATF: EVENT Query 155.1.108.0/24 (0) found route
BGP- ATF: 155.1.108.0/24 (0) Adding route
BGP- ATF: R 155.1.108.0/24 (0) -> Updating route
BGP- ATF: R 155.1.108.0/24 (0) -> Se1/3 155.1.23.2 Notifying
BGP- ATF: T 155.1.108.10/32 (0) c=0x843C4198 Adding to client
notification queue
BGP- ATF: EVENT 155.1.108.0/24 RIB update DOWN
BGP- ATF: T 155.1.108.10/32 (0) c=0x843C4198 Query pending
BGP- ATF: R 155.1.108.0/24 (0) -> Se1/3 155.1.23.2 Deleting
BGP- ATF: EVENT Query 155.1.108.0/24 (0) found route
BGP- ATF: 155.1.108.0/24 (0) Adding route
BGP- ATF: R 155.1.108.0/24 (0) -> Updating route
BGP- ATF: R 155.1.108.0/24 (0) -> Se1/3 155.1.23.2 Notifying
BGP- ATF: T 155.1.108.10/32 (0) c=0x843C4198 Adding to client
notification queue
BGP- ATF: EVENT 155.1.108.0/24 RIB update UP
BGP- ATF: R 155.1.108.0/24 (0) -> Se1/3 155.1.23.2 Updating route
BGP- ATF: R 155.1.108.0/24 (0) -> Se1/3 155.1.23.2 Update route already
in filter.
```

```
BGP- ATF: EVENT 150.1.10.0/24 RIB update DOWN
BGP- ATF: EVENT 150.1.10.0/24 RIB update DOWN
BGP- ATF: EVENT 150.1.10.0/24 RIB update UP
BGP- ATF: EVENT 155.1.10.0/24 RIB update DOWN
BGP- ATF: EVENT 155.1.10.0/24 RIB update DOWN
BGP- ATF: EVENT 155.1.10.0/24 RIB update UP
BGP- ATF: EVENT 155.1.108.0/24 RIB update DOWN
BGP- ATF: T 155.1.108.10/32 (0) c=0x843C4198 Query pending
BGP- ATF: R 155.1.108.0/24 (0) -> Se1/3 155.1.23.2 Deleting
BGP- ATF: EVENT Query 155.1.108.0/24 (0) did not find route
BGP- ATF: Notifying 155.1.108.0/24 (0)
BGP- ATF: T 155.1.108.10/32 (0) c=0x843C4198 Adding to client
notification queue
BGP- ATF: EVENT 155.1.108.0/24 RIB update DOWN
BGP- ATF: EVENT 150.1.10.0/24 RIB update DOWN
BGP- ATF: EVENT 150.1.10.0/24 RIB update DOWN
BGP- ATF: EVENT 155.1.10.0/24 RIB update DOWN
BGP- ATF: EVENT 155.1.10.0/24 RIB update DOWN
BGP- ATF: T 155.1.108.10/32 (0) c=0x843C4198 EVENT Track stop
BGP- ATF: T 155.1.108.10/32 (0) c=0x843C4198 Removing
%BGP-5-ADJCHANGE: neighbor 155.1.108.10 Down Route to peer lost
```

Note

Notice that now BGP brings the peering session down because the route to reach the peer is lost.

Copyright © 2010 Internetwork Expert

7.57 BGP Outbound Route Filtering

- R1 and R4 should filter out the prefixes 112.0.0.0/8 and 114.0.0.0/8 from being advertised to R3 and R5 respectively.
- The filtering configuration should be applied to routers R3 and R5.

Configuration

Note

ORF or outbound route filtering is the technique that allows a BGP peer to "push" a filter to the remote neighbor. The neighbor then applies the prefix filter to the outbound updates sent to the peer that pushed the filter. This feature is particularly helpful in situations when BGP peers exchange large amount of BGP information. Applying filtering outbound on the remote peer instead of inbound on the local peer significantly decreases the amount of routing information send across the link. There are two types of ORF filters defined in IETF's draft – prefix-list based and community based. Cisco IOS supports only the prefix-list ORFs.

In BGP terms, ORF is a special capability negotiated during the establishment of a BGP session. A peer may either advertise its willingness to send, receive or both send and receive the ORFs. You have to enable this capability on peering routers prior to configuring ORFs. The command to enable the feature in the IOS routers is **`neighbor <IP> capability orf prefix-list {send|receive|both}`**. You need to reset the BGP session in order to negotiate the new capabilities.

In order to configure and push an ORF, you need to define a prefix list and apply it to the peer's session using the command **`neighbor <IP> prefix-list <NAME>`** in. The list must be inbound, as this is the natural direction for ORF. If the session has ORF send capability enabled, the list will be pushed to the remote peer and installed as and outbound filter after you do a session refresh using the **`clear ip bgp <IP> soft in prefix-filter`** command. This command pushes the prefix list and requires route refresh (re-advertisement) from the peer.

Copyright © 2010 Internetwork Expert

```
R1:
router bgp 100
 neighbor 155.1.13.3 capability orf prefix-list both

R4:
router bgp 100
 neighbor 155.1.45.5 capability orf prefix-list both

R3:
ip prefix-list ORF deny 112.0.0.0/8
ip prefix-list ORF deny 114.0.0.0/8
ip prefix-list ORF permit 0.0.0.0/0 le 32
!
router bgp 200
 neighbor 155.1.13.1 capability orf prefix-list both
 neighbor 155.1.13.1 prefix-list ORF in

R5:
ip prefix-list ORF deny 112.0.0.0/8
ip prefix-list ORF deny 114.0.0.0/8
ip prefix-list ORF permit 0.0.0.0/0 le 32
!
router bgp 200
 neighbor 155.1.45.4 capability orf prefix-list both
 neighbor 155.1.45.4 prefix-list ORF in
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> To verify ORFs, issue the following "show" command on any of the routers "pushing" the filters, e.g. on R3. Notice that the new capability has been negotiated and then of the list sent.

```
Rack1R3#show ip bgp neighbors 155.1.13.1
BGP neighbor is 155.1.13.1,  remote AS 100, external link
<snip>

 For address family: IPv4 Unicast
  BGP table version 2, neighbor version 2/0
 Output queue size : 0
  Index 4, Offset 0, Mask 0x10
  4 update-group member
  AF-dependant capabilities:
    Outbound Route Filter (ORF) type (128) Prefix-list:
      Send-mode: advertised, received
      Receive-mode: advertised, received
  Outbound Route Filter (ORF): sent;
  Incoming update prefix filter list is ORF
                                Sent       Rcvd
  Prefix activity:              ----       ----
    Prefixes Current:              0          1 (Consumes 52 bytes)
    Prefixes Total:                0          1
    Implicit Withdraw:             0          0
    Explicit Withdraw:             0          0
    Used as bestpath:            n/a          1
    Used as multipath:           n/a          0

                                  Outbound    Inbound
  Local Policy Denied Prefixes:   --------    -------
    Bestpath from this peer:              1        n/a
    Total:                                1          0
  Number of NLRIs in the update sent: max 1, min 1
  <snip>
```

> **Note**
>
> Get to the other side of the connection and check the prefix list received by R1. Notice the name for the list, constructed from the peer's IP address.

```
Rack1R1#show ip bgp neighbors 155.1.13.3 received prefix-filter
Address family: IPv4 Unicast
ip prefix-list 155.1.13.3: 3 entries
   seq 5 deny 112.0.0.0/8
   seq 10 deny 114.0.0.0/8
   seq 15 permit 0.0.0.0/0 le 32
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.58 BGP Soft Reconfiguration

- Configure R4 to accept all prefixes from BB3 irrespective of the configured inbound filters and store them all locally.

Configuration

Note

Until RFC2918 introduced the Route Refresh capability to BGP it was impossible to signal a remote BGP peer to re-advertise the prefixes (Adj-RIB-Out) to the local peer. This feature is very helpful in situations when the local peer changes inbound filtering policy and needs the peer to re-advertise the routing information. The only way to accomplish the policy refresh was to tear down and re-establish the peering session, which could be very resource consuming and causes connectivity disruption. Before the Route Refresh capability became standardized, one workaround was to use the so-called soft-reconfiguration approach.

When a local BGP speaker is configured to apply soft-reconfiguration to a peer using the command **neighbor <IP> soft-reconfiguration inbound** the speaker will accept ALL prefixes from the remote peer and store them in a separate memory buffer (of course, a session reset is required for this operation to initialize). The prefixes are then processed via the inbound filters and the resulting information imported into Adj-RIB-In and finally to the BGP table. Every time the local policy changes, there is no need to re-establish the peering session but simply apply the filters to the stored information. The penalty is the extra memory needed to store the routing information from the peer. Of course, this feature became deprecated with the introduction of RR capability.

```
R4:
router bgp 100
 neighbor 204.12.1.254 soft-reconfiguration inbound
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the routes received and stored from BB2:

```
Rack1R4#show ip bgp neighbors 204.12.1.254 received-routes
BGP table version is 18, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   204.12.1.254             0             0 54 i
*> 28.119.17.0/24   204.12.1.254             0             0 54 i
*> 112.0.0.0        204.12.1.254                           0 54 50 60 i
*> 113.0.0.0        204.12.1.254                           0 54 50 60 i
*> 114.0.0.0        204.12.1.254                           0 54 i
*> 115.0.0.0        204.12.1.254                           0 54 i
*> 116.0.0.0        204.12.1.254                           0 54 i
*> 117.0.0.0        204.12.1.254                           0 54 i
*> 118.0.0.0        204.12.1.254                           0 54 i
*> 119.0.0.0        204.12.1.254                           0 54 i

Total number of prefixes 10
```

> **Note**
>
> Now, apply a prefix filter to the peering session with BB3, filtering all possible prefixes. Apply soft reset to the peering session and check the Adj-RIB-In (routes received from BB3 after filtering) with the total number of routes received from BB3.

```
Rack1R4(config)#ip prefix-list TEST deny 0.0.0.0/0 le 32
Rack1R4(config)#router bgp 100
Rack1R4(config-router)#neighbor 204.12.1.254 prefix-list TEST in

Rack1R4#clear ip bgp 204.12.1.254 soft in
Rack1R4#show ip bgp neighbors 204.12.1.254 routes

Total number of prefixes 0
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ip bgp neighbors 204.12.1.254 received-routes
BGP table version is 38, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  28.119.16.0/24   204.12.1.254             0             0 54 i
*  28.119.17.0/24   204.12.1.254             0             0 54 i
*  112.0.0.0        204.12.1.254                           0 54 50 60 i
*  113.0.0.0        204.12.1.254                           0 54 50 60 i
*  114.0.0.0        204.12.1.254                           0 54 i
*  115.0.0.0        204.12.1.254                           0 54 i
*  116.0.0.0        204.12.1.254                           0 54 i
*  117.0.0.0        204.12.1.254                           0 54 i
*  118.0.0.0        204.12.1.254                           0 54 i
*  119.0.0.0        204.12.1.254                           0 54 i

Total number of prefixes 10
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

7.59 BGP Next-Hop Trigger

- Configure R3 to respond to BGP prefixes next-hop changes within 30 seconds of IGP prefix change.

Configuration

Note

Many times BGP NEXT_HOP attribute is used for recursive lookup through the IGP table to resolve the actual next-hop interface and router. Until IOS 12.3(14)T BGP was accounting for IGP information changes only during periodic BGP scans with the interval defined by the command **bgp scan-time <seconds>**. With the default value of 60 seconds, it was impossible to react to IGP topology changes between the runs of the BGP scanner process.

Since IOS 12.3(14)T the next-hop tracking behavior has been changed from periodic to event-driven. This behavior is enabled by default using the command **bgp nexthop trigger enable**. BGP process registers the NEXT_HOP attribute values with the RIB table watch process. As soon as any change that affects an existing NEXT_HOP occurs, the watch process notifies the BGP router process. If the change results in prefix withdrawn, the BGP process immediately removes the prefix. All other notification are delayed and batched until the time-interval specified by the command **bgp nexthop trigger delay <seconds>** expires. After this, a full BGP table walk occurs, performing best-path computations for all prefixes. The delay value should be tuned according to the IGP convergence speed to avoid unnecessary full table walks. That is, it is desirable to have IGP fully converged after an initial change (or sequence of change) until the full BGP walk has started.

```
R3:
router bgp 200
 bgp nexthop trigger delay 30
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Advertise a new network 10.10.10.0/24 into BGP on SW4. Next, configure R5 so that the Frame-Relay mapping for R3 is removed. This we lead to R3 and R5 losing EIGRP adjacency in 3 minutes. Prior to this, enable the following debugging in R3: **`debug ip bgp event nexthop`**.

```
R5:
interface Serial 0/0
 no frame-relay map ip 155.1.0.3 503 broadcast

SW4:
interface Loobpack1
 ip address 10.10.10.10 255.255.255.0
!
router bgp 200
 network 10.10.10.0 mask 255.255.255.0
```

> **Note**
>
> Observe the debugging output on R3. Notice that the BGP process responds to IGP changes and schedules a BGP table walk in 30 seconds.

```
03:21:28.615: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 155.1.0.5
(Serial1/0.1) is down: holding time expired
03:21:28.631: EvD: accum. penalty decayed to 0 after 232 second(s)
03:21:28.636: BGP(IPv4 Unicast): Nexthop modified, reuse in 00:00:19,
19000 , scheduling nexthop scan in 30 secs
03:21:28.636: EvD: accum. penalty decayed to 500 after 0 second(s)
03:21:28.636: BGP(IPv4 Unicast): Nexthop modified, reuse in 00:00:27,
27000 , timer already running
03:21:58.637: BGP: NHOP scanner event timer
03:21:58.637: BGP: Nexthop walk for IPv4 Unicast
```

Copyright © 2010 Internetwork Expert

7.60 BGP TTL Security

- Configure R3 to accept TCP packets from eBGP peers only if they are no more than one hop away.

Configuration

Note

General TTL Security Mechanism (GTSM) defined in RFC 3682 specifies a protection method against BGP session hijacking and resource exhaustion attacks. Generally, BGP process listens on the TCP port 139 and accepts all TCP SYN packets destined to this port, unless they are filtered by an ACL. It is possible to generate a barrage of spoofed packets imitating a valid BGP session and inject false information (if the session is unauthenticated) or generate a TCP SYN-flooding attack.

GTSM utilizes the simple fact that every router on the path to the BGP speaker decrements the TTL field in IP packets by one. Based on this, it is possible to identify potentially spoofed packets by looking at their TTL field – the packets send from "afar" will have the TTL field below some threshold. It is possible to define a "secure radius" in the number of hop counts to accept the incoming IP packets. For example, if all BGP peers are within 10 hops away from the local BGP speaker, then all incoming IP packets will have their TTL field set to at no less than 245. This is because all IP packets start with TTL=255 and the field is decremented by every hop on the path. Thus, by accepting the IP packets with TTL greater than or equal to 245 it is possible to minimize the risk of spoofed packets reaching the BGP process. Notice that the usefulness of GTSM feature decreases as the diameter of eBGP Multihop session grows.

In order to configure the TTL security checks for a BGP peer use the command **neighbor <IP> ttl-security hops <hop-count>**. This command applies to eBGP peering sessions only (either directly-connected or multihop) and specifies the number of hops the remote peer could be away from the local speaker. Keep in mind the internal BGP sessions are not protected, and therefore the internal network assumed to be "trusted". All packets incoming TCP packets targeted at BGP port with the IP TTL value below (255 - <hop-count>) are silently discarded by the router. In addition, the feature sets TTL value for outgoing TCP/IP packets to 255-<hop-count> to make sure the remote peer will accept the local packets. The GTSM feature is mutually exclusive with the **ebgp-multihop** BGP feature. This is because the eBGP session by default sets TTL=1 in the outgoing IP packets and with the **multihop <n>** session parameter, the TTL value is set to <n>, which is not compatible with GTSM Therefore, make sure you configured GTSM feature on both sides of the peering link.

Copyright © 2010 Internetwork Expert

```
R1:
router bgp 100
 neighbor 155.1.13.3 ttl-security hops 1

R3:
router bgp 200
 neighbor 155.1.13.1 ttl-security hops 1
 neighbor 155.1.37.7 ttl-security hops 1

SW1:
router bgp 300
 neighbor 155.1.37.3 ttl-security hops 1
```

Verification

> **Note**
>
> Check the GTSM settings for eBGP peers on R3. Repeat the same verifications on R1 and SW1.

```
Rack1R3#show ip bgp neighbors 155.1.13.1 | inc hop
  External BGP neighbor may be up to 1 hop away.

Rack1R3#show ip bgp neighbors 155.1.37.7 | inc hop
  External BGP neighbor may be up to 1 hop away.
```

Copyright © 2010 Internetwork Expert

7.61 BGP AllowAS in

- Configure R2 and SW2 to advertise networks 2.2.2.0/24 and 8.8.8.0/24 into BGP.
- Configure AS 200 border routers so that in case AS 200 is partitioned, the remaining segments could transit AS 100 to recover connectivity.

Configuration

Note

BGP loop-prevention mechanism does not allow a BGP speaker to accept prefixes with the local AS number in the AS_PATH list. However, in some cases, it would be desirable to accept the routes originated in the same AS via another AS. There are two common scenarios:

1) The company's network is partitioned and every partition connects to the Internet or ISP. Every network has its own set of prefixes but uses the same AS number. In this case, in order for the partitions to exchange prefixes they must accept the NLRIs with the same AS number.

2) The company connects to an ISP and wants to use it as a transit path in case the company's network becomes segmented due to an emergency. In this case, the prefixes advertised to the ISP must be accepted back by the border peers.

Cisco IOS allows for accepting the prefixes with the local AS number from a specific peer using the command **`neighbor <IP> allowas-in [<count>]`**. Here <count> is the number of the local AS number occurrences in the AS_PATH attribute, which defaults to one. This parameter serves the purpose similar to the hop-count limit in distance-vector protocol and implement the well-know count-to-infinity loop prevention technique.

In order to prevent routing loops with this feature, you should be careful implementing prefix aggregation. Specifically, only one "partition" or border peer could implement summarization, or summarization should not be used at all. Otherwise, the upstream ASes will have troubles selecting the proper entry point to the AS partitions. Needless to mention that using the AllowAS in feature is highly un-recommended and only advised as a last resort.

Copyright © 2010 Internetwork Expert

```
R2:
router bgp 200
 network 2.2.2.0 mask 255.255.255.0
!
interface Loopback1
 ip address 2.2.2.2 255.255.255.0

R3:
router bgp 200
 neighbor  155.1.13.1 allowas-in

R5:
router bgp 200
 neighbor 155.1.45.4 allowas-in

SW2:
router bgp 200
 network 8.8.8.0 mask 255.255.255.0
!
interface Loopback1
 ip address 8.8.8.8 255.255.255.0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Configure the routers so that AS 200 split in two parts. To accomplish this, configure the routers as follows:

R3:

```
router eigrp 1
 passive-interface FastEthernet 0/0
 passive-interface Serial 1/2
```

R5:

```
router eigrp 1
 passive-interface Serial 0/0
 passive-interface Serial 0/1
!
interface Serial 0/0
 shutdown
```

> **Note**
>
> Check the BGP tables of R3 and R5 for the prefixes originated in AS 200. Notice that both R3 and R5 accept those prefixes due to the AllowAS in feature. Next, trace the route from R2 to SW2 between the two configured subnets and make sure connectivity is maintained.

```
Rack1R3#show ip bgp regexp _200$
BGP table version is 79, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 8.8.8.0/24       155.1.13.1                             0 100 200 i
```

```
Rack1R5#show ip bgp regexp _200$
BGP table version is 94, local router ID is 150.1.5.5
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 2.2.2.0/24       155.1.45.4                             0 100 200 i
```

Copyright © 2010 Internetwork Expert

```
Rack1R2#traceroute 8.8.8.8 source loopback 1

Type escape sequence to abort.
Tracing the route to 8.8.8.8

  1 155.1.23.3 [AS 100] 16 msec 16 msec 16 msec
  2 155.1.13.1 [AS 100] 28 msec 28 msec 32 msec
  3 155.1.146.4 [AS 100] 28 msec 28 msec 32 msec
  4 155.1.45.5 [AS 100] 40 msec 44 msec 40 msec
  5 155.1.58.8 [AS 100] 44 msec *  40 msec
```

Copyright © 2010 Internetwork Expert

Multicast Solutions

Note

Load the *Multicast* initial configurations prior to starting.

Configuration Advisory

Throughout this workbook you will be required to use several show commands to verify the success of different configuration tasks. You may find that you will need to perform a “clear ip mroute *” on routers to facilitate expiration of previous table entries or to get things to “synch-up”. Where this fails, and you feel certain your configurations are correct it is suggested that you write/reload your router before beginning extensive troubleshooting. We have found that sometimes this process is necessary when you are making many modifications to a running multicast environment.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

8.1 PIM Dense Mode

- Enable dense-mode multicast delivery on the path between R6 and SW4.
- Do not enable PIM on the Frame-Relay cloud.
- To test this configuration, configure SW4's VLAN 10 interface to join the multicast group 224.10.10.10.
- Make sure R6 can ping this multicast group.

Configuration

Note

Multicast traffic distribution uses the concept of "constrained" flooding. This means that packets destined to a particular multicast group are flooded only on the links that lead to the actual group subscribers. The constrained flooding procedure consists of two parts

1) Building a multicast distribution tree for a group – aka Shortest Path Tree or SPT.
2) Flooding the actual multicast packets down the SPT, using Reverse Path Forwarding (RPF).

We are going to work mostly with Protocol Independent Multicast (PIM) Dense Mode and Sparse Mode. PIM is a special signaling protocol that does not distribute any routing information on its own. Rather, PIM uses the unicast routing table to perform RPF checks. This procedure is the core of multicast routing. When the router receives a multicast packet, it looks at the source IP address for the packet. The router looks up the source IP address in the unicast routing table, and determines the outgoing interface for this packet. If this outgoing interface does not match the interface where the packet was received, the router drops the packet. This behavior intends to eliminate potential packet loops, which may occur when routers flood multicast packets. Only the immediate upstream router (with respect to the source of the multicast stream) is allowed to send us multicast packets.

Copyright © 2010 Internetwork Expert

If the packet passes the RPF check, the router will determine the outgoing interface list (OIL) for this packet, i.e. the branches of the multicast distribution tree. The OIL never includes the input interface – this is the well-known split horizon rule. The SPT should be signaled by PIM, based on the active subscribers across the network. However, PIM Dense Mode (PIM DM) does not use any explicit signaling to join a multicast distribution tree. Instead of that, it uses the inverse logic approach. All routers initially flood packets out of all their multicast-enabled interfaces. If the downstream router determines that it does not have any directly connected subscribers or other routers willing to receive multicast traffic, it sends a special PIM Prune message to the upstream router. The upstream router then excludes the particular interface from the OIL for the pruned group.

PIM DM works fine in small networks with a lot of subscribers. However, it has one inherent scalability limitation – excessive flooding. Every Pruned interface state expires in 3 minutes by default, and then flooding out of this interface resumes again, until the upstream does not receive another PIM Prune message on the interface. This is needed to ensure that no node in the network potentially misses multicast traffic. This periodic flood and prune behavior is what makes PIM Dense mode non-scalable.

The obvious benefit of PIM DM is its “plug-and-play” behavior. As soon as you configure PIM DM in your network, you can start sending multicast traffic and the traffic is flooded to all interested subscribers.

Copyright © 2010 Internetwork Expert

```
R4:
ip multicast-routing
!
interface FastEthernet 0/1
 ip pim dense-mode
!
interface Serial 0/1/0
 ip pim dense-mode

R6:
ip multicast-routing
!
interface FastEthernet 0/0.146
 ip pim dense-mode

R5:
ip multicast-routing
!
interface Serial 0/1/0
 ip pim dense-mode
!
interface FastEthernet 0/0
 ip pim dense-mode

SW2:
ip multicast-routing distributed
!
interface Vlan 58
 ip pim dense-mode
!
interface Port-Channel1
 ip pim dense-mode

SW4:
ip multicast-routing
!
interface Port-Channel1
 ip pim dense-mode
!
interface Vlan 10
 ip igmp join-group 224.10.10.10
 ip pim dense-mode
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Start your basic verification by looking for PIM neighbors and PIM interfaces. Notice the PIM version and mode on every interface and make sure they match the requirements in the tasks.

```
Rack1R5#show ip pim neighbor
PIM Neighbor Table
Mode: B - Bidir Capable, DR - Designated Router, N - Default DR
Priority,
      S - State Refresh Capable
Neighbor          Interface                Uptime/Expires    Ver   DR
Address
Prio/Mode
155.1.45.4        Serial0/1/0              00:09:43/00:01:22 v2    1 /
S
155.1.58.8        FastEthernet0/0          00:09:28/00:01:35 v2    1 /
DR S

Rack1R5#show ip pim interface

Address          Interface                Ver/   Nbr    Query  DR
DR
                                          Mode   Count  Intvl  Prior
155.1.45.5       Serial0/1/0              v2/D   1      30     1
0.0.0.0
155.1.58.5       FastEthernet0/0          v2/D   1      30     1
155.1.58.8
```

> **Note**
>
> Ping the group and confirm that the packets reach the destination.

```
Rack1R6#ping 224.10.10.10 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 224.10.10.10, timeout is 2 seconds:

Reply to request 0 from 155.1.108.10, 32 ms
Reply to request 1 from 155.1.108.10, 32 ms
Reply to request 2 from 155.1.108.10, 28 ms
```

Copyright © 2010 Internetwork Expert

Note

Now verify multicast route states created by the traffic flow. You should see (S,G) entries corresponding to the dense-mode SPT. Notice that (*,G) states have all potential interfaces within their OILs. Look for the RPF neighbor for each entry, and make sure they match the traffic flow. The RPF neighbor of 0.0.0.0 means this router is connected to the source.

```
Rack1R4#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:00:24/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/1/0, Forward/Dense, 00:00:24/00:00:00
    FastEthernet0/1, Forward/Dense, 00:00:24/00:00:00

(155.1.146.6, 224.10.10.10), 00:00:24/00:02:57, flags: T
  Incoming interface: FastEthernet0/1, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/1/0, Forward/Dense, 00:00:25/00:00:00

(*, 224.0.1.40), 00:02:26/00:02:05, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/1/0, Forward/Dense, 00:01:49/00:00:00
    FastEthernet0/1, Forward/Dense, 00:02:26/00:00:00

Rack1R4#
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:00:42/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:00:42/00:00:00
    Serial0/1/0, Forward/Dense, 00:00:42/00:00:00

(155.1.146.6, 224.10.10.10), 00:00:42/00:02:57, flags: T
  Incoming interface: Serial0/1/0, RPF nbr 155.1.45.4
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:00:43/00:00:00

(*, 224.0.1.40), 00:02:07/00:02:59, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:01:52/00:00:00
    Serial0/1/0, Forward/Dense, 00:02:07/00:00:00

Rack1R5#
Rack11AS>8
[Resuming connection 8 to sw2 ... ]

Rack1SW2#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report, Z - Multicast
Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:00:53/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Port-channel1, Forward/Dense, 00:00:53/00:00:00
    Vlan58, Forward/Dense, 00:00:53/00:00:00
```

```
(155.1.146.6, 224.10.10.10), 00:00:53/00:02:52, flags: T
  Incoming interface: Vlan58, RPF nbr 155.1.58.5
  Outgoing interface list:
    Port-channel1, Forward/Dense, 00:00:53/00:00:00

(*, 224.0.1.40), 00:01:48/00:02:49, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Port-channel1, Forward/Dense, 00:01:34/00:00:00
    Vlan58, Forward/Dense, 00:01:48/00:00:00

Rack1SW2#
```

```
Rack1SW4#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report, Z - Multicast
Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:01:32/stopped, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Port-channel1, Forward/Dense, 00:01:32/00:00:00
    Vlan10, Forward/Dense, 00:01:32/00:00:00

(155.1.146.6, 224.10.10.10), 00:01:07/00:02:59, flags: LT
  Incoming interface: Port-channel1, RPF nbr 155.1.108.8
  Outgoing interface list:
    Vlan10, Forward/Dense, 00:01:07/00:00:00, H

(*, 224.0.1.40), 00:01:33/00:02:16, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Port-channel1, Forward/Dense, 00:01:33/00:00:00
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

8.2 Multicast RPF Failure

- Disable PIM on the Serial link between R4 and R5 and activate it on the Frame-Relay connection between R4 and R5.
- Ensure that R6 can still ping the group 224.10.10.10.
- Set R5's periodic RPF checks interval to 6 seconds.
- To reduce the load on the router's CPU configure R5 to perform triggered RPF checks 10 ms after a topology change but don't allow the delay to ever be longer than 200ms.

Configuration

Note

The RPF check procedure has been discussed in the previous task. RPF failures occur in two general situations: packets being flooded out of the wrong interface (good, prevents looping) or unicast shortest paths not matching multicast distribution trees (bad, incorrect multicast routing configuration). Situations of the second type generally occur in the following cases:

1) Duplicate paths to the multicast source exist and PIM is not enabled on all links. In this situation, the router may receive multicast packets on an interface that is not on the shortest path to the source. As an extreme solution to this situation, PIM might be enabled on the links where IGP is not running.

2) There are multiple routing protocols in the network, and the router might have paths to the source learned via different protocols. In this situation, the router may receive multicast packets on an interface running RIP, while thinking the shortest path to the source is via OSPF.

3) Warning! Use of static routes may the change a local router's perception of the shortest paths and force it to reject otherwise valid multicast packets.

Copyright © 2010 Internetwork Expert

Additionally, temporary RPF failures may happen in situations where the network is unstable and the routers keep changing their shortest path tables. The best way to find and isolate the RPF issue in your lab is probably to run the **debug ip mpacket** command on all routers, starting from the one closest to the destination and moving upstream to the source. This command will show you process-switched multicast packets and signal any RPF issues. Note that you will need to enter the command **no ip mroute-cache** on all interfaces where multicast packets are received, this will disable fast-switching of multicast packets on the interfaces where is specified.

The easiest way to deal with an RPF failure is to manually provide RPF information using the **ip mroute** command. This command is local to the router and specifies the RPF next-hop or interface for the given subnet. The syntax for the command is **ip mroute <IP> <Mask> {RPF-IP-Address|<Interface-Name>} [distance]**. It is important to understand that this command does not specify any forwarding rules. Instead, it creates an ordered table of entries to look up for RPF information. When a multicast packet comes in, the router will first try finding a matching entry in the mroute table, to determine the RPF interface. Note that the mroute table is ordered in the same way you enter the ip mroute commands, so you should enter the most specific mroutes first. When the router finds RPF information in both the mroute table and the unicast routing table, it prefers mroute information, since it has a better distance (zero). However, connected routes still have preference over any other RPF source. You may change the mroute administrative distance to make it less preferred than the unicast routing table information.

When a network topology change occurs, the routers will delay their RPF interface re-computation. You may change this delay using the command **ip multicast rpf backoff <min-delay ms> <max-delay ms>.** The **min-delay** value specifies the amount of milliseconds to wait after the topology change to re-calculate RPF interfaces. This delay is doubled after every consequent topology change that happens within the **max-delay** window. The delay however will never become larger that the **max-delay** field. Additionally, you can tune periodic RPF interface computations using the command **ip multicast rpf interval <interval in seconds>**.

Copyright © 2010 Internetwork Expert

R4:

```
interface Serial 0/1/0
 no ip pim dense-mode
!
interface Serial 0/0/0.1
 ip pim dense-mode
```

R5:

```
interface Serial 0/1/0
 no ip pim dense-mode
!
interface Serial 0/0/0
 ip pim dense-mode
!
ip multicast rpf interval 6
ip multicast rpf backoff 10 200
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Try pinging the group before you apply the mroute needed to correct the RPF lookup error.

```
Rack1R6#ping 224.10.10.10 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 224.10.10.10, timeout is 2 seconds:
..
```

> **Note**
>
> Check the multicast routing table on R5. Notice that the (S,G) state for our flow has no Incoming Interface field (it's Null) this means the traffic is not arriving on the RPF interface.

```
Rack1R5#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:05:57/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0/0, Forward/Dense, 00:05:57/00:00:00
    FastEthernet0/0, Forward/Dense, 00:05:57/00:00:00

(155.1.146.6, 224.10.10.10), 00:05:57/00:00:25, flags:
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:05:58/00:00:00

(*, 224.0.1.40), 00:34:33/00:02:34, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0/0, Forward/Dense, 00:06:16/00:00:00
    FastEthernet0/0, Forward/Dense, 00:34:18/00:00:00
```

Copyright © 2010 Internetwork Expert

> **Note**
>
> You can easily see RPF failures when you enable multicast packet debugging.

```
Rack1R5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial 0/0/0
Rack1R5(config-if)#no ip mroute-cache

Rack1R5#debug ip mpacket
IP multicast packets debugging is on
Rack1R5#
IP(0): s=155.1.146.6 (Serial0/0/0) d=224.10.10.10 id=300, ttl=253,
prot=1, len=104(100), not RPF interface
Rack1R5#
IP(0): s=155.1.146.6 (Serial0/0/0) d=224.10.10.10 id=301, ttl=253,
prot=1, len=104(100), not RPF interface
Rack1R5#
```

> **Note**
>
> Now apply the static multicast route on R5. A static default mroute will be enough in this situation.

```
Rack1R5(config)#ip mroute 0.0.0.0 0.0.0.0 155.1.0.4
Rack1R5(config)#
Rack11AS>6
[Resuming connection 6 to r6 ... ]
...
Reply to request 98 from 155.1.108.10, 48 ms
Reply to request 99 from 155.1.108.10, 48 ms
Rack1R6#
```

Copyright © 2010 Internetwork Expert

Note

Now the corresponding mroute state has a valid input interface and the OIL is not empty. Notice that the RPF neighbor information is taken from the mroute cache, per the output of the show command.

```
Rack1R5#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:09:16/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:09:16/00:00:00
    Serial0/0/0, Forward/Dense, 00:09:16/00:00:00

(155.1.146.6, 224.10.10.10), 00:02:52/00:02:56, flags: T
  Incoming interface: Serial0/0/0, RPF nbr 155.1.0.4, Mroute
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:02:53/00:00:00

(*, 224.0.1.40), 00:37:52/00:02:13, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:37:37/00:00:00
    Serial0/0/0, Forward/Dense, 00:09:35/00:00:00
```

Copyright © 2010 Internetwork Expert

8.3 PIM Sparse Mode

- Remove the PIM Dense Mode configuration and replace it with Sparse Mode on all interfaces between R6 and SW4.
- Do NOT run PIM on the Frame Relay link between R4 and R5.
- Remove the static default mroute from R5.
- Use R5's Loopback 0 interface as the static Rendezvous Point address.
- Configure SW4's VLAN 10 interface to join the group 224.10.10.10, and ensure that R6 can send multicast packets to this segment.
- Perform SPT switchover only when the traffic flow exceeds 128Kbps.

Configuration

Note

PIM Sparse Mode is the scalable version of the multicast signaling protocol. Instead of flooding multicast traffic to all potential receivers, PIM SM builds explicit multicast distribution trees from the receivers to the sources. However, this model does not consume large amounts of network resources like PIM DM's flood-and-prune behavior does.

In order to build an explicit tree to the source, receivers and sources should have a way to dynamically discover each other. This is facilitated by the use of Rendezvous Points or RPs. An RP is a special router known to both sources and receivers. If a receiver is interested in traffic for a multicast group G, it first builds a multicast distribution tree towards the RP as the fictive "source". This is facilitated by PIM Join messages, and the resulting tree is called a shared tree and it is designated as (*,G). When a source appears in the network the closest multicast router will contact the RP using PIM Register messages. The PIM Register messages are sent toward the RP across the unicast shortest path. PIM should be enabled along the shortest path to the RP, or the RPF check for the RP will fail and the registration process will not be completed.

Copyright © 2010 Internetwork Expert

When registration completes, the RP builds a new SPT towards the source using PIM Join messages and starts forwarding received multicast traffic down the (*,G) tree. When the receivers see traffic going down the (*,G) tree from the actual source they initiate a PIM Join toward the source, building another SPT designated as the (S,G) tree. This new tree follows the optimal path to the source and removes the RP as a possible "bottleneck" of all multicast traffic flows. This process of switching from (*,G) to the (S,G) tree is called multicast SPT switchover. Finally, the receiving router will send a special Prune message toward the RP designated as (S, G, RPbit). This will prune back the duplicate traffic down the (*,G) tree from the RP.

Configuring the RP for multicast groups is a crucial part of PIM SM setup. In this scenario, we use a static RP configuration on every router using the command **ip pim rp-address <IP> [<ACL>] [override]**. The ACL parameter lists the groups that are mapped to this particular RP. You could have multiple RP's using different group lists on every router, for the purpose of load-balancing. Later we will see ways of automatically disseminating RP information using Auto-RP and BSR protocols. But for now, keep in mind that the **override** parameter will force the router to retain static information even if a different RP for the group is learned via Auto-RP.

Additionally, the task dictates that the switch over to the SPT should be done only when the traffic down the (*,G) tree exceeds 128Kbps. By default, the "leaf" router – the one closest to the receiver – will switch over as soon as it receives any multicast traffic down the (*,G) tree. This threshold can be changed using the command **ip pim spt-threshold {<Rate in Kbps>|infinity}**. By setting the rate to infinity, you can effectively disable the use of the shortest-path tree entirely, and all information will be received down the shared tree.

In this scenario, there is a hidden issue. When you start pinging from R6, and run debug ip mpacket on R4 (the router that registers the source) you will see something like the following:

```
IP(0): MAC sa=0011.bbbd.64a0 (FastEthernet0/1), IP last-hop=155.1.146.6
IP(0): IP tos=0x0, len=100, id=201, ttl=254, prot=1
IP(0): s=155.1.146.6 (FastEthernet0/1) d=224.10.10.10 id=201, ttl=254,
prot=1, len=114(100), RPF lookup failed
```

The problem is that R5's Loopback 0 is known via OSPF, as a /32 route and known as a /24 subnet via EIGRP. PIM will attempt to register across the OSPF path, which is not enabled for PIM. Thus, the RPF check toward the RP will fail, and R4 will never register the source. To fix this problem, we add the **ip ospf network point-to-point** statement on R5's Loopback 0 configuration, to make OSPF advertise a /24 route as well.

Copyright © 2010 Internetwork Expert

R4:
```
ip multicast-routing
ip pim rp-address 150.1.5.5
!
interface FastEthernet 0/1
 ip pim sparse-mode
!
interface Serial 0/1/0
 ip pim sparse-mode
!
interface Serial 0/0/0.1
 no ip pim dense-mode
```

R5:
```
no ip mroute 0.0.0.0 0.0.0.0 155.1.0.4
!
ip multicast-routing
ip pim rp-address 150.1.5.5
!
interface Serial 0/1/0
 ip pim sparse-mode
!
interface Serial 0/0/0
 no ip pim dense-mode
 !
interface FastEthernet 0/0
 ip pim sparse-mode

!
! Needed to fix RPF check for R5's Loopback0
!
interface Loopback0
 ip ospf network point-to-point
```

R6:
```
ip multicast-routing
ip pim rp-address 150.1.5.5
!
interface FastEthernet 0/0.146
 ip pim sparse-mode
```

SW2:
```
ip multicast-routing distributed
ip pim rp-address 150.1.5.5
!
interface Vlan 58
 ip pim sparse-mode
!
interface Port-Channel1
 ip pim sparse-mode
```

Copyright © 2010 Internetwork Expert

SW4:

```
ip multicast-routing
ip pim rp-address 150.1.5.5
ip pim spt-threshold 128
!
interface Port-Channel1
 ip pim sparse-mode
!
interface Vlan 10
 ip igmp join-group 224.10.10.10
 ip pim sparse-mode
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Before you source any traffic, check that SW4 has joined the shared tree toward the RP. Notice the (*,G) states and the RP address for the group.

```
Rack1SW4#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report, Z - Multicast
Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:00:58/00:02:06, RP 150.1.5.5, flags: SJCL
  Incoming interface: Port-channel1, RPF nbr 155.1.108.8
  Outgoing interface list:
    Vlan10, Forward/Sparse, 00:00:56/00:02:06, H

Rack1SW2#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report, Z - Multicast
Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:02:41/00:02:47, RP 150.1.5.5, flags: S
  Incoming interface: Vlan58, RPF nbr 155.1.58.5
  Outgoing interface list:
    Port-channel1, Forward/Sparse, 00:02:41/00:02:47
```

Copyright © 2010 Internetwork Expert

> **Note**
>
> The RP itself shows the RPF neighbor of 0.0.0.0 for the (*,G) state, meaning it is the RPF neighbor. At the same time, R4 has no (*,G) state for the group 224.10.10.10 as it is not on the shared tree.

```
Rack1R5#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:03:36/00:02:52, RP 150.1.5.5, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:03:36/00:02:52

Rack1R4#show ip mroute 224.10.10.10
Group 224.10.10.10 not found
```

Copyright © 2010 Internetwork Expert

Note

When you start pinging from R6 and check mroute states on R4, you will notice that the (*,G) state for the group 224.10.10.10 is “prune”, as there are no receivers for this group below R4. At the same time, there is an (S,G) state toward R6 that enables traffic flow from R6 down to SW4.

```
Rack1R4#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:00:05/stopped, RP 150.1.5.5, flags: SP
  Incoming interface: Serial0/1/0, RPF nbr 155.1.45.5
  Outgoing interface list: Null

(155.1.146.6, 224.10.10.10), 00:00:05/00:02:54, flags: T
  Incoming interface: FastEthernet0/1, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/1/0, Forward/Sparse, 00:00:05/00:03:24
```

Copyright © 2010 Internetwork Expert

Note

When you look at the mroutes on R5, notice that there is a (*,224.10.10.10) state representing the shared tree and a (155.1.146.6,224.10.10.10) state representing the SPT built by the RP. This SPT is used to connect the multicast source to the shared tree.

```
Rack1R5#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:54:20/stopped, RP 150.1.5.5, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:54:20/00:02:42

(155.1.146.6, 224.10.10.10), 00:00:11/00:02:56, flags: T
  Incoming interface: Serial0/1/0, RPF nbr 155.1.45.4
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:00:11/00:02:48
```

Copyright © 2010 Internetwork Expert

Note

When you check the mroute states on SW2 and SW4 you will only see (*,G) entries. This is because the SPT switchover rate is set to 128Kbps and we haven't exceeded it yet. Thus SW4 never initialized SPT-switchover toward the source.

```
Rack1SW2#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report, Z - Multicast
Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:54:31/00:03:27, RP 150.1.5.5, flags: S
  Incoming interface: Vlan58, RPF nbr 155.1.58.5
  Outgoing interface list:
    Port-channel1, Forward/Sparse, 00:54:31/00:03:05
<snip>

Rack1SW4#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report, Z - Multicast
Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:55:15/00:02:58, RP 150.1.5.5, flags: SCL
  Incoming interface: Port-channel1, RPF nbr 155.1.108.8
  Outgoing interface list:
    Vlan10, Forward/Sparse, 00:55:14/00:02:54, H
<snip>
```

Copyright © 2010 Internetwork Expert

Note

In order to see the SPT switchover process, disable the threshold setting on SW4 and repeat the multicast testing again.

SW4:

```
no ip pim spt-threshold 128
```

```
Rack1R6#ping 224.10.10.10 repeat 1000

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 224.10.10.10, timeout is 2
seconds:

Reply to request 0 from 155.1.108.10, 49 ms
Reply to request 1 from 155.1.108.10, 32 ms
```

Note

Now you can see the SPT entry in the multicast routing table of SW2. In our case, the SPT and shared tree overlap (follow the same paths) but in other scenarios they may differ and this can result in RPF issues.

```
Rack1SW2#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report, Z - Multicast
Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 01:03:15/stopped, RP 150.1.5.5, flags: S
  Incoming interface: Vlan58, RPF nbr 155.1.58.5
  Outgoing interface list:
    Port-channel1, Forward/Sparse, 01:03:15/00:03:13

(155.1.146.6, 224.10.10.10), 00:00:00/00:03:29, flags:
  Incoming interface: Vlan58, RPF nbr 155.1.58.5
  Outgoing interface list:
    Port-channel1, Forward/Sparse, 00:00:00/00:03:29
```

Copyright © 2010 Internetwork Expert

8.4 PIM Sparse-Dense Mode

- Remove the previous PIM Sparse Mode configuration and replace it with Sparse-Dense-Mode on all interfaces.
- Do not enable PIM on the Frame Relay link between R4 and R5.
- Modify the static RP assignments so that R5 is only used as the RP for groups in the range 224.0.0.0/8.
- Configure SW4's VLAN 10 interface to join the groups 224.10.10.10 and 239.0.0.1, and ensure that R6 can send multicast packets to this segment for both groups.

Configuration

Note

PIM Sparse-Dense mode is a hybrid of the Sparse and Dense mode operations. However, it does not define any extension to the PIM protocol. Rather, when you apply the command **`ip pim sparse-dense-mode`** to an interface, the router will forward traffic for both sparse and dense multicast groups out of this interface. A "sparse" multicast group is the one that has an RP defined. This mode of operation is useful when you have a range of groups that you want to be delivered using the simple dense-mode. As we will see later, Auto-RP groups are a good example of dense-mode groups needed to be flooded along with sparse-groups.

Most of the time you don't have to define PIM SM/DM interfaces, unless you are using the Cisco proprietary Auto-RP protocol. The problem with PIM SM/DM is that any group that does not have an RP defined is treated like a dense-mode group. Consider the situation where some routers lose RP information for their sparse-mode groups. For example, say some routers haven't received Auto-RP announcements for some time and the old information has expired. In this situation, these routers will switch all groups to dense-mode and start flooding the network with multicast traffic. In order to prevent this behavior, you should either use PIM SM only along with a standard RP information dissemination protocol, such as BSR, or issue the command **`no ip dm-fallback`** on all PIM SM/DM routers. This will prevent the DM fallback behavior and only allow forwarding for sparse-mode groups.

Copyright © 2010 Internetwork Expert

R4:
```
ip access-list standard SPARSE_GROUPS
 permit 224.0.0.0 0.255.255.255
!
ip multicast-routing
ip pim rp-address 150.1.5.5 SPARSE_GROUPS
!
interface FastEthernet 0/1
 ip pim sparse-dense-mode
!
interface Serial 0/1/0
 ip pim sparse-dense-mode
```

R5:
```
ip access-list standard SPARSE_GROUPS
 permit 224.0.0.0 0.255.255.255
!
ip multicast-routing
ip pim rp-address 150.1.5.5 SPARSE_GROUPS
!
interface Serial 0/1/0
 ip pim sparse-dense-mode
!
interface FastEthernet 0/0
 ip pim sparse-dense-mode

!
! Needed to fix RPF check for R5's Loopback0
!
interface Loopback0
 ip ospf network point-to-point
```

R6:
```
ip access-list standard SPARSE_GROUPS
 permit 224.0.0.0 0.255.255.255
!
ip multicast-routing
ip pim rp-address 150.1.5.5 SPARSE_GROUPS
!
interface FastEthernet 0/0.146
 ip pim sparse-dense-mode
```

SW2:
```
ip multicast-routing distributed
ip access-list standard SPARSE_GROUPS
 permit 224.0.0.0 0.255.255.255

!
ip pim rp-address 150.1.5.5 SPARSE_GROUPS
!
interface Vlan 58
 ip pim sparse-dense-mode
!
interface Port-Channel1
 ip pim sparse-dense-mode
```

Copyright © 2010 Internetwork Expert

SW4:

```
ip access-list standard SPARSE_GROUPS
 permit 224.0.0.0 0.255.255.255
!
ip multicast-routing
ip pim rp-address 150.1.5.5 SPARSE_GROUPS
!
interface Port-Channel1
 ip pim sparse-dense-mode
!
interface Vlan 10
 ip igmp join-group 224.10.10.10
 ip igmp join-group 239.0.0.1
 ip pim sparse-dense-mode
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> First, ping the group that has no RP from R6. Notice that all routers on the path to the receiver create (S,G) dense state entries. There is also a corresponding (*,G) state as well, but it has an RP value of 0.0.0.0 and has all PIM-enabled interfaces listed in the OIL.

```
Rack1R6#ping 239.0.0.1 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 239.0.0.1, timeout is 2 seconds:

Reply to request 0 from 155.1.108.10, 37 ms
Reply to request 1 from 155.1.108.10, 32 ms

Rack1R4#show ip mroute 239.0.0.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.0.0.1), 00:00:10/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/1/0, Forward/Sparse-Dense, 00:00:10/00:00:00
    FastEthernet0/1, Forward/Sparse-Dense, 00:00:10/00:00:00

(155.1.146.6, 239.0.0.1), 00:00:10/00:02:55, flags: T
  Incoming interface: FastEthernet0/1, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/1/0, Forward/Sparse-Dense, 00:00:12/00:00:00
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip mroute 239.0.0.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.0.0.1), 00:00:26/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 00:00:26/00:00:00
    Serial0/1/0, Forward/Sparse-Dense, 00:00:26/00:00:00

(155.1.146.6, 239.0.0.1), 00:00:26/00:02:53, flags: T
  Incoming interface: Serial0/1/0, RPF nbr 155.1.45.4
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 00:00:27/00:00:00

Rack1SW2#show ip mroute 239.0.0.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report, Z - Multicast
Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.0.0.1), 00:00:42/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Port-channel1, Forward/Sparse-Dense, 00:00:42/00:00:00
    Vlan58, Forward/Sparse-Dense, 00:00:42/00:00:00

(155.1.146.6, 239.0.0.1), 00:00:42/00:02:53, flags: T
  Incoming interface: Vlan58, RPF nbr 155.1.58.5
  Outgoing interface list:
    Port-channel1, Forward/Sparse-Dense, 00:00:42/00:00:00
```

Note

Now, again from R6, ping the group that has an RP configured. Notice that (*,G) now represents the shared tree, and is used to forward the first packets from the sources. The SPT is built by the RP and the router is connected to the receiver.

```
Rack1R5#show ip mroute 224.10.10.10
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 02:20:25/stopped, RP 150.1.5.5, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 02:20:25/00:02:41

(155.1.146.6, 224.10.10.10), 00:00:10/00:03:21, flags: T
  Incoming interface: Serial0/1/0, RPF nbr 155.1.45.4
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 00:00:10/00:03:19
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show ip mroute 224.10.10.10
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report, Z - Multicast
Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 02:29:57/stopped, RP 150.1.5.5, flags: S
  Incoming interface: Vlan58, RPF nbr 155.1.58.5
  Outgoing interface list:
    Port-channel1, Forward/Sparse-Dense, 02:29:57/00:03:12

(155.1.146.6, 224.10.10.10), 00:00:03/00:03:29, flags: T
  Incoming interface: Vlan58, RPF nbr 155.1.58.5
  Outgoing interface list:
    Port-channel1, Forward/Sparse-Dense, 00:00:03/00:03:26
```

Copyright © 2010 Internetwork Expert

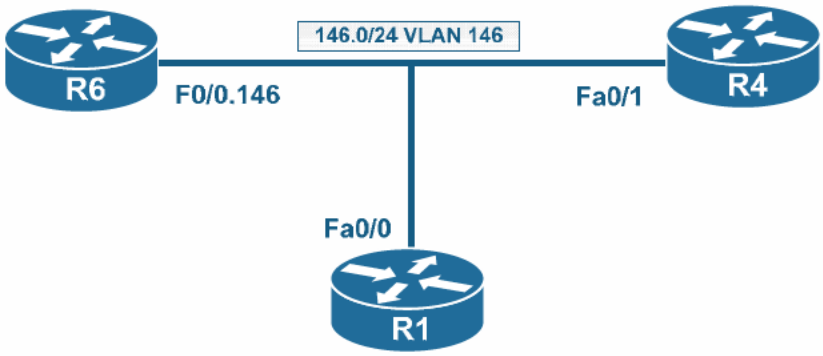
8.5 PIM Assert

- Enable multicast routing on R1 and configure PIM sparse-dense mode on the Fast Ethernet interface of R1.
- Configure PIM sparse-dense mode on the connection between R1 and R5.
- Use the same RP assignment that was configured on the other routers.
- Ensure R1 is always elected via PIM to flood traffic onto the shared VLAN 146 segment.

Configuration

Note

If multiple multicast routers share a single segment, then all of them could flood this segment with the same multicast traffic.



For example, if both R1 and R4 receive the same multicast flow from their upstream neighbors, they will both send it to R6. From R6's point of view, both flows are the same with respect to RPF validation. Thus, traffic duplication occurs.

Copyright © 2010 Internetwork Expert

In order to avoid this situation, only one router is allowed to flood traffic on the shared segment. When a router detects that someone is sending traffic for the same source IP/destination group on the segment, where it has an active (S,G) state for the same group/source, it immediately originates a PIM Assert message. This message contains the source IP, the group and the path cost to the source. The path cost is a touple (AD, Metric), where AD is the administrative distance of the routing protocol used to look up the source IP, and Metric is that same protocol's metric to reach the source. The router with the best (lowest) AD value on the segment wins the assertion. If the ADs are equal, then metric is used as tie-breaker. If both the AD and the metric are the same, the router with the highest IP address wins. If another router on the segment receives the Assert message and determines that it loses the assertion, it will remove the (S,G) state on its interface and stop flooding traffic. However, if it sees itself as the winner, it will emit a superior PIM Assert message to inform the other router that it should stop flooding the traffic for this (S,G) pair.

Note that the PIM Assert procedure might be dangerous on NBMA interfaces. PIM will treat those interfaces as fully broadcast capable networks, even though not all nodes are capable of hearing each-other's broadcast messages. Imagine a hub-and-spoke Frame-Relay topology. If both the hub and the spoke start flooding the segment with the same multicast traffic, PIM Assert will occur. If for some reason the spoke should win, then the hub will stop sending its multicast traffic. However, all traffic coming from the spoke to the hub is NOT relayed back to the other spokes sharing the same segment, based on the RPF rule. Thus, all other spokes will effectively stop receiving the multicast traffic. To avoid this situation, either use PIM NBMA mode (described in a separate task) or make sure that the hub always wins the PIM Assert procedure.

In our scenario, R1 and R4 run different IGPs. This is not a well-designed multicast solution, as you may want all routers to be under the same IGP. In our case, under normal circumstances R4 would be the assert winner as EIGRP has a better AD (90) than OSPF (110). However, by adjusting the AD of OSPF on R1, we artificially make R1 the assert winner on the shared segment.

Copyright © 2010 Internetwork Expert

R1:

```
access-list 10 permit 224.0.0.0 0.255.255.255
!
ip multicast-routing
ip pim rp-address 150.1.5.5 10
!
interface FastEthernet0/0
 ip pim sparse-dense-mode
!
interface Serial 0/0.1
 ip pim sparse-dense-mode
!
router ospf 1
 distance 80
```

R5:

```
interface Serial 0/0/0
 ip pim sparse-dense-mode
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Join the group 239.6.6.6 on R6 and ping this IP address from SW4. The group will use simple dense-mode forwarding, and R1 will be elected as the assert winner on the VLAN 146 segment. Please note that we receive two responses for the first ping. This is due to the fact that both R1 and R4 initially try to source the traffic to R6. The PIM Assert Process stops this parallel forwarding of traffic when one router becomes the PIM Forwarder. In this case we "rigged" the election by assigning a better Administrative Distance to OSPF on R1.

```
R6:
interface FastEthernet 0/0.146
 ip igmp join-group 239.6.6.6

Rack1SW4#ping 239.6.6.6 repeat 1000

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 239.6.6.6, timeout is 2 seconds:

Reply to request 0 from 155.8.146.6, 36 ms
Reply to request 0 from 155.8.146.6, 76 ms
Reply to request 1 from 155.8.146.6, 72 ms
Reply to request 2 from 155.8.146.6, 72 ms
Reply to request 3 from 155.8.146.6, 72 ms
Reply to request 4 from 155.8.146.6, 72 ms

Rack6AS>1
```

> **Note**
>
> The mroute entry on R1 is marked with an "A" flag, which means "Assert Winner".

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip mroute 239.6.6.6
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.6.6.6), 00:02:05/stopped, RP 0.0.0.0, flags: DC
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 00:02:05/00:00:00
    Serial0/0.1, Forward/Sparse-Dense, 00:02:05/00:00:00

(155.1.108.10, 239.6.6.6), 00:00:22/00:02:46, flags: T
  Incoming interface: Serial0/0.1, RPF nbr 155.1.0.5
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 00:00:24/00:00:00, A
```

Note

If you were to enable the following debugging on R1 before sending pings from SW4, you can catch the PIM Assert message exchange between R1 and R4. Notice that R1 wins due to the better AD.

```
Rack1R1#debug ip pim

PIM(0): Received v2 Assert on FastEthernet0/0 from 155.1.146.4
PIM(0): Assert metric to source 155.1.108.10 is [90/2174976]
PIM(0): We win, our metric [80/20]
PIM(0): (155.1.108.10/32, 239.6.6.6) oif FastEthernet0/0 in Forward
state
<snip>
PIM(0): Send v2 Assert on FastEthernet0/0 for 239.6.6.6, source
155.1.108.10, metric [80/20]
PIM(0): Assert metric to source 155.1.108.10 is [80/20]
PIM(0): We win, our metric [80/20]
PIM(0): (155.1.108.10/32, 239.6.6.6) oif FastEthernet0/0 in Forward
state
```

Copyright © 2010 Internetwork Expert

Note

R4 prunes the interface where it loses the assert procedure from the OIL for the group 239.6.6.6

```
Rack1R4#show ip mroute 239.6.6.6
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.6.6.6), 00:07:12/stopped, RP 0.0.0.0, flags: DC
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/1/0, Forward/Sparse-Dense, 00:07:12/00:00:00
    FastEthernet0/1, Forward/Sparse-Dense, 00:07:12/00:00:00

(155.1.108.10, 239.6.6.6), 00:05:27/00:00:39, flags: PT
  Incoming interface: Serial0/1/0, RPF nbr 155.1.45.5
  Outgoing interface list:
    FastEthernet0/1, Prune/Sparse-Dense, 00:02:26/00:00:33
```

Note

What happens if the Assert Winner stops working? Well, unfortunately, the Assert "Loser" has no way of knowing the Assert "Winner" has failed and will wait three (3) minutes before timing out its pruned interface. Thus we face a worst case scenario of loss of traffic for three (3) minutes should the PIM Assert winner go down right after winning the election.

Copyright © 2010 Internetwork Expert

8.6 PIM Accept RP

- Ensure that SW2 and R5 will only accept (*,G) joins toward R5's Loopback 0 for groups 224.10.10.10 and 224.110.110.110

Configuration

> **Note**
>
> This is a security feature used by IOS routers to prevent unwanted RPs or groups from becoming active in the PIM SM multicast domain. When you configure **`ip pim accept-rp {rp-address | auto-rp} [access-list]`** the local router will only accept (*,G) Join/Prune messages towards the RP-address specified in the command. Additionally, if the access-list parameter is specified, the router will only accept Join/Prune messages for groups matching this access-list. Note that regular SPT joins are not affected by this configuration.
>
> In order to implement thorough security using this feature, you must configure every router on all potential paths to the RP, or simply configure your RP. Performing this on the RP alone, however will allow illegal joins across the network, but they will eventually be dropped at the RP.

R5:

```
ip access-list standard ALLOWED_GROUPS
 permit 224.10.10.10
 permit 224.110.110.110
!
ip pim accept-rp 150.1.5.5 ALLOWED_GROUPS
```

SW2:

```
ip access-list standard ALLOWED_GROUPS
 permit 224.10.10.10
 permit 224.110.110.110
!
ip pim accept-rp 150.1.5.5 ALLOWED_GROUPS
```

Copyright © 2010 Internetwork Expert

Verification

Note

Join SW4's VLAN 10 interface to the groups 224.10.10.10 and 224.11.11.11. Observe the console output on **SW2:**

```
Rack1SW2#
%PIM-6-INVALID_RP_JOIN: Received (*, 224.11.11.11) Join from 0.0.0.0
for invalid RP 150.1.5.5

%PIM-6-INVALID_RP_JOIN: Received (*, 224.0.1.40) Join from 0.0.0.0 for
invalid RP 150.1.5.5
```

Note

Notice that SW2 rejected joins toward R5's Loopback 0 for the invalid groups. If you are wondering what group 224.0.1.40 is it's the Auto-RP discovery group, which all routers join by default. Since we happened to specify an RP for the group range covering this particular one, every router attempts to join the shared tree for it.

Using the **`show ip mroute`** command you can observe that only group 224.10.10.10 has an mroute state corresponding to the shared tree.

```
Rack1SW2#show ip mroute 224.10.10.10
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report, Z - Multicast
Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 01:32:57/00:03:10, RP 150.1.5.5, flags: S
  Incoming interface: Vlan58, RPF nbr 155.1.58.5
  Outgoing interface list:
    Port-channel1, Forward/Sparse-Dense, 01:32:57/00:03:10

Rack1SW2#show ip mroute 224.11.11.11
Group 224.11.11.11 not found
```

Copyright © 2010 Internetwork Expert

8.7 PIM DR Election

- Ensure that R1 is responsible for multicast source registration with the RP for devices on the VLAN 146 segment.

Configuration

Note

A PIM Designated Router (DR) is elected on every multiple-access segment, i.e. every segment where multiple routers share the same medium/subnet. This election process is based on the highest priority and highest IP address – the router with the numerically higher value wins the election. This process is pre-emptive and every new router with a better priority will preempt the previous DR.

One purpose of a DR is signaling multicast delivery trees using PIM messages when it sees interested receivers on the shared segment by means of IGMP. Another purpose is registering active sources on the segment with the regional RP. When the DR hears multicast packets on the segment, it will check to determine if the destination group has an RP. If that's the case, the data packets are encapsulated into special PIM Register messages and sent to the RP. The RP will start forwarding them down the shared tree, if there are any active subscribers. At the same time, the RP will build a shortest-path tree towards the DR and send a PIM Register-Stop message to the DR to inform it that regular forwarding may start now. After this, the multicast traffic is delivered over the SPT.

Note that PIM Register messages are subject to RPF checks, as usual. If the Register message is received on a non-RPF interface, the check will fail.

R1:

```
interface FastEthernet0/0
 ip pim dr-priority 100
```

Verification

```
Rack1R1#show ip pim interface fastEthernet 0/0 detail
FastEthernet0/0 is up, line protocol is up
  Internet address is 155.1.146.1/24
  Multicast switching: fast
  Multicast packets in/out: 4624/811
  Multicast TTL threshold: 0
  PIM: enabled
    PIM version: 2, mode: sparse-dense
    PIM DR: 155.1.146.1 (this system)
    PIM neighbor count: 2
    PIM Hello/Query interval: 30 seconds
    PIM Hello packets in/out: 603/304
    PIM State-Refresh processing: enabled
    PIM State-Refresh origination: disabled
    PIM NBMA mode: disabled
    PIM ATM multipoint signalling: disabled
    PIM domain border: disabled
  Multicast Tagswitching: disabled
```

Copyright © 2010 Internetwork Expert

8.8 PIM Accept Register

- Configure R5 so that the only source allowed on the VLAN 146 segment is R6.
- Your configuration should not affect sources on any other segments.

Configuration

Note

This security feature is configured on the PIM SM Rendezvous Point and specifies the sources that are allowed to register with the RP. Remember that registration is performed by the PIM DR router, and every register message contains the original multicast packet which includes the IP address of the multicast source and the group destination address. If the RP denies the registration, it sends a PIM Register-Stop to the DR immediately and never builds the SPT towards the source.

The command to enable PIM Register message filtering is **`ip pim accept-register {list <Extended-ACL> | route-map <Route-Map>}`**. The basis of filtering is an extended ACL in the format:

```
ip access-list REGISTER_FILTER
 permit ip <source-ip> <source-wildcard> <group-address>
<group-wildcard>
```

In these access-lists we can match both sources and destination groups at the same time. For example, to permit the source 155.1.146.1 to send traffic to any group via the RP, use the entry `permit ip host 155.1.146.1 224.0.0.0 15.255.255.255`. Be advised that if the RP is the DR for the same segment, this filtering will not work.

For our scenario, we create an access-list applied with “negative” filtering in a route-map. The access-list permits all sources on VLAN 146 with the exception of R6. Later, all those sources are denied by the route-map. Everything else is permitted.

Copyright © 2010 Internetwork Expert

R5:

```
ip access-list extended VLAN146_SOURCES
 deny ip host 155.1.146.6 any
 permit ip 155.1.146.0 0.0.0.255 any
!
route-map ACCEPT_REGISTER deny 10
 match ip address VLAN146_SOURCES
!
route-map ACCEPT_REGISTER permit 100
!
ip pim accept-register route-map ACCEPT_REGISTER
```

Copyright © 2010 Internetwork Expert

Verification

Note

First of all, we join SW4 to group 224.10.10.10 and then try pinging it from R6. Since the IP address of R6 is permitted to register with the RP, everything goes smoothly.

```
SW4:
interface Vlan 10
 ip igmp join-group 224.10.10.10

Rack1R6#ping 224.10.10.10 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 224.10.10.10, timeout is 2 seconds:

Reply to request 0 from 155.1.108.10, 64 ms
Reply to request 1 from 155.1.108.10, 32 ms
Reply to request 2 from 155.1.108.10, 32 ms
Reply to request 3 from 155.1.108.10, 32 ms
```

Note

Now **temporarily** create interface vlan 146 on SW1 as follows:

```
SW1:

ip multicast-routing distributed

interface vlan 146
 ip address 155.1.146.7 255.255.255.0
 ip pim dense-mode
```

And enable PIM debugging on R1 (the DR) using the command **debug ip pim**. Notice that R1 receives a Register-Stop message immediately after sending the first Register packet. However, this time the RP does not build an SPT toward the source.

```
Rack1SW1#ping 224.10.10.10 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 224.10.10.10, timeout is 2 seconds:
...
Rack6AS>1
[Resuming connection 1 to r1 ... ]


Rack1R1#show logging
```

```
<snip>

PIM(0): Send v2 Register to 150.1.5.5 for 155.1.146.7, group
224.10.10.10
PIM(0): Received v2 Register-Stop on Serial0/0.1 from 150.1.5.5
PIM(0):   for source 155.1.146.7, group 224.10.10.10
PIM(0): Clear Registering flag to 150.1.5.5 for (155.12.146.7/32,
224.10.10.10)
```

Note

You may discover more on R5 – the RP. Notice the console message that warns you about an INVALID source. Also, there is no SPT entry toward the new VLAN 146 IP address of SW1 in the mroute table of R5. Thus, the source is not connected to the shared tree.

```
Rack1R5#
%PIM-4-INVALID_SRC_REG: Received Register from 155.1.0.1 for
(155.1.146.7, 224.10.10.10), not willing to be RP

Rack1R5#show ip mroute 224.10.10.10
IP Multicast Routing Table
<snip>

(*, 224.10.10.10), 05:39:10/00:02:52, RP 150.1.5.5, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 05:39:10/00:02:52

(155.1.146.6, 224.10.10.10), 00:02:34/00:01:07, flags: T
  Incoming interface: Serial0/1/0, RPF nbr 155.1.45.4
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 00:02:34/00:02:52
```

Note

Please observe that in this lab we avoided doing verification on R1 and R4 for the Accept-Register configuration. This is to avoid issues related to packets being flooded out all PIM enabled interfaces on these devices. In this particular configuration this situation leads to traffic being able hit R5 (the RP) and thus being able to flow down the shared tree.

Be sure to remove Interface VLAN 146 and ip multicast routing from SW1 before continuing to the next lab.

Copyright © 2010 Internetwork Expert

8.9 Multicast Tunneling

- Without configuring multicast routing on the transit nodes, make sure that receivers on SW3 may receive multicast feeds from R1.
- Do not configure any additional IGP adjacencies to accomplish this task.

Configuration

Note

Tunneling is a common technique used with multicast technologies in order to traverse non-multicast capable networks. This greatly simplifies configuration, but has the major drawback of "hiding" the real network topology and thus often times preventing effective multicast replication. When configuring a tunnel, you have an option of enabling or disabling the IGP on the tunnel interface. If you plan to use the tunnel for multicast traffic delivery, it is necessary to enable PIM on the same interface. Because of the following two factors, you may easily run into RPF failure issues.

1) If no IGP is running on the tunnel, then you will need to provide static mroutes to correct any RPF checks.

2) If IGP is running on the tunnel interface, the cost of traversing the tunnel might be higher than the cost of traversing the underlying network. It is generally not the best idea to run the same IGP on the underlying network and on the tunnel. You may need to adjust metrics or provide static mroutes to correct any resulting problems.

In our situation, we aren't running an IGP on the tunnel. Thus we must provide static mroute statements to allow the endpoints to perform RPF checks correctly. Also, we need to replicate the static RP configuration, to allow SW3 to receive traffic for sparse groups.

Copyright © 2010 Internetwork Expert

R1:

```
interface Tunnel 0
 ip unnumbered Loopback0
 tunnel source Loopback0
 tunnel destination 150.1.9.9
 ip pim sparse-dense-mode
!
router ospf 1
 passive-interface Tunnel0
```

SW3:

```
ip multicast-routing
!
interface Tunnel 0
 ip unnumbered Loopback0
 tunnel source Loopback0
 tunnel destination 150.1.1.1
 ip pim sparse-dense-mode
!
router ospf 1
 passive-interface Tunnel0
!
ip mroute 0.0.0.0 0.0.0.0 tunnel 0
!
ip access-list standard SPARSE_GROUPS
 permit 224.0.0.0 0.255.255.255
!
ip pim rp-address 150.1.5.5 SPARSE_GROUPS
```

Copyright © 2010 Internetwork Expert

Verification

Note

To verify your configuration, Join SW3's interface VLAN 9 to the multicast group 224.10.10.10 and ping this group from R1. Note that enabling PIM on the interface is mandatory, as otherwise IOS will not forward multicasts out of this interface.

SW3:

```
interface Vlan 9
 ip igmp join-group 224.10.10.10
 ip pim sparse-dense-mode
```

Notice that all RPF information on SW3 is condensed into one single default mroute. It overrides all checks using the unicast routing table and allows SW3 to receive multicasts successfully.

```
Rack1SW3#show ip mroute 224.10.10.10
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report, Z - Multicast
Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:01:24/00:02:41, RP 150.1.5.5, flags: SJCL
  Incoming interface: Tunnel0, RPF nbr 150.1.1.1, Mroute
  Outgoing interface list:
    Vlan9, Forward/Sparse-Dense, 00:01:24/00:02:41
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip mroute 224.10.10.10
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.10.10.10), 00:01:35/00:02:54, RP 150.1.5.5, flags: S
  Incoming interface: Serial0/0.1, RPF nbr 155.1.0.5
  Outgoing interface list:
    Tunnel0, Forward/Sparse-Dense, 00:01:35/00:02:54

Rack1R1#ping 224.10.10.10

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 224.10.10.10, timeout is 2 seconds:

Reply to request 0 from 150.1.9.9, 64 ms
Reply to request 0 from 155.1.108.10, 104 ms
Reply to request 0 from 150.1.9.9, 88 ms
```

Copyright © 2010 Internetwork Expert

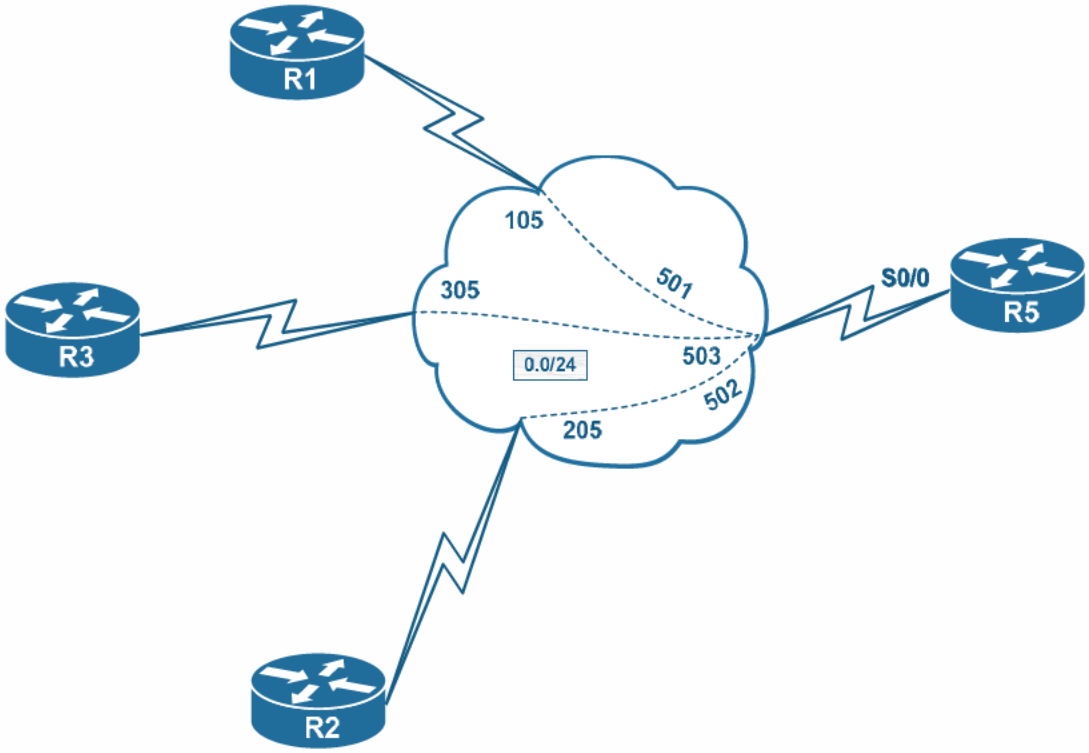
8.10 PIM NBMA Mode

- Configure R3 to join the multicast group 224.110.110.110 and make sure R6 can ping it.

Configuration

Note

PIM NBMA mode is an extension to PIM protocol operations on WAN interfaces, particularly for Frame-Relay and ATM which utilize the concept of virtual-circuits.



Look at the figure above. First, imagine that R5 is forwarding multicast traffic down to R1, R2 and R3. By default, all multicast packets are treated as broadcast on the Frame-Relay segment, and queued in the special broadcast queue. This queue is a priority queue, used to store the routing protocol updates and L2 keepalives. Packets from this queue are replicated and sent over all active VCs terminating on the interface. This is called “pseudo-broadcast” and is performed at the router’s CPU level. Needless to say, this procedure is very CPU intensive when multicast flows are intensive. Not to mention that it may also block critical routing updates on the interface.

Copyright © 2010 Internetwork Expert

Secondly, PIM treats the Frame-Relay interface as a “broadcast” media. That is, it assumes that all multicast packets received on this interface are heard by all members connected to the cloud. However, Frame-Relay is a non-broadcast technology, and it only works like broadcast in case of a fully meshed topology. If you look at the figure above, you will see that the hub-and-spoke topology does not satisfy this requirement. When R1 sends a multicast packet, only R5 receives it. Based on the RPF rule, it will never send the packet back out of the same interface. Thus, any multicast packets sent from one spoke will never reach other spokes. If you want to resolve this issue, you have to configure sub-interfaces on R5 for all spokes.

The other issue that comes from the NBMA nature of Frame-Relay technology is that spokes don’t hear each other’s PIM messages. For example, if R1, R2 and R3 all have multicast subscribers, they all build SPT trees across R5. At R5, all the trees converge to a single (S,G) state. Now suppose that R1 loses its last receiver, and sends a PIM Prune message to R5. Since R5 assumes that all other routers also heard this message, it declares that the shared segment is free of any receivers. After this, R5 prunes the (S,G) state and leaves R2 and R3 without multicast traffic. Both R2 and R3 never had a chance to override the PIM Prune message sent from R1, as they never heard it.

PIM NBMA mode resolves all of the above issues. First of all, it works only with PIM SM (sparse mode groups), as it relies on PIM Join messages to function. Secondly, it treats the “monolithic” Frame-Relay interface as a collection of point-to-point links connected to other nodes. It does so by tracking the PIM Join messages received from those neighbors. For every neighbor that sends a Join message the router creates a separate (S,G) state bound to this neighbor’s IP address. Now this state emulates a point-to-point link connected to this neighbor. All PIM Prune messages received from this neighbor will only affect its own (S,G) state, leaving other neighbors’ states intact. Also, any multicast traffic received from this neighbor is treated like it has been received from a point-to-point link. Thus, the RPF procedure will allow forwarding the multicast packets back out of the same interface to all other interested neighbors. Finally, PIM NBMA mode replaces the ineffective pseudo-broadcast replication on all PVCs with fast-switched packet distribution only to the PVCs with subscribed neighbors. This saves router CPU resources and WAN bandwidth.

Notice that you may enable NBMA mode on a PIM SM/DM interface, even though this is not the recommended configuration. It will only work for groups signaled via sparse mode.

Copyright © 2010 Internetwork Expert

R3:

```
ip multicast-routing
!
ip access-list standard SPARSE_GROUPS
 permit 224.0.0.0 0.255.255.255
!
ip pim rp-address 150.1.5.5 SPARSE_GROUPS
!
interface Serial 1/0.1
 ip pim sparse-dense-mode
!
interface Loopback0
 ip pim sparse-dense-mode
 ip igmp join-group 224.110.110.110
```

R5:

```
interface Serial 0/0/0
 ip pim nbma-mode
```

Copyright © 2010 Internetwork Expert

Verification

Note

First we are going to temporarily disable the serial link between R4 and R5 to prevent R5 from building a SPT to R6 over that link. With this serial link down R1 the, DR for the VLAN 146 segment, will register R6 with R5, the RP. R5 will then build the SPT to R6 across the FR cloud. As you have probably already noticed there is an additional situation we must deal with - by default R3 will prefer R5's Loopback via SW1. To correct this, we will lower the metric on R3's Frame Relay interface to a value of 10, to ensure R3 builds the SPT towards the RP correctly.

R4:

```
interface Serial 0/1/0
 shut
```

R3:

```
interface Serial 1/0.1
 ip ospf cost 10
```

Now ping the group from R6 before you apply the NBMA mode configuration. The operation will fail, and you can see that the OIL for the group on R5 is Null. Note that the multicast traffic has to go out the Serial 0/0/0 interface of R5 to reach R3, but it can't because it arrived on S0/0/0 from R1 thus the OIL is "empty" or Null.

```
Rack1R6#ping 224.110.110.110 repeat 100

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 224.110.110.110, timeout is 2
seconds:
...
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#show ip mroute 224.110.110.110
IP Multicast Routing Table
<snip>

(*, 224.110.110.110), 00:00:27/00:02:57, RP 150.1.5.5, flags: SJCL
  Incoming interface: Serial1/0.1, RPF nbr 155.1.0.5
  Outgoing interface list:
    Loopback0, Forward/Sparse-Dense, 00:00:27/00:02:57

Rack1R5#show ip mroute 224.110.110.110
IP Multicast Routing Table
<snip>
(*, 224.110.110.110), 00:00:55/stopped, RP 150.1.5.5, flags: SPX
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0/0, Forward/Sparse-Dense, 00:00:44/00:00:00

(155.1.146.6, 224.110.110.110), 00:00:38/00:02:21, flags: PT
  Incoming interface: Serial0/0/0, RPF nbr 155.1.0.1
  Outgoing interface list: Null
```

Note

After you enable PIM NBMA mode on the hub-and-spoke interface, notice the change in mroute states. Now there are NBMA IP addresses next to the interface name, tracking the state of sparse neighbors. Since the interface is now treated as an array of point-to-point links, the split-horizon rule is no longer in effect. Pings from R6 will now succeed.

Please note that it may be necessary to execute a "clear ip mroute *" on the hub router in order to get the show commands to match the ones in the documentation. It is not uncommon to initially see a value of 224.0.0.2 in the show command output for **`show ip mroute`** when dealing with NBMA-MODE configuration. This can easily be corrected with the clear command we mentioned or waiting for the process to refresh on its own.

```
Rack1R5#show ip mroute 224.110.110.110
IP Multicast Routing Table
<snip>

(*, 224.110.110.110), 00:02:58/00:03:28, RP 150.1.5.5, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0/0, 155.1.0.3, Forward/Sparse-Dense, 00:00:01/00:03:28

(155.1.146.6, 224.110.110.110), 00:02:41/00:02:54, flags: T
  Incoming interface: Serial0/1/0, RPF nbr 155.1.45.4
  Outgoing interface list:
    Serial0/0/0, 155.1.0.3, Forward/Sparse-Dense, 00:00:01/00:03:28
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#ping 224.110.110.110 repeat 1000

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 224.110.110.110, timeout is 2
seconds:

Reply to request 0 from 155.1.0.3, 65 ms
Reply to request 1 from 155.1.0.3, 60 ms
Reply to request 2 from 155.1.0.3, 60 ms
Reply to request 3 from 155.1.0.3, 60 ms
```

Note

Remember to remove the temporary modifications made to R3, and R4.

Copyright © 2010 Internetwork Expert

8.11 Auto-RP

- Remove all static RP settings and previous Accept-RP/Register configurations.
- Using a Cisco proprietary technology, configure R5 so that it advertises itself as the RP for all multicast groups.

Configuration

Note

Auto-RP was the first protocol to automatically distribute RP information across the multicast domain when using PIMv1. Auto-RP is a Cisco proprietary protocol, and it was later "replaced" by the standards-based Bootstrap Router (BSR) protocol with the advent of PIMv2. However, all Cisco routers still support Auto-RP along with PIMv2 protocol.

Auto-RP defines two new concepts – Candidate RP (cRP) and Mapping Agents (MA). cRP is any router that is willing to become an RP. You configure this router using the command **`ip pim send-rp-announce <Interface> scope <TTL> [group-list <Std-ACL>] [interval <seconds>]`**. The router will start sending UDP packets to the IP/Port 224.0.1.39/496 with the list of groups serviced by this particular RP. The cRP announcements are originated every 60 seconds by default, with the Time-to-Live field in the IP headers set to **`<TTL>`** – a form of administrative scoping. The interface that you specify in this command must be enabled for PIM and its IP address will be used as the RP's IP address. The list of groups is defined using the **`<Std-ACL>`** access-list. You configure this list using standard access-list syntax, e.g.

```
access-list GROUPS permit 239.0.0.0 0.0.0.255
access-list GROUPS permit 232.0.2.0 0.0.0.255
access-list GROUPS deny 224.0.1.50 0.0.0.0
```

The Auto-RP code will convert wildcard masks into the prefix-lengths, so you cannot use discontinuous masks. Next, the "deny" statements are interpreted as a special type of *negative* group announcement. This means that all groups matching this range should be treated as "dense" mode groups. We will discuss how the routers interpret the mapping entries a bit later.

Copyright © 2010 Internetwork Expert

The cRP announcements are flooded across the network and reach special routers called Mapping Agents. You configure these routers using the command **ip pim send-rp-discovery <Interface> scope <TTL> interval <Seconds>**. Mapping Agents listen on the standard address 224.0.1.39 and collect all announcements from candidate RPs. After that, every mapping agent compiles a resulting list of Group to RP mappings and starts sending "RP discovery" messages to the special multicast address 224.0.1.40 port 496. The discovery messages contain an amalgam of all information learned by the mapping agents. Notice that if there are multiple MAs in the network, they will hear each other, and then all of them, except the one with the highest IP address, will cease sending discoveries.

When building a discovery message, an MA will follow a few simple rules:

1) If there are two announcements with the same group range but different RPs, the MA will select the announcement with the highest RP IP address.
2) If there are two announcements, where one group is a subset of another, but the RPs are different, both will be sent.
3) All other announcements are grouped together without any conflict resolution.

All regular routers join the multicast group 224.0.1.40 and listen to the discovery messages. Based on their content, they populate their Auto-RP cache and learn about Group-to-RP mappings. The cache contains both "negative" and "positive" entries. When looking for an RP, Auto-RP code will first scan through negative entries. If a match is found, the group is considered to be dense. Note that RP information for the negative entries will be effectively ignored. If the group is not found in the "negative" list, the code looks it up in the positive list. Since every group in the list is bound to a particular RP, there could be conflicts when multiple RPs try to service overlapping group ranges. The receiving router uses the longest-match rule to resolve all conflicts: if there are multiple matches, only the one with the longest prefix length is selected.

Note that "negative" statements could be defined at any cRP and affect ALL routers in the multicast region. For example if a single group list would contain **deny any** statement, then all groups will be treated as dense, even if there are "positive" entries.

Copyright © 2010 Internetwork Expert

The last thing to discuss about Auto-RP is how the multicast groups 224.0.1.39 and 224.0.1.40 are propagated across the network. Since there is no explicit RP information for these groups, they must use dense mode forwarding. This requires the use of **pim sparse-dense-mode** on all interfaces within the multicast domain. As it has been discussed before, this is not the safest thing to do in a large-scale network. You may want to use the **no ip dm-fallback** global command in such situations or use the Auto-RP Listener feature, discussed in a separate task.

Note that you may use PIM SM mode along with Auto-RP if you define a static RP value for the Auto-RP groups (224.0.1.39 and 224.0.1.40). This will require you to use the **override** option when defining the static RP. By default, Auto-RP announcements override a statically configured RP. If you want them to persist, use the **override** keyword along with the **ip pim rp-address** command.

You may be asking one question – why is there a need for a Mapping Agent? Couldn't candidate RPs just broadcast themselves to all routers and the latter learn/elect the best RPs directly? This is possible, but might result in different routers electing different RPs for the same group ranges. Some routers may miss announcements of a particular candidate RP due to network outages or RP failures. Therefore, cRP information should be collected at one single point before being disseminated to the multicast routers.

R1, R3, R4, R5, R6, SW2, SW3, SW4:

```
no ip pim rp-address 150.1.5.5
```

R5:

```
no ip pim accept-register route-map ACCEPT_REGISTER
no ip pim accept-rp 150.1.5.5 ALLOWED_GROUPS
!
interface Loopback0
 ip pim sparse-dense-mode
!
ip pim send-rp-announce Loopback 0 scope 10
ip pim send-rp-discovery loopback 0 scope 10
```

SW2:

```
no ip pim accept-rp 150.1.5.5 ALLOWED_GROUPS
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Initial verification includes checking for RP to group mappings. In our case, there is just one RP, thus all groups map to it. Make sure you perform the verification on all routers, as some of them might reject discovery announcements due to RPF issues.

```
Rack1R5#show ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP (Auto-RP)
This system is an RP-mapping agent (Loopback0)

Group(s) 224.0.0.0/4
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.5.5 (?), elected via Auto-RP
         Uptime: 00:00:34, expires: 00:02:24
```

> **Note**
>
> If you are wondering what the (?) sign means, it is for the hostname of the RP. IOS will display the hostname only if the router is configured to do DNS name lookup but the IP address of the RP also has to resolve to a name.

Copyright © 2010 Internetwork Expert

8.12 Auto-RP - Multiple Candidate RPs

- Configure SW2 and SW4 as RPs for group ranges 224.0.0.0-231.255.255.255 and 232.0.0.0-239.255.255.255 respectively.
- In case one RP fails, the other one should provide backup for the other's groups.
- The group 224.110.110.110 should be always switched in dense-mode.

Configuration

Note

As discussed in the previous task, Auto-RP MAs amalgamate information learned from multiple candidate RPs and advertise RP discovery messages. You may want to use multiple RPs in cases where you want load-balancing, redundancy or both.

1) If your goal is load balancing, try to configure the group-mapping access-lists so that every RP services a range of groups.
2) If your aim is redundancy, make both RPs service the same group ranges. The one with the highest IP address will be selected by the MA.
3) If you want to achieve both load-balancing and redundancy, you may map RP1 to a specific group range, say 224.0.0.0-231.255.255.255 and permit 224.0.0.0 15.255.255.255 in the end of the respective ACL. Use the entry to permit the range 232.0.0.0-239.255.255.255 for RP2 along with the entry 224.0.0.0 15.255.255.255 in the end. This will ensure that RP1 and RP2 are used for specific group ranges, while RP1 is used for the rest of the groups when RP2 fails or vice versa. This is based on the longest match selection criteria used by the multicast routers and the fact that for overlapping ranges the MA will only advertise the RP with the highest IP address.

Notice that the mapping list is compiled by the MA, but still the final RP selection is performed by the multicast router. Longest match criteria is an effective rule to allow for unique RP selection along with providing redundancy.

Copyright © 2010 Internetwork Expert

R5:
```
no ip pim send-rp-announce Loopback 0 scope 10
```

SW2:
```
ip access-list standard SW2_GROUPS
 permit 224.0.0.0 7.255.255.255
 permit 224.0.0.0 15.255.255.255
 deny 224.110.110.110
!
interface Loopback0
 ip pim sparse-dense-mode
!
ip pim send-rp-announce Loopback 0 scope 10 group-list SW2_GROUPS
```

SW4:
```
no ip access-list standard SW4_GROUPS
ip access-list standard SW4_GROUPS
 permit 232.0.0.0 7.255.255.255
 permit 224.0.0.0 15.255.255.255
 deny 224.110.110.110
!
interface Loopback0
 ip pim sparse-dense-mode
!
ip pim send-rp-announce Loopback 0 scope 10 group-list SW4_GROUPS
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Look at the RP mappings on R5. Notice how the MA elects the best candidate RP for a given range. The group 224.110.110.110 is negatively cached, and thus is always processes in dense-mode.

```
Rack1R5#show ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP-mapping agent (Loopback0)

Group(s) 224.0.0.0/5
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:01:15, expires: 00:02:42
Group(s) 224.0.0.0/4
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.10.10 (?), elected via Auto-RP
         Uptime: 00:01:05, expires: 00:02:51
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.8.8 (?), via Auto-RP
         Uptime: 00:01:15, expires: 00:02:42
Group(s) (-)224.110.110.110/32
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.10.10 (?), elected via Auto-RP
         Uptime: 00:01:05, expires: 00:02:53
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.8.8 (?), via Auto-RP
         Uptime: 00:01:15, expires: 00:02:45
Group(s) 232.0.0.0/5
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.10.10 (?), elected via Auto-RP
         Uptime: 00:01:05, expires: 00:02:51
```

Copyright © 2010 Internetwork Expert

Note

Check the Auto-RP cache of R6. Notice that it only has a single RP for every range. SW4 is elected as the RP for all ranges except 224.0.0.0/5.

```
Rack1R6#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/5
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.5.5 (?), elected via Auto-RP
         Uptime: 00:02:16, expires: 00:02:49
Group(s) 224.0.0.0/4
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.5.5 (?), elected via Auto-RP
         Uptime: 00:02:06, expires: 00:02:49
Group(s) (-)224.110.110.110/32
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.5.5 (?), elected via Auto-RP
         Uptime: 00:02:06, expires: 00:02:50
Group(s) 232.0.0.0/5
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.5.5 (?), elected via Auto-RP
         Uptime: 00:05:06, expires: 00:02:54
```

Note

Now check to see if the group 224.110.110.110 is forwarded using PIM Dense mode. Recall that R3 has joined this group before, and ping it from SW2. However, make sure you temporality shut down R1's Frame-Relay interface. Otherwise, R1 will send PIM Prune message to R5 and R5 will remove the whole interface from OIL for the group. You cannot override this behavior with PIM DM, unless you use sub-interfaces.

R1:

```
interface Serial 0/0
 shutdown
```

Notice that the group has no RP in the mroute output, just as it is supposed to be for the dense group.

Copyright © 2010 Internetwork Expert

```
Rack1SW2#ping 224.110.110.110 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 224.110.110.110, timeout is 2
seconds:

Reply to request 0 from 155.1.0.3, 75 ms
Reply to request 1 from 155.1.0.3, 67 ms
Reply to request 2 from 155.1.0.3, 67 ms
Reply to request 3 from 155.1.0.3, 67 ms
Reply to request 4 from 155.1.0.3, 68 ms
Reply to request 5 from 155.1.0.3, 67 ms
Reply to request 6 from 155.1.0.3, 67 ms

Rack1R5#show ip mroute 224.110.110.110
IP Multicast Routing Table
<snip>

(*, 224.110.110.110), 00:00:24/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0/0, Forward/Sparse-Dense, 00:00:24/00:00:00
    Serial0/1/0, Forward/Sparse-Dense, 00:00:24/00:00:00
    FastEthernet0/0, Forward/Sparse-Dense, 00:00:24/00:00:00

(155.1.58.8, 224.110.110.110), 00:00:24/00:02:51, flags: T
  Incoming interface: FastEthernet0/0, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0/0, Forward/Sparse-Dense, 00:00:25/00:00:00
    Serial0/1/0, Forward/Sparse-Dense, 00:00:25/00:00:00
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

8.13 Auto-RP - Filtering Candidate RPs

- Configure R5 to filter announcements for the group 224.110.110.110 received from SW4.

Configuration

Note

The Auto-RP mapping agent allows for the filtering of incoming RP announcements sent by RP candidates. This type of filtering applies only to Auto-RP announcements received by listening to the 224.0.1.39 multicast IP address and thus is only effective on Mapping Agents. You configure a filtering statement using the following syntax: **`ip pim rp-announce-filter {group-list <access-list> | rp-list <access-list> [group-list <access-list>]}`**. All access-lists are either numbered or named standard ACLs. RP announcements are inspected and if a match is found for the group and the RP IP address, the action is taken based on the "permit" or "deny" statement.

If you omit the **`rp-list`** keyword, then any announcements containing groups matching the **`group-list`** are matched. If you omit the **`group-list`** parameter, then all updates from the RPs in **`rp-list`** are matched.

R5:

```
ip access-list standard RP_LIST
 permit 150.1.10.10
!
ip access-list standard GROUP_LIST
 deny 224.110.110.110
 permit any
!
ip pim rp-announce-filter rp-list RP_LIST group-list GROUP_LIST
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Use the command **debug ip pim auto-rp** on the MA to monitor the updates that are filtered.

```
Rack1R5#debug ip pim auto-rp
PIM Auto-RP debugging is on
Auto-RP(0): Received RP-announce, from 150.1.10.10, RP_cnt 1, ht 181
Auto-RP(0): Filtered -224.110.110.110/32 for RP 150.1.10.10
Auto-RP(0): Update (232.0.0.0/5, RP:150.1.10.10), PIMv2 v1
Auto-RP(0): Update (224.0.0.0/4, RP:150.1.10.10), PIMv2 v1
Auto-RP(0): Received RP-announce, from 150.1.10.10, RP_cnt 1, ht 181
Auto-RP(0): Filtered -224.110.110.110/32 for RP 150.1.10.10
Auto-RP(0): Update (232.0.0.0/5, RP:150.1.10.10), PIMv2 v1
Auto-RP(0): Update (224.0.0.0/4, RP:150.1.10.10), PIMv2 v1
```

> **Note**
>
> Check to verify that the Auto-RP cache on the MA does not have the entry for 224.110.110.110 mapped to the RP.

```
Rack1R5#show ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP-mapping agent (Loopback0)

Group(s) 224.0.0.0/5
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:07:14, expires: 00:02:44
Group(s) 224.0.0.0/4
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.10.10 (?), elected via Auto-RP
         Uptime: 00:07:04, expires: 00:02:55
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.8.8 (?), via Auto-RP
         Uptime: 00:07:14, expires: 00:02:41
Group(s) (-)224.110.110.110/32
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:07:14, expires: 00:02:44
Group(s) 232.0.0.0/5
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.10.10 (?), elected via Auto-RP
         Uptime: 00:07:04, expires: 00:02:55
```

Copyright © 2010 Internetwork Expert

8.14 Auto-RP Listener

- Switch from PIM SM/DM on all interfaces to PIM SM, except for the loopback 0, and Tunnel 0 interfaces.
- Ensure that the S0/0 interface of R1 is not shutdown.
- Ensure all devices still hear Auto-RP announcements.

Configuration

> **Note**
>
> Auto-RP listener is another solution designed to allow using Auto-RP without risking any groups falling back to dense mode forwarding. This feature works in tandem with PIM sparse mode enabled on all interfaces (not PIM SM/DM). However, two Auto-RP multicast groups 224.0.1.39 and 224.0.1.40 are flooded in dense mode. No other groups are allowed to use dense mode and thus dangerous flooding fallback is eliminated.

R1:
```
interface FastEthernet 0/0
 ip pim sparse-mode
!
interface Serial 0/0.1
 ip pim sparse-mode
!
ip pim autorp listener
```

R3:
```
interface Serial 1/0.1
 ip pim sparse-mode
!
ip pim autorp listener
```

R4:
```
interface FastEthernet 0/1
 ip pim sparse-mode
!
interface Serial 0/1/0
 ip pim sparse-mode
!
ip pim autorp listener
```

R5:
```
interface FastEthernet 0/0
 ip pim sparse-mode
!
interface Serial 0/0/0
 ip pim sparse-mode
!
interface Serial 0/1/0
 ip pim sparse-mode
```

Copyright © 2010 Internetwork Expert

```
!
ip pim autorp listener
```

R6:

```
interface FastEthernet 0/0.146
 ip pim sparse-mode
!
ip pim autorp listener
```

SW2:

```
interface Vlan 58
 ip pim sparse-mode
!
interface Port-Channel 1
 ip pim sparse-mode
!
ip pim autorp listener
```

SW4:

```
interface Port-Channel 1
 ip pim sparse-mode
!
ip pim autorp listener
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Make sure that all transit interfaces are configured for PIM SM. Repeat the following operation on all routers and pay attention to the Ver/Mode field.

```
Rack1R5#show ip pim interface

Address          Interface                Ver/   Nbr    Query  DR
DR                                        Mode   Count  Intvl  Prior
155.1.45.5       Serial0/1/0              v2/S   1      30     1
0.0.0.0
155.1.58.5       FastEthernet0/0          v2/S   1      30     1
155.1.58.8
155.1.0.5        Serial0/0/0              v2/S   2      30     1
155.1.0.5
150.1.5.5        Loopback0                v2/SD  0      30     1
150.1.5.5

Rack1R1#show ip pim autorp
AutoRP Information:
  AutoRP is enabled.
  AutoRP groups over sparse mode interface is enabled

PIM AutoRP Statistics: Sent/Received
  RP Announce: 0/0, RP Discovery: 0/41
```

> **Note**
>
> Ensure that all routers are still capable of learning Auto-RP information.

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/5
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.5.5 (?), elected via Auto-RP
         Uptime: 00:07:37, expires: 00:02:45
Group(s) 224.0.0.0/4
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.5.5 (?), elected via Auto-RP
         Uptime: 00:25:33, expires: 00:02:46
Group(s) (-)224.110.110.110/32
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.5.5 (?), elected via Auto-RP
         Uptime: 00:21:33, expires: 00:02:47
Group(s) 232.0.0.0/5
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.5.5 (?), elected via Auto-RP
         Uptime: 00:09:43, expires: 00:02:46
```

Note

Check that the Auto-RP groups (224.0.1.39 and 224.0.1.40) are still forwarded without any RP information. Observe that the output shows Forward/Sparse, but the actual forwarding uses dense mode.

```
Rack1R1#show ip mroute 224.0.1.40
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.0.1.40), 00:20:40/stopped, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Tunnel0, Forward/Sparse-Dense, 00:20:40/00:00:00
    Serial0/0.1, Forward/Sparse, 00:20:40/00:00:00
    FastEthernet0/0, Forward/Sparse, 00:20:40/00:00:00

(150.1.5.5, 224.0.1.40), 00:19:46/00:02:40, flags: LT
  Incoming interface: Serial0/0.1, RPF nbr 155.1.0.5
  Outgoing interface list:
    Tunnel0, Forward/Sparse-Dense, 00:19:47/00:00:00
    FastEthernet0/0, Forward/Sparse, 00:19:47/00:00:00, A
```

Copyright © 2010 Internetwork Expert

8.15 Auto-RP and RP/MA Placement

- Make R1 another MA that preempts the existing MA on R5.
- Provide a solution for R3 to hear Auto-RP discovery messages from R1.

Configuration

Note

As discussed previously, PIM by default treats NBMA interfaces as if they were broadcast-capable. That is, it assumes that all neighbors on a WAN cloud can hear multicast packets sent by any neighbor. However, in non-fully meshed topologies such as hub-and-spoke networks, this is not the case. When a spoke sends multicast traffic over the NBMA segment, only the hub can hear it. The hub will not forward multicast back to other spokes, based on the split-horizon rule. This problem could be solved using PIM NBMA mode, but this only works with PIM Sparse mode.

Now the real issue is that the Auto-RP uses dense mode for RP information dissemination, and thus the PIM NBMA solution will not work. If a candidate RP or a Mapping Agent is placed behind the spoke node, then RP information could be lost. Imagine that an MA is located behind the spoke node. Then any Auto-RP discovery messages it sends will only reach the hub node, but not the other spokes. This could also be the case, if the RP and the MA are both placed behind NBMA spokes. Therefore, when designing your multicast network, take care and place the MA behind the hub. RPs could still be located at spokes, as long as announcements can reach the hub.

Other solutions include the use of sub-interfaces on the hub router or creating tunnels between the hub and the spokes. Both solutions effectively break the split-horizon rule, by receiving and sending multicast packets on different interfaces. Notice that you will need static mroutes if you don't run an IGP across the tunnel interfaces.

Copyright © 2010 Internetwork Expert

R1:

```
interface Loopback0
 ip pim sparse-mode
!
ip pim send-rp-discovery loopback 0 scope 10
!
interface tunnel 1
 tunnel source Loopback0
 tunnel destination 150.1.3.3
 ip unnumbered Loopback0
 ip pim sparse-mode
!
router ospf 1
 passive-interface Tunnel1
```

R3:

```
interface tunnel 1
 tunnel source Loopback0
 tunnel destination 150.1.1.1
 ip unnumbered Loopback0
 ip pim sparse-mode
!
! Static mroute is needed to allow R3 hearing
! Auto-RP discoveries from R1.
!
ip mroute 150.1.1.1 255.255.255.255 Tunnel 1
!
router ospf 1
 passive-interface Tunnel1
```

R5:

```
no ip pim send-rp-discovery loopback 0 scope 10
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> First, make sure that R1 is fully functional as an MA. It should receive announcements from both SW2 and SW4. Check the multicast routing table for the group 224.0.1.39 for this and ensure that the RPF neighbor is R5.

```
Rack1R1#show ip mroute 224.0.1.39
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.0.1.39), 00:02:34/stopped, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Loopback0, Forward/Sparse, 00:02:34/00:00:00
    Tunnel0, Forward/Sparse-Dense, 00:02:34/00:00:00
    Serial0/0.1, Forward/Sparse, 00:02:34/00:00:00
    Tunnel1, Forward/Sparse, 00:02:34/00:00:00
    FastEthernet0/0, Forward/Sparse, 00:02:34/00:00:00

(150.1.8.8, 224.0.1.39), 00:02:08/00:00:56, flags: LT
  Incoming interface: Serial0/0.1, RPF nbr 155.1.0.5
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:02:08/00:00:00
    Tunnel0, Forward/Sparse-Dense, 00:02:08/00:00:00
    Loopback0, Forward/Sparse, 00:02:08/00:00:00
    Tunnel1, Forward/Sparse, 00:02:08/00:00:00

(150.1.10.10, 224.0.1.39), 00:02:18/00:00:46, flags: LT
  Incoming interface: Serial0/0.1, RPF nbr 155.1.0.5
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:02:18/00:00:00, A
    Tunnel0, Forward/Sparse-Dense, 00:02:20/00:00:00
    Loopback0, Forward/Sparse, 00:02:20/00:00:00
    Tunnel1, Forward/Sparse, 00:02:20/00:00:00
```

Copyright © 2010 Internetwork Expert

Note

Check that the Auto-RP cache is populated with the announcements from SW2 and SW4.

```
Rack5R1#sh ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP-mapping agent (Loopback0)

Group(s) 224.0.0.0/5
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:04:20, expires: 00:02:40
Group(s) 224.0.0.0/4
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.10.10 (?), elected via Auto-RP
         Uptime: 00:03:55, expires: 00:02:03
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.8.8 (?), via Auto-RP
         Uptime: 00:04:20, expires: 00:02:39
Group(s) (-)224.110.110.110/32
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.10.10 (?), elected via Auto-RP
         Uptime: 00:03:55, expires: 00:02:01
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.8.8 (?), via Auto-RP
         Uptime: 00:04:20, expires: 00:02:41
Group(s) 232.0.0.0/5
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.10.10 (?), elected via Auto-RP
         Uptime: 00:03:55, expires: 00:02:00
```

Note

Next, verify that R5 can hear Auto-RP discoveries from R1. Notice that the OIL for the group 224.0.1.40 does not contain the Frame-Relay interface, and thus R3 cannot hear the discovery messages relayed via R5.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip mroute 224.0.1.40
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.0.1.40), 00:13:04/stopped, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0/0, Forward/Sparse, 00:13:04/00:00:00
    Loopback0, Forward/Sparse-Dense, 00:13:04/00:00:00
    Serial0/1/0, Forward/Sparse, 00:13:04/00:00:00
    FastEthernet0/0, Forward/Sparse, 00:13:04/00:00:00

(150.1.1.1, 224.0.1.40), 00:02:30/00:02:23, flags: LT
  Incoming interface: Serial0/0/0, RPF nbr 155.1.0.1
  Outgoing interface list:
    Serial0/1/0, Forward/Sparse, 00:02:30/00:00:00
    Loopback0, Forward/Sparse-Dense, 00:02:30/00:00:00
    FastEthernet0/0, Forward/Sparse, 00:02:30/00:00:00
```

Note

Check the Auto-RP information on R3 after you have implemented the tunnel workaround. Notice that this workaround will not work without the static mroute. The tunnel is used only to deliver the Auto-RP discoveries to R3, while regular multicast will take paths across the physical links.

Copyright © 2010 Internetwork Expert

```
Rack1R3#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/5
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.1.1 (?), elected via Auto-RP
         Uptime: 00:00:23, expires: 00:02:33
Group(s) 224.0.0.0/4
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.1.1 (?), elected via Auto-RP
         Uptime: 00:00:23, expires: 00:02:33
Group(s) (-)224.110.110.110/32
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.1.1 (?), elected via Auto-RP
         Uptime: 00:00:23, expires: 00:02:33
Group(s) 232.0.0.0/5
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.1.1 (?), elected via Auto-RP
         Uptime: 00:00:23, expires: 00:02:36

Rack1R3#show ip mroute 224.0.1.40
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.0.1.40), 01:11:20/stopped, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Tunnel1, Forward/Sparse, 00:05:59/00:00:00
    Loopback0, Forward/Sparse, 00:43:53/00:00:00
    Serial1/0.1, Forward/Sparse, 01:11:20/00:00:00

(150.1.1.1, 224.0.1.40), 00:02:28/00:02:38, flags: LT
  Incoming interface: Tunnel1, RPF nbr 150.1.1.1, Mroute
  Outgoing interface list:
    Serial1/0.1, Forward/Sparse, 00:02:30/00:00:00
    Loopback0, Forward/Sparse, 00:02:30/00:00:00
```

Copyright © 2010 Internetwork Expert

8.16 Filtering Auto-RP Messages

- Configure R1 so that SW4 cannot hear Auto-RP discovery messages.

Configuration

> **Note**
>
> Auto-RP uses special dense-mode groups to disseminate RP information, and the information is flooded across the multicast domain. The only way to control the scope of the information is by setting the TTL to the value that roughly represents the diameter of the multicast domain. However, the problem is that this does not enforce strict administrative limits on the information scope, and per-link flooding is not controlled. The other way to filter Auto-RP announcements is to use the **`ip multicast boundary`** command, discussed in a separate task.

```
R1:
ip pim send-rp-discovery Loopback0 scope 2
```

Verification

> **Note**
>
> Ensure that SW2 still has entries in the Auto-RP cache while SW4 does not. Please note that it may take as long as three minutes for these mappings to expire from the rp mapping table.

```
Rack1SW4#show ip pim rp ma
PIM Group-to-RP Mappings
This system is an RP (Auto-RP)

Rack1SW4#
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show ip pim rp ma
PIM Group-to-RP Mappings
This system is an RP (Auto-RP)

Group(s) 224.0.0.0/5
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.1.1 (?), elected via Auto-RP
         Uptime: 00:32:32, expires: 00:01:58
Group(s) 224.0.0.0/4
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.1.1 (?), elected via Auto-RP
         Uptime: 00:31:43, expires: 00:02:02
Group(s) (-)224.110.110.110/32
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.1.1 (?), elected via Auto-RP
         Uptime: 00:31:43, expires: 00:01:59
Group(s) 232.0.0.0/5
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.1.1 (?), elected via Auto-RP
         Uptime: 00:31:43, expires: 00:02:01
Rack1SW2#
```

Copyright © 2010 Internetwork Expert

8.17 Multicast Boundary

- Configure R5's connection to VLAN 58 so that traffic to the group range 232.0.0.0/5 cannot reach SW2.
- Filter the Auto-RP messages to remove the information about this group range.

Configuration

Note

The multicast boundary feature allows for setting administrative borders for multicast traffic. This feature applies filtering to both the control plane traffic (IGMP, PIM, AutoRP) and the data plane (installing multicast route states out of the configured interface). It is much more flexible than using Auto-RP TTL scoping and allows the application of finer and more granular access control. Using this feature you may contain multicast traffic within the boundaries of your administrative domain, without relying on TTL-based filtering.

When you apply the command **`ip multicast boundary <access-list> [filter-autorp]`** to an interface, the following filtering rules apply.

1) If the access-list is a standard ACL, then any ingress IGMP or PIM messages are inspected to see if the group being joined or tree being built has a match in the access-list. This might be an (S,G) or (*,G) join. Additionally, the interface is used as an outgoing interface to forward a group "G" only if the group matches the access-list.

2) If the access-list is an extended ACL, then it specifies both multicast sources and groups, using the format **`permit ip <src-ip> <src-wildcard> <group-address> <group-mask>.`** Any incoming PIM/IGMP messages are inspected, and if both the source and group are matched, they are permitted. At the same time, the interface could be used as outgoing for multicast traffic sourced off the IP addresses matching the extended access-list with the group matching the same access-list entry. If you want to match only (*,G) shared tree signaling, specify the source IP address of 0.0.0.0 – this will affect PIM Join/Prune messages.

Keep in mind that unicast PIM Register messages are not affected by the multicast-boundary configuration, and must be filtered using the respective feature.

Copyright © 2010 Internetwork Expert

If you have specified the **filter-autorp** keyword, then the router will inspect any Auto-RP messages (announces or discovery) and filter away those not matching the access-list. Note that the access-list must be a *standard* ACL if you use Auto-RP filtering. In order for the Auto-RP group range to be permitted, the whole range must be covered by permit statements in the access-list. If any part of the range's group is not permitted, the whole range is removed from the advertisement.

Starting with version 12.3(17)T you may use **in** or **out** options with the multicast boundary command (this does not work with the **filter-autorp** option though). When applied as an ingress filter, the command affects control plane traffic – IGMP/PIM Joins and Auto-RP messages. When configured as an egress filter, it will control the interface being added to the OIL for multicast groups, allowing only the groups permitted by the access-list.

R5:

```
ip access-list standard PERMITTED_GROUPS
 deny 232.0.0.0 7.255.255.255
 permit any
!
interface FastEthernet 0/0
 ip multicast boundary PERMITTED_GROUPS filter-autorp
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Notice the following console message on R5. Because the range 224.0.0.0-239.255.255.255 advertised by both SW2 and SW4 *overlaps* with the denied group range, it is also filtered from Auto-RP announces, in addition to the explicitly denied range. This results in the following RP to group mapping in the MA. (Again please be advised that it may take a few moments for mappings to expire from the RP Mapping list.)

```
Rack1R5#
%AUTORP-4-OVERLAP: AutoRP Announcement packet, group 224.0.0.0 with
mask 240.0.0.0 removed because of multicast boundary for 232.0.0.0 with
mask 248.0.0.0

Rack1R1#show ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP-mapping agent (Loopback0)

Group(s) 224.0.0.0/5
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 01:34:10, expires: 00:02:46
Group(s) (-)224.110.110.110/32
  RP 150.1.10.10 (?), v2v1
    Info source: 150.1.10.10 (?), elected via Auto-RP
         Uptime: 01:33:20, expires: 00:02:34
  RP 150.1.8.8 (?), v2v1
    Info source: 150.1.8.8 (?), via Auto-RP
         Uptime: 01:34:10, expires: 00:02:46
```

> **Note**
>
> Based on the above output, SW2 won't even be able to join the shared trees for the denied groups, as it does not know the RP for those groups.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

8.18 PIM Bootstrap Router

- Remove the Auto-RP configuration on SW2, SW4 and R1.
- Remove the multicast boundary configuration on R5.
- Configure R5 to advertise itself as the RP for all multicast groups using the standards-based protocol.

Configuration

Note

Bootstrap router or BSR is a standards-based solution available with PIMv2, which performs the same function as Auto-RP, i.e. disseminates RP information. Both protocols use the concept of a candidate RP. However, BSR does not use any dense-mode groups to distribute RP-to-Group mapping information. Instead, the information is flooded using PIM messages, on a hop-by-hop basis. That is, when a router receives a candidate RP announcement inside a PIM message, it applies an RPF check, validating that the announcement was received on the interface that is on the shortest path to the RP. If the RPF check succeeds, the message is flooded out of all PIM-enabled interfaces.

In order to configure a candidate RP use the command **`ip pim rp-candidate <PIM-Enabled-Interface> [group-list <Standard-ACL>] [interval <Seconds>] [priority <0-255>]`**. If you omit all arguments, the router will start advertising itself as the RP for all groups. You may specify a list of groups using the **`group-list`** argument. All entries in this list are "positive", if you compare that to Auto-RP, and you cannot use "negative" groups. The priority value is used when the routers select the best RP for a given group, and lower values are preferred. The default priority is zero (which is against the standard, which specifies a value of 192) and which is the highest possible value. You would very rarely want to change the priority value for a candidate RP, possibly only in cases when you want to gracefully take the RP out of service.

Now, the part where BSR differs from Auto-RP is the bootstrap router itself. This router performs the role similar to the Auto-RP MA, by listening to candidate RP announcements. However, unlike the Auto-RP MA, BSR does not elect the best RP for every group range. Instead of this, for every group range it builds a set of candidate RPs, including all routers that advertised their willingness to service this group range. This is called the group range to RP set mapping.

Copyright © 2010 Internetwork Expert

The resulting array of group range to RP set mappings is distributed by the BSR using PIM messages and the same flooding procedure described above. The command to configure a router as a BSR is **ip pim bsr-candidate <Interface-Name> [hash-mask-length] [priority]**. Ignore the **hash-mask-length** parameter for the moment, and notice the priority field. By default, the priority of zero is advertised in all BSR messages. The higher the priority value, the more preferred the BSR. The IP address of the interface used to source the BSR messages is used as a tie-breaker in case two priorities match – the higher IP is preferred. If there are multiple BSRs, they all listen to other potential BSR messages. If a BSR hears a message with a higher priority or IP address, it immediately stops its own BSR advertisements. This process ensures a unique BSR in the domain while maintaining some redundancy.

The bootstrap messages are received by every multicast router and used to populate their RP cache. Note that it's up to the routers to select the best matching RP from the sets advertised by the BSR router. In order to facilitate RP load-balancing, routers may use a special hash function to select the best RP from a set that services the same group range.

R1:
```
no ip pim send-rp-discovery Loopback 0
```

R5:
```
ip pim rp-candidate Loopback0
ip pim bsr-candidate Loopback0
!
interface FastEthernet 0/0
 no ip multicast boundary PERMITTED_GROUPS filter-autorp
```

SW2:
```
no ip pim send-rp-announce Loopback 0
```

SW4:
```
no ip pim send-rp-announce Loopback 0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Enable PIM BSR debugging on R5 to see how BSR messages are originated. Notice that R5 sends both candidate RP and BSR router messages. After that, check the RP mappings on all multicast routers to ensure they all received the BSR information.

```
Rack1R5#debug ip pim bsr
PIM-BSR debugging is on
Rack1R5#
PIM-BSR(0): RP-set for 224.0.0.0/4
PIM-BSR(0):   RP(1) 150.1.5.5, holdtime 150 sec priority 0
PIM-BSR(0): Bootstrap message for 150.1.5.5 originated
PIM-BSR(0): Build v2 Candidate-RP advertisement for 150.1.5.5 priority
0, holdtime 150
PIM-BSR(0):  Candidate RP's group prefix 224.0.0.0/4
PIM-BSR(0): Send Candidate RP Advertisement to 150.1.5.5
PIM-BSR(0):  RP 150.1.5.5, 1 Group Prefixes, Priority 0, Holdtime 150

Rack1R1#show ip pim bsr-router
PIMv2 Bootstrap information
  BSR address: 150.1.5.5 (?)
  Uptime:      00:21:04, BSR Priority: 0, Hash mask length: 32
  Expires:     00:01:35

Rack1R1#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/4
  RP 150.1.5.5 (?), v2
    Info source: 150.1.5.5 (?), via bootstrap, priority 0, holdtime 150
         Uptime: 00:05:23, expires: 00:02:06


Rack1R6#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/4
  RP 150.1.5.5 (?), v2
    Info source: 150.1.5.5 (?), via bootstrap, priority 0, holdtime 150
         Uptime: 00:06:34, expires: 00:01:52
```

Copyright © 2010 Internetwork Expert

Note

Always remember that BSR messages are subject to RPF checks. If your router does not receive BSR information, enable the command **debug ip pim bsr** and see if the locally received BSR packets are dropped due to RPF checks. For example, temporarily configure a static route on R1 for R5's Loopback 0 to make R1 think it's reachable via **R6:**

R1:

```
ip route 150.1.5.5 255.255.255.255 155.1.146.6
```

```
Rack1R1#
PIM-BSR(0): bootstrap from non-RPF neighbor 155.1.146.6
PIM-BSR(0): bootstrap on non-RPF path Tunnel1
PIM-BSR(0): bootstrap on non-RPF path Serial0/0.1

Rack1R1#show ip pim rp mapping
PIM Group-to-RP Mappings

Rack1R1#
```

Copyright © 2010 Internetwork Expert

8.19 BSR - Multiple RP Candidates

- Configure SW2 and SW4 to advertise themselves as RP candidates for all multicast groups.
- R5 should distribute this information and instruct all routers to load-balance multicast groups between the two RPs.
- Use the maximum possible hash mask length to evenly distribute the load across the RPs.

Configuration

Note

As mentioned in the previous task, BSR protocol distributes multicast group ranges along with the RP-Set information. This allows the multicast routers to effectively load-balance among multiple RPs using a special procedure. This procedure is deterministic, and ensures that for a given group ALL multicast routers will select the same RP. This is needed to make sure a source for a given group will not register to an RP that is not selected for this group. Here is how the procedure works.

Input: Group Address (G), RP-Set (R1, R2… Rn), Mask (distributed by BSR).

1) Among the routers in the RP-Set, select those that have the numerically lowest priority values. By default all cRPs advertise a priority of zero, so they all are eligible. You may adjust the priority and take some of the RPs out of service gracefully.

2) For every RP IP address, calculate the hash function value:

value1 = Hash(G&Mask, R1), value2 = Hash(G&Mask, R2) … valueN = Hash(G&Mask,Rn).

Notice that the Group IP address is ANDed with the Mask value. The mask value is propagated by the BSR using the **`hash-mask-length`** parameter. Thus, only first **`hash-mask-length`** bits of the Group are used to calculate the hash value. The default value is zero – i.e. the group IP address is ignored when computing the hash value and all groups map to the same RP.

3) Select the RP with the highest hash function value for a given group. If the values are the same, select the RP with the highest IP address.

Copyright © 2010 Internetwork Expert

Using this “pseudo-random” selection procedure, the whole multicast address space is partitioned among different RPs. Every RP will get approximately 2^[32-**hash-mask-length]** groups assigned, provided that there is enough RPs to evenly distribute the load.

BSR protocol implements an automatic failover procedure. If any one of the RPs were to fail, the BSR will exempt it from bootstrap messages and automatic failover will occur. The failover delay is based on the RP/BSR advertisement intervals. Naturally, if there are redundant BSRs, and the primary fails, the secondary would revive and take its role.

In this task, odd/even groups are distributed among two RPs, both covering the same group ranges, based on the hash mask length of 31 bits. This ensures the load is evenly distributed among the two RPs.

R5:
```
no ip pim rp-candidate Loopback0
ip pim bsr-candidate Loopback0 31
```

SW2:
```
ip pim rp-candidate Loopback0
```

SW4:
```
ip pim rp-candidate Loopback0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> You can quickly find which RP will be selected for a given group by using the show **ip pim rp-hash** command. For example:

```
Rack1R4#show ip pim rp-hash 239.1.1.1
  RP 150.1.10.10 (?), v2
    Info source: 150.1.5.5 (?), via bootstrap, priority 0, holdtime 202
         Uptime: 00:05:03, expires: 00:02:51
  PIMv2 Hash Value (mask 255.255.255.254)
    RP 150.1.10.10, via bootstrap, priority 0, hash value 989207280
    RP 150.1.8.8, via bootstrap, priority 0, hash value 718054422

Rack1R4#show ip pim rp-hash 239.1.1.2
  RP 150.1.8.8 (?), v2
    Info source: 150.1.5.5 (?), via bootstrap, priority 0, holdtime 205
         Uptime: 00:05:07, expires: 00:02:55
  PIMv2 Hash Value (mask 255.255.255.254)
    RP 150.1.10.10, via bootstrap, priority 0, hash value 1093093598
    RP 150.1.8.8, via bootstrap, priority 0, hash value 1364246456
```

> **Note**
>
> Notice the RP selected for the group and the hash function values.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

8.20 Filtering BSR Messages

- Configure your network so that R6 does not learn any RP information,

Configuration

> **Note**
>
> As you may recall, filtering Auto-RP messages requires applying the administrative multicast boundary command. Filtering BSR messages is much easier, as those are carried inside PIM. When you apply the command **`ip pim bsr-border`** on a link, BSR messages are no longer flooded or received on that link. This means that any RP or BSR advertisements are filtered, effectively creating two isolated domains of RP information exchange.

R1:
```
interface FastEthernet0/0
 ip pim bsr-border
```

R4:
```
interface FastEthernet0/1
 ip pim bsr-border
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Ensure that R6 did not learn any RP information via the BSR. Again it may take a few moments for any previously existing entries to age out.

```
Rack1R6#show ip pim rp mapping
PIM Group-to-RP Mappings

Rack1R6#
```

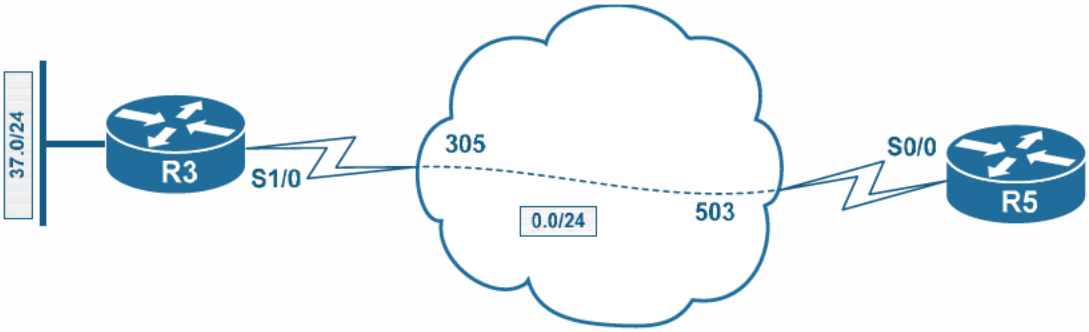
8.21 Stub Multicast Routing & IGMP Helper

- Remove the tunnel connecting R1 and R3. Configure R3 as the stub multicast router with SW1 emulating a host on the stub segment.

Configuration

Note

Stub multicast routing is useful in situations where you have small remote sites connected to a centralized WAN cloud across low-bandwidth links that use low-end routers. Such remote sites never perform any transit function, and may suffer from resource starvation due to PIM DM's flood-and-prune behavior, or PIM SM RP announcements, RP cache growth and mroute state proliferation.



Look at the figure. Assume R5 is the central router and R3 is the stub remote router with directly connected receivers. In the case where both routers are running PIM DM, periodic flood-and-prune behavior has the capacity to overwhelm the WAN link. If both routers were to run PIM SM, then R3 will have to accept RP discovery messages, build its own RP cache and maintain states for all multicast groups joined by local receivers, thus possibly over taxing a small router's resources.

Being a stub multicast router allows R3 to be exempted from PIM and IGMP message processing. Instead of that, R3 is configured to forward all IGMP messages received on its client-facing interface to R5. Thus, R3 never creates any group states. The command to achieve this is **`ip igmp helper-address <Upstream-IP>`**. As a result, R5 will track all IGMP states for any client on the segment connected to R3.

Next, R3 is configured with PIM DM on both the uplink (Frame-Relay in our case) and the client-facing interfaces. This will allow the router to flood ANY multicast traffic received from upstream sources down to clients, this is based on default PIM DM behavior, as there is no downstream router to prune the tree.

Copyright © 2010 Internetwork Expert

Finally, R5 is configured with PIM enabled on its downstream interface, but with a special neighbor filter applied, to make sure the upstream and the downstream routers never form a PIM adjacency. This allows the upstream router to flood multicast traffic downstream, as IOS will not send a multicast packet out of non-PIM interface. At the same time, the downstream router will not be able to prune any group using PIM signaling. The command used to filter PIM neighbors is **`ip pim neighbor-filter <ACL>`** where the standard ACL permits or denies particular neighbors.

The net effect is that the upstream router (R5) builds a multicast distribution tree on behalf of the downstream router (R3) by virtue of the IGMP proxy function performed by the downstream router. At the same time, no excessive load is being put on the stub router or the stub link, effectively saving bandwidth and router resources. Essentially, R5 performs all multicast routing functions for R3, and the latter is only used as a "dumb" packet forwarder.

R3:

```
no interface Tunnel 1
!
interface FastEthernet0/0
 ip igmp helper-address 155.1.0.5
 ip pim dense-mode
!
interface Serial 1/0.1
 ip pim dense-mode
```

R5:

```
no access-list 33
access-list 33 deny 155.1.0.3
access-list 33 permit any
!
interface Serial 0/0/0
 ip pim sparse-mode
 ip pim neighbor-filter 33
```

Copyright © 2010 Internetwork Expert

Note

We need to enable PIM DM on SW1's interface connected to R3 to activate multicast processing on this interface. A neighbor filter is used allow SW1 to avoid becoming PIM neighbors with R3.

SW1:

```
ip multicast-routing distributed
!
access-list 7 deny any
!
interface FastEthernet 0/3
 ip igmp join-group 239.1.1.7
 ip pim dense-mode
 ip pim neighbor-filter 7
```

Verification

Note

Check that R5 sees the group 239.1.1.7 as directly connected on its Frame-Relay interface. Next, ping the group from SW2.

```
Rack1R5#show ip igmp groups
IGMP Connected Group Membership
Group Address    Interface                Uptime    Expires   Last
Reporter   Group Accounted
239.1.1.7        Serial0/0/0                00:00:15  00:02:44
155.1.0.3
<snip>

Rack1SW2#ping 239.1.1.7 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 239.1.1.7, timeout is 2 seconds:
.
Reply to request 1 from 155.1.37.7, 67 ms
Reply to request 2 from 155.1.37.7, 67 ms
Reply to request 3 from 155.1.37.7, 67 ms
```

Copyright © 2010 Internetwork Expert

Note

Check the multicast route states on R5 and R3. Notice that R5 uses sparse-mode SPT to forward traffic down to R3. In turn, R3 simply floods the traffic using dense mode forwarding to the directly connected source.

```
Rack1R5#show ip mroute 239.1.1.7
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.1.1.7), 00:07:21/stopped, RP 150.1.8.8, flags: SJC
  Incoming interface: FastEthernet0/0, RPF nbr 155.1.58.8
  Outgoing interface list:
    Serial0/0/0, Forward/Sparse, 00:02:15/00:02:42

(155.1.58.8, 239.1.1.7), 00:00:15/00:02:58, flags: JT
  Incoming interface: FastEthernet0/0, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0/0, Forward/Sparse, 00:00:15/00:02:44

Rack1R5#
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#show ip mroute 239.1.1.7
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.1.1.7), 00:14:41/stopped, RP 150.1.8.8, flags: SJC
  Incoming interface: FastEthernet0/0, RPF nbr 155.1.37.7
  Outgoing interface list:
    Serial1/0.1, Forward/Dense, 00:14:41/00:00:00

(155.1.58.8, 239.1.1.7), 00:00:27/00:02:54, flags: JT
  Incoming interface: Serial1/0.1, RPF nbr 155.1.0.5
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:00:27/00:00:00
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

8.22 IGMP Filtering

- Only permit R3 to accept IGMP joins for groups in the range 239.1.1.0/24 via its connection to SW1.
- Limit the number of concurrent IGMP states on the interface to 10.

Configuration

Note

IGMP is the protocol used by multicast receivers to communicate their willingness to listen to a particular multicast group. When a host wants to join a multicast group, it sends an IGMP report message to the multicast address heard by all routers on the segment. This report contains the multicast group that the host wants to join. The multicast router may control groups allowed to be joined by the receivers. When you apply the command **`ip igmp access-group <ACL>`** to an interface, the router will filter all attempts to join groups not matching the access-list. Recall that can accomplish this goal using the command **`ip multicast boundary`**, but **`ip igmp access-group`** is more commonly used on the interfaces facing receivers. Notice that you can use either standard or extended access-lists with this command.

If you use a standard access-list, your configuration applies to IGMP v1, v2 and v3 receivers. The hosts are allowed to listen to the channels (multicast groups) matching an entry in the access-list. If you use an extended access-list, then you may also selectively filter IGMPv3 reports. IGMPv3 allows receivers to join explicit sources along with the multicast group. That is, every IGMPv3 report contains the list of groups along with the multicast sources that the receiver wants to listen. The access-list entry will have the format **`permit ip <src-ip> <src-mask> <group-ip> <group-mask>`**. If you want to filter joins to any source, use the 0.0.0.0 255.255.255.255 wildcard pair. However, if you want to filter IGMPv2 or v1 joins, that don't support explicit source specification, use the host IP address of **`0.0.0.0`** for the source.

Another useful feature is limiting the number of mroute states created for the interface as a result of IGMP reports. The same command **`ip igmp limit <N>`** can be applied globally and per-interface at the same time. In the first case, it limits the aggregate number of multicast groups joined by directly connected receivers on all multicast interfaces. When applied per-interface, it limits the number of different multicast groups that can be joined on this particular interface.

Copyright © 2010 Internetwork Expert

R3:

```
ip access-list standard IGMP_FILTER
 permit 239.1.1.0 0.0.0.255
!
interface FastEthernet0/0
 ip igmp access-group IGMP_FILTER
 ip igmp limit 10
```

Copyright © 2010 Internetwork Expert

Verification

Note

Start by checking the IGMP settings on R3's interface. Notice that the maximum number of allowed IGMP states is 10.

```
Rack1R3#show ip igmp interface fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  Internet address is 155.1.37.3/24
  IGMP is enabled on interface
  Current IGMP host version is 2
  Current IGMP router version is 2
  IGMP query interval is 60 seconds
  IGMP querier timeout is 120 seconds
  IGMP max query response time is 10 seconds
  Last member query count is 2
  Last member query response interval is 1000 ms
  Inbound IGMP access group is IGMP_FILTER
  IGMP activity: 6 joins, 5 leaves
  Interface IGMP State Limit : 1 active out of 10 max
  Multicast routing is enabled on interface
  Multicast TTL threshold is 0
  Multicast designated router (DR) is 155.1.37.7
  IGMP querying router is 155.1.37.3 (this system)
  IGMP helper address is 155.1.0.5
  No multicast groups joined by this system
```

Note

Now configure SW1 to join a different group, e.g. 239.2.2.7.

SW1:

```
interface FastEthernet 0/3
 ip igmp join-group 239.2.2.7
```

Check that there is still just one IGMP state created on R3:

```
Rack1R3#show ip igmp groups
IGMP Connected Group Membership
Group Address    Interface                Uptime    Expires   Last
Reporter   Group Accounted
239.1.1.7        FastEthernet0/0          00:03:09  00:02:56
155.1.37.7      Ac
224.0.1.40       Loopback0                04:57:32  00:01:57  150.1.3.3
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#show ip igmp interface fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  Internet address is 155.1.37.3/24
  IGMP is enabled on interface
  Current IGMP host version is 2
  Current IGMP router version is 2
  IGMP query interval is 60 seconds
  IGMP querier timeout is 120 seconds
  IGMP max query response time is 10 seconds
  Last member query count is 2
  Last member query response interval is 1000 ms
  Inbound IGMP access group is IGMP_FILTER
  IGMP activity: 6 joins, 5 leaves
  Interface IGMP State Limit : 1 active out of 10 max
  Multicast routing is enabled on interface
  Multicast TTL threshold is 0
  Multicast designated router (DR) is 155.1.37.7
  IGMP querying router is 155.1.37.3 (this system)
  IGMP helper address is 155.1.0.5
  No multicast groups joined by this system
```

Copyright © 2010 Internetwork Expert

8.23 IGMP Timers

- Configure the designated IGMP querier on the VLAN 146 segment so that failed multicast traffic receivers are detected and removed within 60 seconds.
- Every active receiver should respond to general IGMP queries within 4 seconds.
- Designated querier failures should be detected two times faster than by default.
- Since there is just one receiver on R3's connection to SW1, configure the router to remove multicast group state immediately after a corresponding IGMP leave report has been received.

Configuration

Note

IGMP reports are sent asynchronously, and thus might be missed by the router. Therefore, one of the multicast-capable routers sharing the same segment will be elected as the designated IGMP querier, and will periodically send IGMP Membership queries to all hosts on the segment. Every host receiving the Membership query should respond with an IGMP report containing all joined groups. The designated querier is the router with the lowest IP address on the segment, if the running IGMP version is 2 or 3. Notice that the PIM DR on the segment is elected based on the highest IP address by default, and thus some functional decoupling and load-balancing may occur if there is more than one router on the segment. In addition to sending the periodic membership queries, the designated querier also builds PIM SM shared trees, on behalf of the receivers signaling multicast group membership.

Periodic queries are sent at the interval defined by the interface-level command **`ip igmp query-interval <seconds>`**. If a non-designated multicast router does not hear the membership queries for more than the interval defined by the command **`ip igmp querier-timeout <seconds>`** it will attempt to become a designated querier itself, assuming that the old one failed. If the latter command is not configured on the interface, the querier timeout equals twice the query-interval configured on the same interface. The default query-interval value is 60 seconds (though the RFC recommends 125 seconds).

Copyright © 2010 Internetwork Expert

When IGMPv1 is configured on an interface explicitly, the router times out the group state if there is no report for this group during 3 consecutive membership queries (180 seconds by default). This might result in very high leave latency, as IGMPv1 has no explicit group leave message. IGMPv2 significantly enhances the procedure of leaving a multicast group as detailed below:

1) Every membership query message contains a special timer value, defined by the command **ip igmp query-max-response-time [time-in-seconds].** Every host on the segment is supposed to send an IGMP report during the time-window defined by this interval. All hosts count to the interval value and randomly fire the IGMP report. If a host hears another report for the same group, it suppresses its own message. Thus, excessive report flooding is avoided. If there is no report for a given group during the query-max-response interval, the router removes the mroute state for this group.

2) Every host, willing to leave a group, may send out a special IGMP Leave report for this particular group. As soon as the router receives a leave report, it generates a special IGMP group specific query for the multicast channel in question. This query is needed to check if there are any other hosts on the segment that still need this group. The number of special queries generated is defined by the command **ip igmp last-member-query-count <N>** which is set to 2 by default. The Queries are sent at the intervals specified by the command **ip igmp last-member-query-interval <milliseconds>** which is 1000ms by default. If there is no answer to the special queries, the group state is removed from the mroute table.

3) If there is just one receiver for a particular group on the segment, e.g. if there is just one host connected to the router, the leave latency could be further reduced by configuring the **ip igmp immediate-leave group-list <access-list>** command. If there is an IGMP leave message received on the interface for a group matching the access-list, the mroute state is immediately removed, without any further delays. The access-list is a standard ACL that defines the groups eligible for this feature.

Notice that the explicit leave feature is also available for IGMPv3, with the extension to the multicast group source. That is to say, a host may leave a particular sender for the group, while continuing to listen to other senders.

Copyright © 2010 Internetwork Expert

For our scenario, R1 is the designated querier for the segment. Since R4 has the next lowest IP address, it is the backup querier. In real life, you would configure every router on the segment with the same settings, but this task's goal is to check your understanding of the querier election. For the immediate leave feature we configure only the groups permitted by the IGMP access-group. You may configure it for all groups, as the resulting effect would be the same.

Copyright © 2010 Internetwork Expert

R1:
```
interface FastEthernet0/0
  ip igmp query-interval 20
  ip igmp query-max-response-time 4
```

R3:
```
ip access-list standard IMMEDIATE_LEAVE
 permit 239.1.1.0 0.0.0.255
!
interface FastEthernet 0/0
 ip igmp immediate-leave group-list IMMEDIATE_LEAVE
```

R4:
```
interface FastEthernet0/1
 ip igmp querier-timeout 60
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Verify timers using **the show igmp interface** command.

```
Rack1R1#show ip igmp interface fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  Internet address is 155.1.146.1/24
  IGMP is enabled on interface
  Current IGMP host version is 2
  Current IGMP router version is 2
  IGMP query interval is 20 seconds
  IGMP querier timeout is 40 seconds
  IGMP max query response time is 4 seconds
  Last member query count is 2
  Last member query response interval is 1000 ms
  Inbound IGMP access group is not set
  IGMP activity: 4 joins, 1 leaves
  Multicast routing is enabled on interface
  Multicast TTL threshold is 0
  Multicast designated router (DR) is 155.1.146.1 (this system)
  IGMP querying router is 155.1.146.1 (this system)

Rack1R4#show ip igmp interface fastEthernet 0/1
FastEthernet0/1 is up, line protocol is up
  Internet address is 155.1.146.4/24
  IGMP is enabled on interface
  Current IGMP host version is 2
  Current IGMP router version is 2
  IGMP query interval is 60 seconds
  IGMP querier timeout is 60 seconds
  IGMP max query response time is 10 seconds
  Last member query count is 2
  Last member query response interval is 1000 ms
  Inbound IGMP access group is not set
  IGMP activity: 5 joins, 2 leaves
  Multicast routing is enabled on interface
  Multicast TTL threshold is 0
  Multicast designated router (DR) is 155.1.146.1
  IGMP querying router is 155.1.146.1
  No multicast groups joined by this system
```

Copyright © 2010 Internetwork Expert

Note

Now enable IGMP debugging on R3 and unsubscribe SW1 from the group 239.1.1.7. Notice that the group state is removed immediately, without any additional last member queries.

SW1:

```
interface FastEthernet 0/3
 no ip igmp join-group 239.1.1.7
```

```
Rack1R3#debug ip igmp
IGMP(0): Received Leave from 155.1.37.7 (FastEthernet0/0) for 239.1.1.7
IGMP(0): Leave group 239.1.1.7 immediately on FastEthernet0/0
IGMP_ACL(0): Group 239.1.1.7 unaccounted on FastEthernet0/0
IGMP(0): Send v2 general Query on FastEthernet0/0
IGMP(0): Deleting 239.1.1.7 on FastEthernet0/0
IGMP(0): Helper Leave for group 239.1.1.7 out Serial1/0.1
IGMP(0): Received v2 Query on Serial1/0.1 from 155.1.0.5
IGMP(0): Previous querier timed out, v2 querier for Serial1/0.1 is this
systemv2 querier for Serial1/0.1 is this system.
IGMP(0): Received v2 Query on Serial1/0.1 from 155.1.0.5
IGMP(0): Received v2 Query on Serial1/0.1 from 155.1.0.5
IGMP(0): Previous querier timed out, v2 querier for Serial1/0.1 is this
systemv2 querier for Serial1/0.1 is this system.
IGMP(0): Received v2 Report on FastEthernet0/0 from 155.1.37.7 for
239.2.2.7
IGMP(*): Group 239.2.2.7 access denied on FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

8.24 Multicast Helper Map

- When R6 sends broadcast UDP packets on port 5000, those packets should be transported across the network and broadcast on the VLAN 37 segment.
- Use the group 239.1.1.100 to accomplish this task and use DNS broadcasts for testing.
- You may use static mroutes if needed to accomplish this task.

Configuration

Note

The purpose of this feature is to allow forwarding of broadcast traffic across a multicast capable network. Generally, you need a single Layer 2 domain between two nodes to let them hear each other's broadcast packets. However, broadcast UDP packets can be relayed between two subnets using a special IOS feature known as the helper-address. There are two variations of this feature:

1) Unicast helper (**`ip helper-address`**), which converts the broadcast destination address to the fixed unicast IP address. Most often this feature is used with DHCP to forward requests to the server.
2) Multicast helper (**`ip multicast helper-map`**), which converts the broadcast destination to a fixed multicast address.

The multicast helper-map feature allows scalable forwarding of broadcast traffic between disjoined segments. This is often needed to support legacy applications like stock tickers that use broadcast to deliver information to multiple sources simultaneously. To configure multicast helper, follow the steps outlined below:

Step 1:
Set up a multicast network between the segments that should exchange broadcast packets. You should select a group to deliver the broadcast packets and decide which PIM mode to use. If you chose PIM SM, make sure the group you chose maps to an RP. Make sure multicasting works, by joining an interface on the egress router (closest to the broadcast receiver) to the selected group and ping this group from the ingress router (closest to the broadcast source).

Copyright © 2010 Internetwork Expert

Step 2:
Enabling broadcast forwarding on the ingress router – the one directly connected to the source. If there are multiple sources, you have to configure all respective routers. Use the command **ip forward-protocol udp <port-number>** to enable forwarding of broadcast UDP packets sent to the specified port-number.

Step 3:
Configure a multicast helper-map on the ingress routers to redirect broadcast packets to the selected multicast address. The syntax for this interface-level command is **ip multicast helper-map broadcast <mcast-address> <ACL>**. The access-list controls which broadcast packets are eligible to be converted into the multicast. For example, if you want to forward UDP packets destined to port 5000 use an access-list similar to the following: **access-list 100 permit udp any any eq 5000**. Note that the same UDP port must be enabled for broadcast forwarding at **Step 2**. All broadcast traffic received on the configured interface that matchs the access-list is converted and sent to the specified multicast address. If the group is in sparse mode, the ingress router will register the source with the RP, per the usual procedure.

Step 4:
Enable broadcast forwarding on the egress router, i.e. the router directly connected to the destination subnet. Use the same command that you used in Step 2 **ip forward-protocol udp <port-number>** to accomplish this. Next, enable multicast helper map on the egress router for all interfaces that may *receive* multicast traffic. Note that you should not configure the multicast-helper on the interface connected to the destination. Use the command **ip multicast helper-map <mcast-group> <directed-broadcast-IP> <ACL>** where mcast-group is the same group you used in **Step 3** and directed-broadcast-IP is the broadcast subnet IP address on the segment that receives the broadcast traffic.

Step 5:
Enable directed broadcasts on the interface connected to the receiving segment using the command **ip directed-broadcast**. This is needed to successfully send broadcasts out of this segment. By default, the broadcasts are sent to the address 255.255.255.255 irrespective of the **directed-broadcast-IP** configured at **Step 4**. If you want to use a different address, put the command **ip broadcast-address <IP>** on the same segment.

Copyright © 2010 Internetwork Expert

To test your configuration, you will need a broadcast packet source. You may use the IP SLA command to generate UDP packets to a segment broadcast address, but this might not work on some platforms/IOS versions. If that's the case, you may use either of the following two methods:

1) Enable DNS name resolution but do not configure a DNS server. After this, the router will broadcast for any DNS name entered in the command line using the address 255.255.255.255 out all interfaces. You will need to adjust the port in your access-lists to forward this broadcast traffic.

2) Configure an extended **`traceroute`** command using parameters to trace to the broadcast destination, starting off the port number that is covered by your ACL.

Copyright © 2010 Internetwork Expert

Pitfall

There are some caveats associated with our particular scenario.

1) You need to select the proper ingress router. The group 239.1.1.100 is going to be transported in sparse mode. Thus, the source should be registered with the RP first. The only router allowed to register sources is the DR, which is R1. Thus, we must configure R1 to generate the multicast packets.

2) The next issue, is that R3 is a stub multicast router. Thus, it does not exchange any PIM messages with R5 and it won't join group 239.1.1.100 once you configure the multicast helper on the ingress interface. To resolve this issue, SW1 should be configured with a static IGMP join for group 239.1.1.100 to let R5 know about that.

3) If you loaded our default configuration, then you will see that R3 has three equal-cost routes to reach the VLAN 146 segment. This will cause RPF failures, unless you configure **`ip multicast multipath`** on R3, allowing R3 to perform RPF across the equal-cost paths.

4) Finally, R5 will prefer the path to the VLAN 146 segment across the Serial connection to R4. Thus, in order to allow R5, which acts on behalf of R3, to join the SPT towards the VLAN 146 segment, you need a static mroute on R5, to make R1 the RPF neighbor for the VLAN 146 subnet.

Also notice that this scenario requires PIM NBMA mode to be configured on R5's Frame-Relay interface to work properly.

Copyright © 2010 Internetwork Expert

R1:
```
ip forward-protocol udp 5000
!
ip access-list extended TRAFFIC
 permit udp any any eq 5000
 permit udp any any eq 53
!
interface FastEthernet 0/0
 ip multicast helper-map broadcast 239.1.1.100 TRAFFIC
```

R3:
```
ip forward-protocol udp 5000
ip multicast multipath
!
ip access-list extended TRAFFIC
 permit udp any any eq 5000
 permit udp any any eq 53
!
interface FastEthernet 0/0
 ip directed-broadcast
 ip broadcast-address 155.1.37.255
!
interface Serial 1/0.1
 ip multicast helper-map 239.1.1.100 155.1.37.255 TRAFFIC
```

R5:
```
ip mroute 155.1.146.0 255.255.255.0 155.1.0.1
```

SW1:
```
interface FastEthernet0/3
ip igmp join-group 239.1.1.100
```

Copyright © 2010 Internetwork Expert

Verification

Note

For verification we are going to use DNS broadcasts sent from R6. Configure R6 to resolve DNS names, but do not provide any DNS server:

R6:
```
ip domain lookup
```

Enable debugging and start a traffic flow on R6:

```
Rack1R1#debug ip mpacket
Rack1R1#conf t
Rack1R1(config)#access-list 100 permit udp any any eq 53
Rack1R1#debug ip packet detail 100

Rack1R3#debug ip mpacket
Rack1R3#conf t
Rack1R3(config)#access-list 100 permit udp any any eq 53
Rack1R3#debug ip packet detail 100

Rack1R6#dddd
Translating "dddd"...domain server (255.255.255.255)
```

Copyright © 2010 Internetwork Expert

Note

R1 accepts the broadcasts and converts them to multicast packets. Initially, the SPT is not built and some packets are lost, and the OIL for the (S,G) is empty. When the SPT is finally finished, everything will work smoothly.

```
Rack1R1#
IP: tableid=0, s=155.1.146.6 (FastEthernet0/0), d=155.1.146.255
(FastEthernet0/0), routed via RIB
IP: s=155.1.146.6 (FastEthernet0/0), d=155.1.146.255 (FastEthernet0/0),
len 50, rcvd 3
    UDP src=54670, dst=53
IP(0): s=155.1.146.6 (FastEthernet0/0) d=239.1.1.100 id=0, ttl=254,
prot=17, len=64(50), mroute OIL null
IP: tableid=0, s=155.1.146.6 (FastEthernet0/0), d=155.1.146.255
(FastEthernet0/0), routed via RIB
IP: s=155.1.146.6 (FastEthernet0/0), d=155.1.146.255 (FastEthernet0/0),
len 50, rcvd 3
    UDP src=54670, dst=53
IP(0): s=155.1.146.6 (FastEthernet0/0) d=239.1.1.100 (Serial0/0.1)
id=1, ttl=254, prot=17, len=50(50), mforward
```

Note

R3 receives the multicast packets and sends them as broadcasts to SW1.

```
Rack1R3#
IP(0): s=155.1.146.6 (Serial1/0.1) d=239.1.1.100 (FastEthernet0/0)
id=0, ttl=252, prot=17, len=50(50), mforward
IP: tableid=0, s=155.1.146.6 (Serial1/0.1), d=155.1.37.255
(FastEthernet0/0), routed via RIB
```

Copyright © 2010 Internetwork Expert

> **Note**
>
> Check the mroute states on R1, R3 and R5 to ensure the traffic follows the SPT.

```
Rack1R1#show ip mroute 239.1.1.100
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.1.1.100), 00:00:13/stopped, RP 150.1.8.8, flags: SPF
  Incoming interface: Serial0/0.1, RPF nbr 155.1.0.5
  Outgoing interface list: Null

(155.1.146.6, 239.1.1.100), 00:00:13/00:03:23, flags: FT
  Incoming interface: FastEthernet0/0, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0.1, Forward/Sparse, 00:00:13/00:03:16

Rack1R5#show ip mroute 239.1.1.100
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.1.1.100), 00:37:47/stopped, RP 150.1.8.8, flags: SJC
  Incoming interface: FastEthernet0/0, RPF nbr 155.1.58.8
  Outgoing interface list:
    Serial0/0/0, 155.1.0.3, Forward/Sparse, 00:37:47/00:02:05

(155.1.146.6, 239.1.1.100), 00:00:23/00:02:54, flags: T
  Incoming interface: Serial0/0/0, RPF nbr 155.1.0.1, Mroute
  Outgoing interface list:
    Serial0/0/0, 155.1.0.3, Forward/Sparse, 00:00:24/00:02:59
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#show ip mroute 239.1.1.100
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.1.1.100), 00:39:01/stopped, RP 150.1.8.8, flags: SJCL
  Incoming interface: FastEthernet0/0, RPF nbr 155.1.37.7
  Outgoing interface list:
    Serial1/0.1, Forward/Dense, 00:39:01/00:00:00

(155.1.146.6, 239.1.1.100), 00:00:32/00:02:53, flags: LJT
  Incoming interface: Serial1/0.1, RPF nbr 155.1.0.5
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:00:32/00:00:00
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

8.25 Multicast Rate Limiting

- Configure R3 to limit the aggregate rate of the multicast traffic leaving the VLAN 37 interface to 1Mbps.
- Any source sending to the group IP 239.1.1.7 should be limited to 128Kbps.
- Multicast feeds from R6 to the IP address 239.1.1.100 should be limited to 256Kbps.

Configuration

Note

You could use regular QoS policing and rate-limiting commands to control multicast traffic flow. However, there is a special command for multicast traffic rate control that has its own unique features. The command is **`ip multicast rate-limit {in | out} [group-list <acl>] [source-list <acl>] [<limit>]`** where **`limit`** is specified in Kilobits per second. The **`in`** and **`out`** keywords control the ingress or egress traffic respectively. If you omit the group-list and source-list, the limit will apply to the aggregate multicast traffic rate, for all groups. However, if you omit the speed limit parameter, the command will discard any multicast traffic matching both the group and source-list, if those are specified. When applied without any parameters, this command will simply drop all multicast traffic.

This command behaves differently when configured with group-list (and possible source-list). When you designate the groups to be rate limited, the limit will apply to each source/group pair *individually*. That is, if you configure the command to limit the rate of traffic destined to the group 239.0.0.1 to 128Kbs and there are 3 independent sources, then the limit will apply to every source, resulting in 3x128 Kbps of aggregate rate.

You may combine the group-specific limits with the aggregate limit on the same interface. This way, you will apply per-flow rate-limits and at the same time control the aggregate rate. For example:

```
ip multicast rate-limit out group-list 100 128
ip multicast rate-limit out 512
```

Copyright © 2010 Internetwork Expert

Also, you may apply multiple group-specific multicast rate-limiting commands. In this case, the particular (S,G) pair is limited based on the first match in the configured access-lists. Notice that this restriction makes the order of the statements particularly important! If you put the aggregate limit in the beginning of the list, it will match everything and other entries will never be matched.

R3:

```
ip access-list standard GROUP7
 permit 239.1.1.7
!
ip access-list standard GROUP100
 permit 239.1.1.100
!
interface FastEthernet0/0
 ip multicast rate-limit out group-list GROUP7 128
 ip multicast rate-limit out group-list GROUP100 256
 ip multicast rate-limit out 1000
```

SW1:

```
interface FastEthernet0/3
ip igmp join-group 239.1.1.100
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> To verify the limits, source some multicast packets to the groups 239.1.1.7 and 239.1.1.100. Then check the mroute states on R3. Notice that every group limit appears next to the respective (S,G) entry:

```
Rack1R1#ping 239.1.1.7 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 239.1.1.7, timeout is 2 seconds:

Reply to request 0 from 155.1.37.7, 128 ms
Reply to request 0 from 155.1.37.7, 128 ms

Rack1R6#dddd
Translating "dddd"...domain server (255.255.255.255)

Rack1R3#show ip mroute
IP Multicast Routing Table
<snip>

(*, 239.1.1.100), 01:17:18/stopped, RP 150.1.8.8, flags: SJCL
  Incoming interface: FastEthernet0/0, RPF nbr 155.1.37.7
  Outgoing interface list:
    Serial1/0.1, Forward/Dense, 01:17:18/00:00:00

(155.1.146.6, 239.1.1.100), 00:00:04/00:02:57, flags: LJT
  Incoming interface: Serial1/0.1, RPF nbr 155.1.0.5
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:00:04/00:00:00, limit 256 kbps

(*, 239.1.1.7), 02:08:57/stopped, RP 150.1.8.8, flags: SJCF
  Incoming interface: FastEthernet0/0, RPF nbr 155.1.37.7
  Outgoing interface list:
    Serial1/0.1, Forward/Dense, 02:08:57/00:00:00

(150.1.1.1, 239.1.1.7), 00:02:38/00:02:59, flags: JT
  Incoming interface: Serial1/0.1, RPF nbr 155.1.0.5
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:02:38/00:00:00, limit 128 kbps

(155.1.0.1, 239.1.1.7), 00:02:38/00:02:59, flags: FT
  Incoming interface: Serial1/0.1, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:02:38/00:00:00, limit 128 kbps
```

Copyright © 2010 Internetwork Expert

```
(155.1.146.1, 239.1.1.7), 00:02:38/00:00:22, flags: JT
  Incoming interface: Null, RPF nbr 155.1.13.1
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:02:38/00:00:00, limit 128 kbps
    Serial1/0.1, Forward/Dense, 00:02:38/00:00:00, A

(*, 224.110.110.110), 02:14:36/00:02:31, RP 150.1.10.10, flags: SJCL
  Incoming interface: FastEthernet0/0, RPF nbr 155.1.37.7
  Outgoing interface list:
    Loopback0, Forward/Sparse-Dense, 02:14:37/00:02:30

(*, 224.0.1.40), 02:20:07/00:02:36, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 02:08:58/00:00:00
    Serial1/0.1, Forward/Dense, 02:20:07/00:00:00
```

Copyright © 2010 Internetwork Expert

8.26 Bidirectional PIM

- The group range 238.0.0.0/8 is used for network video-conferencing with many participants.
- Configure the network so that this group uses a single shared tree rooted on R5 for multicast traffic delivery.

Configuration

Note

Bidirectional PIM or PIM BiDir is a special extension to the PIM SM concept that uses only the shared tree for multicast distribution. This mode of operation is useful in situations where most receivers are also senders at the same time. For example, this might be the case when you run videoconferencing. In this situation, in addition to joining the shared tree rooted at the RP, every receiver needs to join the shortest-path multicast distribution tree rooted at every other participant. If the number of participants is significant, the amount of multicast route states in the core of the network will grow at a quadric rate.

One special feature of PIM SM shared and shortest path trees are that they are unidirectional – traffic passes down from the root to the leaves of the tree. PIM BiDir uses a single distribution tree rooted at the RP for all sources and receivers at the same time. If there are multiple RPs, there could be many BiDir trees. Unlike the classic tree, traffic may flow up and down this tree. When a source sends multicast packets, they first flow up to the root of the tree (toward the RP) and then down to all receivers.

To build the bi-directional tree, PIM elects special designated forwarders (DFs) on every link in the network. A DF is elected based on the rules similar to the ones used in the PIM Assert procedure – i.e., the router on the link that has the shortest metric to reach the RP is selected as the DF. Notice that a single router might be the DF on one link and a non-DF on another. After the elections, DF routers are the only routers that are allowed to forward traffic toward the RP – via the bi-directional tree (this is considered the “upstream” portion of the BiDir tree). Every router in the multicast domain creates a (*,G) state for each BiDir group, with the OIL built based on PIM Join messages received from its neighbors. This is the “downstream” portion of the BiDir tree. Any packet received on a valid RPF interface is forwarded based on the OIL. At the same time, the DF will forward a copy of these packets toward the RP - upstream through the shared tree - provided that the packet is not received on the interface pointing to the RP.

Copyright © 2010 Internetwork Expert

Notice that PIM BiDir does not utilize the source registration procedure, via PIM Register/Register-Stop messages. Every source connected to a PIM BiDir capable router may start sending at any time, and the packets will flow upwards to the RP. After reaching the RP, packets are either dropped, if there are no receivers for this group (i.e. the OIL for this (*,G) state is empty) or forwarded down the BiDir tree. There is no way for the RP to signal the source to stop sending traffic even if there are no receivers. This means commands like "ip pim accept-register" covered in lab 8.8 will not work with PIM BiDir, due to the fact that they rely on these "register-stop" messages to work.

Configuring PIM BiDir is relatively simple. You just need to enable BiDir PIM on all multicast routers by using the command **ip pim bidir-enable** and designate particular RP/Group combinations as bi-directional. You can do this in a number of ways.

1) Using a static RP configuration with the command **ip pim rp-address <IP> <ACL> bidir**.

2) Using BSR or Auto-RP for RP information dissemination you may flag particular group/RP combinations as bi-directional using the following syntax:

Auto-RP:
ip pim send-rp-announce <interface> scope <TTL> group-list <ACL> bidir

BSR:
ip pim rp-candidate <interface> group-list <ACL> bidir

This is all you need to enable bi-directional PIM. However, remember to enable bi-directional mode on all routers in your network, or you might end up with routing loops.

```
R1, R3, R4, SW2, SW4:
ip pim bidir-enable

R5:
ip pim bidir-enable
ip access-list standard GROUP238
 permit 238.0.0.0 0.255.255.255
!
ip pim rp-candidate Loopback 0 group-list GROUP238 bidir
```

Copyright © 2010 Internetwork Expert

Verification

Note

To verify, join R1 and SW4 to the bi-directional group 238.1.1.1.

R1:

```
interface FastEthernet 0/0
 ip igmp join-group 238.1.1.1
```

SW4:

```
interface Vlan 10
 ip igmp join-group 238.1.1.1
```

and then ping this group from R5.

```
Rack1R5#ping 238.1.1.1 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 238.1.1.1, timeout is 2 seconds:

Reply to request 0 from 155.1.108.10, 4 ms
Reply to request 0 from 155.1.0.1, 188 ms
Reply to request 0 from 155.1.0.1, 104 ms
Reply to request 0 from 155.1.108.10, 8 ms
Reply to request 1 from 155.1.108.10, 4 ms
Reply to request 1 from 155.1.0.1, 189 ms
Reply to request 1 from 155.1.0.1, 104 ms
Reply to request 1 from 155.1.108.10, 8 ms
```

Note

Check the mroute states on all routers. Notice that some interfaces are marked as Bidir-Upstream – these interfaces are used to send packets upwards to the root of the tree. The root of the tree (the RP) will have no upstream interfaces

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip mroute 238.1.1.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 238.1.1.1), 00:08:15/00:02:59, RP 150.1.5.5, flags: BCL
  Bidir-Upstream: Serial0/0.1, RPF nbr 155.1.0.5
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:08:15/00:02:40
    Serial0/0.1, Bidir-Upstream/Sparse, 00:08:15/00:00:00

Rack1R5#show ip mroute 238.1.1.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 238.1.1.1), 00:26:24/00:03:05, RP 150.1.5.5, flags: B
  Bidir-Upstream: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0/0, 155.1.0.1, Forward/Sparse, 00:12:58/00:00:00
    FastEthernet0/0, Forward/Sparse, 00:26:24/00:02:53
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show ip mroute 238.1.1.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report, Z - Multicast
Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 238.1.1.1), 00:20:36/00:02:39, RP 150.1.5.5, flags: B
  Bidir-Upstream: Vlan58, RPF nbr 155.1.58.5
  Outgoing interface list:
    Port-channel1, Forward/Sparse, 00:20:36/00:02:39
    Vlan58, Bidir-Upstream/Sparse, 00:20:36/00:00:00

Rack1SW4#show ip mroute 238.1.1.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report, Z - Multicast
Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 238.1.1.1), 00:20:54/00:02:54, RP 150.1.5.5, flags: BCL
  Bidir-Upstream: Port-channel1, RPF nbr 155.1.108.8
  Outgoing interface list:
    Port-channel1, Bidir-Upstream/Sparse, 00:20:54/00:00:00
    Vlan10, Forward/Sparse-Dense, 00:20:53/00:02:54, H
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

8.27 Source Specific Multicast

- Using the default multicast group range enable PIM SSM functionality on your network.
- R6 should join the multicast feed (150.1.10.10,232.6.6.6).
- Ensure that the "ip pim bsr-border" commands have been removed from R1 and R4 VLAN 146 interfaces.

Configuration

Note

Classic multicast delivery technologies use IGMPv2 and PIM DM/SM and are known as "Any Source Multicast" or ASM. That is, receivers agree to accept traffic from any source. This is why Rendezvous Points are actually needed in PIM SM - to allow receivers to discover new sources. The core of PIM SSM protocol is the use of IGMPv3 signaling by clients. This client-side protocol allows receivers to specify sources that they want to listen to explicitly. That is, the host may explicitly ask to join group G at source S. PIM SSM works in association with IGMPv3 in building shortest-path trees (SPT) only, towards the sources. There are *no* shared trees in PIM SSM and no RPs are used. Thus, there is no need to use auxiliary protocols like BSR or Auto-RP to distribute RP information. Notice that source discovery is outside the scope of PIM SSM and IGMPv3, and must be accomplished via some other means, like global directory services.

Configuring PIM SSM is relatively straight-forward, since it uses regular PIM messages. You just need to specify the range of groups that are using SSM signaling with the command **`ip pim ssm range {default|range <Standard-ACL>}`**. The **`default`** keyword means that range 232.0.0.0/8 will be used for SSM. For the groups in the SSM range, no shared trees are allowed and the (*,G) joins are dropped.

The second step in configuring PIM SSM is enabling IGMPv3 on the interfaces connected to the receivers capable of using this protocol. Without IGMPv3 there can be no use of PIM SSM, as no other IGMP version allows sources to be selected by the receivers explicitly in order to build shortest-path trees.

Copyright © 2010 Internetwork Expert

R1, R3, R4, R5, SW2, SW4:
```
ip pim ssm default
```

R1:
```
interface FastEthernet 0/0
 ip igmp version 3
```

R6:
```
interface FastEthernet 0/0.146
 ip igmp version 3
 ip igmp join 232.6.6.6 source 150.1.10.10
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> PIM SSM is generally easier to configure than ASM, as it does not require the complicated RP infrastructure. All you need to do is verify the SPT toward the explicit source.

```
Rack1R1#show ip igmp groups 232.6.6.6 detail

Flags: L - Local, U - User, SG - Static Group, VG - Virtual Group,
       SS - Static Source, VS - Virtual Source,
       Ac - Group accounted towards access control limit

Interface:      FastEthernet0/0
Group:          232.6.6.6
Flags:          SSM
Uptime:         00:26:06
Group mode:     INCLUDE
Last reporter:  155.1.146.6
Group source list: (C - Cisco Src Report, U - URD, R - Remote, S -
Static,
                    V - Virtual, M - SSM Mapping, L - Local,
                    Ac - Channel accounted towards access control
limit)
  Source Address   Uptime    v3 Exp   CSR Exp   Fwd  Flags
  150.1.10.10      00:26:06  00:02:54  stopped   Yes  R

Rack1R1#show ip mroute 232.6.6.6 150.1.10.10
IP Multicast Routing Table
<snip>

(150.1.10.10, 232.6.6.6), 00:26:09/00:02:26, flags: sTI
  Incoming interface: Serial0/0.1, RPF nbr 155.1.0.5
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:26:09/00:02:26

Rack1R5#show ip mroute 232.6.6.6 150.1.10.10
IP Multicast Routing Table
<snip>

(150.1.10.10, 232.6.6.6), 00:25:39/00:02:38, flags: sT
  Incoming interface: FastEthernet0/0, RPF nbr 155.1.58.8
  Outgoing interface list:
    Serial0/0/0, 155.1.0.1, Forward/Sparse, 00:25:39/00:02:38
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show ip mroute 232.6.6.6
IP Multicast Routing Table
<snip>

(150.1.10.10, 232.6.6.6), 00:26:11/00:03:04, flags: sT
  Incoming interface: Port-channel1, RPF nbr 155.1.108.10
  Outgoing interface list:
    Vlan58, Forward/Sparse, 00:26:11/00:03:04

Rack1SW4#show ip mroute 232.6.6.6
IP Multicast Routing Table
<snip>

(150.1.10.10, 232.6.6.6), 00:26:46/00:02:31, flags: sT
  Incoming interface: Loopback0, RPF nbr 0.0.0.0
  Outgoing interface list:
    Port-channel1, Forward/Sparse, 00:26:46/00:02:31

Rack1SW4#ping 232.6.6.6

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 232.6.6.6, timeout is 2 seconds:

Reply to request 0 from 155.1.146.6, 48 ms
```

Copyright © 2010 Internetwork Expert

8.28 DVMRP Interoperability

- Configure R4 for this task. Enable multicast routing and configure PIM dense mode on the VLAN 146 interface and Loopack 0 interfaces. Create a DVMRP tunnel sourced off the Loopback 0 interface with a (non-existent) destination 204.12.X.100 (where X is your rack number). Advertise the VLAN 146 subnets to the DVMRP backbone with an offset value of 3. Configure the use of DVMRP routes for RPF check by PIM on VLAN 146 interface.

Configuration

Note

DVMRP or "Distance-Vector Multicast Routing Protocol" is defined in RFC 1075. This protocol was implemented in the UNIX *mrouted* daemon and was the first to gain more or less widespread adoption. DVMRP is based on RIP routing protocol, and uses IGMPv1 messages to carry its routing information. For many years, DVMRP was used as the core routing protocol of the MBONE – an experimental set of multicast-capable networks, used to facilitate multicast testing. However, DVMRP never was a scalable protocol, and most enterprises use PIM as the standards based multicast routing protocol nowadays.

Similar to RIP, DVMRP propagates distance-vector information, but routing updates carry subnets describing multicast sources and metrics to reach them. The metric used by DVMRP is the same hop count used by RIP. When a router receives a DVMRP update, it extracts the subnets contained in the update along with their metrics and stores them in a separate multicast routing table. This table is used to perform RPF checks only, not to route packets based on their destination addresses.

DVMRP implements TRPB – Truncated Reverse Path Broadcasting, which is another name for Constrained RPF. With respect to multicast flooding DVMRP is very much similar to PIM Dense Mode - traffic is flooded using RPF checks and then routers with no subscribers send "prune" messages upstream toward the source.

Copyright © 2010 Internetwork Expert

Cisco IOS routers do not implement complete DVMRP stacks and rely on PIM for multicast routing and signaling. However, Cisco routers might be configured to border with a DVMRP domain and receive DVMRP routing updates. IOS routers are capable of storing DVMRP information and using it to perform RPF checks on packets received from the DVMRP cloud. At the same time, the routers will generate DVMRP updates to cover sources in the PIM cloud and let DVMRP systems receive multicast feeds. IOS routers may generate DVMRP prune and graft messages in response to the respective PIM messages.

Use the command **ip dvmrp interoperability** to configure the IOS router for interoperation with DVMRP systems. This will allow the router to accept/send DVMRP updates and populate the multicast route cache. In order to use this information for a particular interface, enter the **ip dvmrp unicast-routing** interface-level command. This will make the router prefer cached DVMRP information over unicast routes for RPF checks. Like RIP, DVMRP performs automatic summarization when crossing major subnet boundaries. You may disable this behavior, using the command **no ip dvmrp auto-summary**.

By default, the local router will only advertise directly connected subnets in DVMRP updates. If you want to advertise more information, use the interface-level command **ip dvmrp metric <hops> [list <access-list>] [protocol <process-id>]** to redistribute static subnets or IGP networks to all DVMRP neighbors on the interface. If you omit the protocol specification, the command will only advertise connected routes. If you want to filter out certain updates, use the metric value of zero to remove the matching subnets from the DVMRP updates.

Another thing you might want to configure is a DVMRP tunnel. DVMRP tunnels are supported on IOS routers to connect to remote DVMRP clouds over non-multicast networks. Notice that you cannot configure a DMVRP tunnel between two IOS routers, and these configurations are always unidirectional. DMVRP tunnels are commonly used to link your multicast network with the MBONE. Use the following syntax to configure a DVMRP tunnel:

Copyright © 2010 Internetwork Expert

```
interface tunnel 0
 ip unnumbered Loopback0
 ip pim dense-mode
 tunnel source Loopback0
 tunnel destination <IP in MBONE>
 tunnel mode dvmrp
```

Notice that PIM is enabled on the interface, to allow multicast feeds to flow to the MBONE, even though no real PIM adjacencies are ever established over the tunnel.

Copyright © 2010 Internetwork Expert

```
R4:
ip dvmrp interoperability
!
access-list 40 permit 155.1.0.0 0.0.255.255
!
interface FastEthernet0/1
 ip dvmrp metric 1 list 40 eigrp 100
!
interface tunnel 0
 ip unnumbered Loopback0
 ip pim dense-mode
 tunnel source Loopback0
 tunnel destination 204.12.1.100
 tunnel mode dvmrp
```

Verification

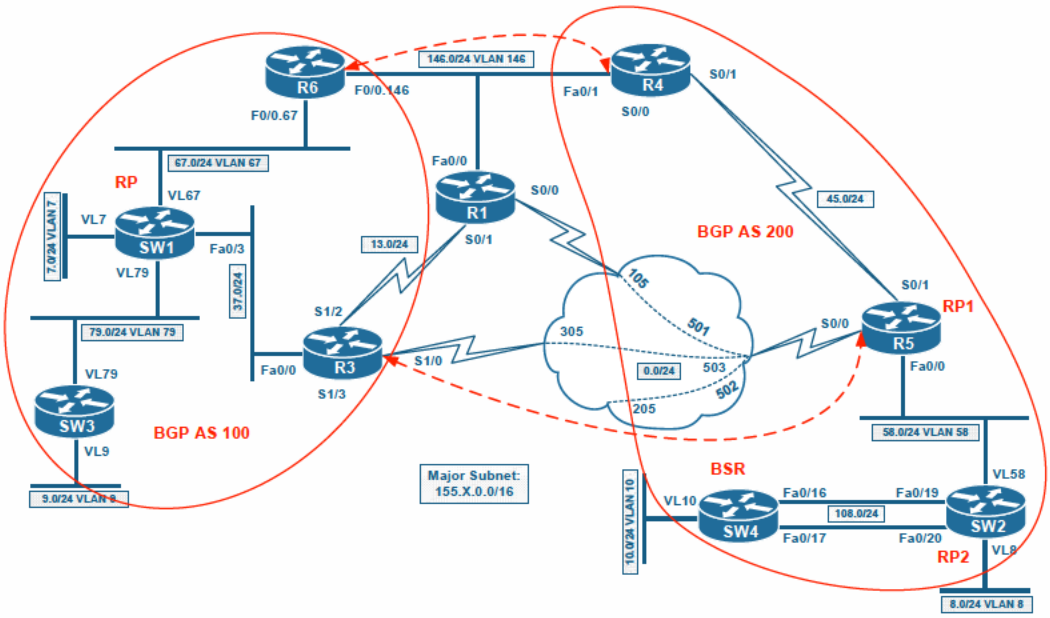
> **Note**
>
> There is no way to fully verify DVMRP interoperability unless you have a real DVMRP router. However, you may verify that the router actually generates DVMRP updates using debug commands.

```
Rack1R4#debug ip dvmrp detail
DVMRP(0): Building Report for FastEthernet0/1
DVMRP(0): Report 155.1.146.0/24, metric 32
DVMRP(0): Report 155.1.10.0/24, metric 1
DVMRP(0): Report 155.1.8.0/24, metric 1
DVMRP(0): Report 155.1.5.0/24, metric 1
DVMRP(0): Report 155.1.58.0/24, metric 1
DVMRP(0): Report 155.1.45.0/24, metric 1
DVMRP(0): Report 155.1.67.0/24, metric 32
DVMRP(0): Report 155.1.108.0/24, metric 1
DVMRP(0): Report 150.1.6.0/24, metric 32
DVMRP(0): Report 150.1.5.0/24, metric 1
DVMRP(0): Report 150.1.10.0/24, metric 1
DVMRP(0): Report 150.1.8.0/24, metric 1
DVMRP(0): Delay Report on FastEthernet0/1
DVMRP(0): 12 unicast, 0 MBGP, 0 DVMRP routes advertised
DVMRP(0): Send Report on FastEthernet0/1 to 224.0.0.4
```

Copyright © 2010 Internetwork Expert

Note

At this point, erase all devices running configs and load the *Inter Domain Multicast* initial configuration files. Refer to the diagram below, when working with the tasks that follow.



Copyright © 2010 Internetwork Expert

8.29 Multicast BGP Extension

- Enable multicast exchange between AS 100 and AS 200 on both peering links.
- Multicast traffic should prefer to be routed across the Frame-Relay link, using the Fast Ethernet link as backup.

Configuration

Note

A multicast BGP extension is commonly needed when you plan to exchange multicast traffic between two different administrative domains, i.e. different autonomous systems. To achieve this goal, you need to fulfill the following tasks

1) Enable PIM between the two domains, to allow signaling of shared and shortest-path trees between them.

1.1) PIM SM is most often used for multicast traffic exchange between different domains. Each domain usually has its own set of RPs, and thus you should prevent BSR/Auto-RP information from leaking between the domains.

1.2) You must exchange information about active sources between the RPs in every domain. Since the RPs are separated, one domain cannot easily learn about the sources in another domain. As we'll see later, a special protocol called MSDP is used for this purpose.

2) In order to facilitate multicast traffic forwarding, you need to exchange information on routes towards the multicast sources in each domain, to allow routers performing RPF checks to do so correctly. PIM uses the unicast routing table to perform these RPF checks, and thus it may use routes learned via either IGP or BGP. However it is BGP is most commonly used to exchange this routing information.

Copyright © 2010 Internetwork Expert

In some cases, you may want to apply different policies to unicast specific routes exchanged via BGP as well as to the information about multicast sources. This is possible thanks to Multi-Protocol BGP extensions. Using a special address family, you may exchange prefixes under the "multicast" address-family, and apply a different policy to this information. These prefixes are interpreted in the same as the **mroute** command information – they are used for RPF checks on the router that receives them. That is, if a prefix is learned via multicast BGP extension, it is assumed to have RPF neighbor towards the next-hop IP address found in the update. If needed, BGP performs recursive routing lookups for the next hop via the IGP routing table to find the immediate RPF neighbor. Unlike the **mroute** command, which is purely local, the information is propagated via BGP to every neighbor configured for the multicast address family.

Using separate policies for multicast inter-domain RPF information allows the use of different inter-domain links for unicast and multicast traffic. Or you may selectively filter out certain multicast sources from another domain, while leaving unicast routes intact.

These tasks require us to enable BGP multicast extensions an all BGP routers. Notice the use of peer-group under the multicast address family. This is needed to propagate multicast RPF information through both domains, as route-reflection is configured separately per address family. Notice the use of AS-PATH prepending to designate the primary path. Multicast prefixes are subject to the same best-path selection procedure, and thus you may use the same methods of path manipulation you used with unicast prefixes. Finally, PIM is activated on the links connecting the two autonomous systems. The PIM BSR border command is used to stop BSR information from leaking into AS 100.

Copyright © 2010 Internetwork Expert

R3:

```
router bgp 100
 address-family ipv4 multicast
 neighbor 155.1.0.5 activate
 redistribute ospf 1
 neighbor 150.1.7.7 activate
 neighbor 150.1.7.7 next-hop-self
!
interface Serial 1/0.1
 ip pim sparse-mode
```

R4:

```
route-map PREPEND
 set as-path prepend 200 200 200
!
router bgp 200
 address-family ipv4 multicast
 neighbor 155.1.146.6 activate
 redistribute eigrp 100
 neighbor 155.1.146.6 route-map PREPEND out
 neighbor 150.1.5.5 activate
!
interface FastEthernet 0/1
 ip pim sparse-mode
 ip pim bsr-border
```

Copyright © 2010 Internetwork Expert

R5:
```
router bgp 200
 address-family ipv4 multicast
 neighbor 155.1.0.3 activate
 redistribute eigrp 100
 neighbor IBGP route-reflector-client
 neighbor 150.1.4.4 peer-group IBGP
 neighbor 150.1.8.8 peer-group IBGP
 neighbor 150.1.10.10 peer-group IBGP
 neighbor IBGP next-hop-self
!
interface Serial 0/0/0
 ip pim sparse-mode
 ip pim bsr-border
```

R6:
```
route-map PREPEND
 set as-path prepend 100 100 100
!
router bgp 100
 address-family ipv4 multicast
 neighbor 155.1.146.4 activate
 redistribute ospf 1
 neighbor 155.1.146.4 route-map PREPEND out
 neighbor 150.1.7.7 activate
 neighbor 150.1.7.7 next-hop-self
!
interface FastEthernet 0/0.146
 ip pim sparse-mode
```

SW1:
```
router bgp 100
 address-family ipv4 multicast
 neighbor IBGP route-reflector-client
 neighbor 150.1.3.3 peer-group IBGP
 neighbor 150.1.6.6 peer-group IBGP
 neighbor 150.1.9.9 peer-group IBGP
```

SW2:
```
router bgp 200
 address-family ipv4 multicast
 neighbor 150.1.5.5 activate
```

SW3:
```
router bgp 100
 address-family ipv4 multicast
 neighbor 150.1.7.7 activate
```

SW4:
```
router bgp 200
 address-family ipv4 multicast
 neighbor 150.1.5.5 activate
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Use the regular show BGP commands to check that the multicast address family is activated between the routers. Repeat it on every BGP router to make sure you didn't miss anything.

```
Rack1R6#show ip bgp ipv4 multicast summary
BGP router identifier 150.1.6.6, local AS number 100
BGP table version is 104, main routing table version 104
19 network entries using 2223 bytes of memory
19 path entries using 912 bytes of memory
24/11 BGP path/bestpath attribute entries using 2976 bytes of memory
1 BGP rrinfo entries using 24 bytes of memory
2 BGP AS-PATH entries using 48 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 6183 total bytes of memory
BGP activity 103/64 prefixes, 351/295 paths, scan interval 60 secs

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down
State/PfxRcd
150.1.7.7       4   100     362     454      104    0    0 00:09:36
0
155.1.146.4     4   200     294     258      104    0    0 01:10:05
10
```

> **Note**
>
> Before testing can proceed we need to ensure that EIGRP is running on the S0/1/0 interface of R5.
>
> router eigrp 100
> network 155.1.45.5 0.0.0.0
>
> Next, check the BGP tables on the border routers to make sure that best paths toward the multicast prefixes are across the Frame-Relay cloud:

```
Rack1R4#show ip bgp ipv4 multicast regexp 100$
BGP table version is 91, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  150.1.3.0/24     155.1.146.6              3             0 100 100 100 100 ?
*>i                 150.1.5.5                0    100      0 100 ?
*> 150.1.6.0/24     155.1.146.6              0             0 100 100 100 100 ?
*>i150.1.6.6/32     150.1.5.5                3    100      0 100 ?
*>i150.1.7.0/24     150.1.5.5                2    100      0 100 ?
```

Copyright © 2010 Internetwork Expert

```
*                  155.1.146.6              2             0 100 100 100 100 ?
*>i150.1.9.9/32    150.1.5.5                3    100      0 100 ?
*                  155.1.146.6              3             0 100 100 100 100 ?
*>i155.1.7.0/24    150.1.5.5                2    100      0 100 ?
*                  155.1.146.6              2             0 100 100 100 100 ?
*>i155.1.9.0/24    150.1.5.5                3    100      0 100 ?
*                  155.1.146.6              3             0 100 100 100 100 ?
*>i155.1.37.0/24   150.1.5.5                0    100      0 100 ?
*                  155.1.146.6              2             0 100 100 100 100 ?
*>i155.1.67.0/24   150.1.5.5                2    100      0 100 ?
*                  155.1.146.6              0             0 100 100 100 100 ?
*>i155.1.79.0/24   150.1.5.5                2    100      0 100 ?
   Network          Next Hop            Metric LocPrf Weight Path
*                  155.1.146.6              2             0 100 100 100 100 ?
```

Check the best paths in the other AS also.

```
Rack1R6#show ip bgp ipv4 multicast regexp 200$
BGP table version is 115, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i150.1.4.0/24    150.1.3.3           2297856    100      0 200 ?
*                  155.1.146.4               0             0 200 200 200 200 ?
*>i150.1.5.0/24    150.1.3.3                 0    100      0 200 ?
*                  155.1.146.4         2297856             0 200 200 200 200 ?
*>i150.1.8.0/24    150.1.3.3            156160    100      0 200 ?
*                  155.1.146.4         2300416             0 200 200 200 200 ?
*>i150.1.10.0/24   150.1.3.3            158720    100      0 200 ?
*                  155.1.146.4         2302976             0 200 200 200 200 ?
*>i155.1.0.0/24    150.1.3.3                 0    100      0 200 ?
*                  155.1.146.4               0             0 200 200 200 200 ?
*>i155.1.8.0/24    150.1.3.3             28416    100      0 200 ?
*                  155.1.146.4         2172672             0 200 200 200 200 ?
*>i155.1.10.0/24   150.1.3.3             30976    100      0 200 ?
*                  155.1.146.4         2175232             0 200 200 200 200 ?
*>i155.1.45.0/24   150.1.3.3                 0    100      0 200 ?
*                  155.1.146.4               0             0 200 200 200 200 ?
*>i155.1.58.0/24   150.1.3.3                 0    100      0 200 ?
   Network          Next Hop            Metric LocPrf Weight Path
*                  155.1.146.4         2172416             0 200 200 200 200 ?
*>i155.1.108.0/24  150.1.3.3             30720    100      0 200 ?
*                  155.1.146.4         2174976             0 200 200 200 200 ?
```

Copyright © 2010 Internetwork Expert

8.30 MSDP

- Change PIM dense-mode on all links where it's configured to PIM sparse-mode.
- Configure R5 as the RP for AS 200 and SW1 as the RP for AS 100. Use the BSR method to distribute RP information, and configure BSR border on the link between R3 and SW1.
- Create an MSDP peering session between SW1 and R5 sourcing it off the Loopback 0 interfaces.

Configuration

Note

When implementing inter-domain multicast using PIM SM, each domain usually has its own RP. In order to allow sources and receivers from different domains to locate each other, RPs need to exchange the information about their local active sources. After this information is exchanged between the RPs, all routers that joined the respective shared trees may build shortest-path trees toward the actual sources.

MSDP or Multicast Source Discovery Protocol is used to exchange multicast source information between RPs. It is configured as a TCP connection between the RPs, and used to exchange the so-called Source Active (SA) messages. Note that all MSDP peerings are configured manually, using the command **`ip msdp peer`** at both endpoints. When a source in one PIM SM domain starts sending the multicast traffic, the respective DR will start the registration process with the local RP. When the local RP receives the PIM Register message, it replicates it to all of its MSDP neighbors as an SA message. The SA message contains the IP address of the multicast source as well as the destination group and the IP address of the RP sending the SA message. The latter is known as the MSDP ID and can be changed using the command **`ip msdp originator-id`**.

When any RP receives a new SA message, it checks if there are local receivers that have joined the shared tree for the encapsulated group. If there are any, the message is forwarded down the tree, allowing the receivers to learn about the sources in another domain. After that, the receivers might join the SPT toward the source in the other domain. This is only possible if the source IP address is learned via BGP or some other inter-domain route exchange procedure. Till the moment there is an active source in a domain, the respective RP will forward periodic SA messages with an empty payload to refresh the active state for this group/source in all other domains.

Copyright © 2010 Internetwork Expert

MSDP allows us to connect RPs in an arbitrary meshed topology, to include loops. In order to prevent the SA messages from cycling the topology, as a result of these loops, every MSDP peer forwards SA messages only after they pass the RPF check. The RPF check is performed based on the RP IP address (originator-ID) inside the message and the IP address of the MSDP peer that relayed the message. If the MSDP peer is on the shortest path towards the originating RP, the message is accepted, otherwise it is dropped. This RPF check requires full routing information from other domains to in order to discover routes to other RPs. If you have a stub multicast domain, lacking full BGP information, you may use the command **ip msdp default-peer** to identify the upstream RP that forwards SA messages. RPF checks are not applied to default peers and all SA messages are accepted.

Our scenario is a bit tricky, as it has two RPs in AS 200. However, the BSR protocol ensures that all routers in AS 200 will select the same RP for a given group. Thus you only need to peer SW1 with SW2 and R5 via MSDP, but there is no need to peer SW2 with R5 via MSDP.

```
R5:
ip msdp peer 150.1.7.7 connect-source Loopback 0 remote-as 100

SW2:
ip msdp peer 150.1.7.7 connect-source Loopback 0 remote-as 100

SW1:

ip msdp peer 150.1.5.5 connect-source Loopback 0 remote-as 200
ip msdp peer 150.1.8.8 connect-source Loopback 0 remote-as 200
```

Copyright © 2010 Internetwork Expert

Verification

Note

We want to ensure that we have two multicast groups that map to different RPs inside AS 200. In order make this happen we will have to alter the "rp-hash" value used on SW4:

Rack1SW4(config)#no ip pim bsr-candidate loopback0 0
Rack1SW4(config)#ip pim bsr-candidate loopback0 31

Now the groups 239.1.1.1 and 239.1.1.2 map to RPs R5 and SW2 respectively:

```
Rack1R4#show ip pim rp-hash 239.1.1.1
  RP 150.1.5.5 (?), v2
    Info source: 150.1.10.10 (?), via bootstrap, priority 0, holdtime
90
         Uptime: 00:08:12, expires: 00:01:17
  PIMv2 Hash Value (mask 255.255.255.254)
    RP 150.1.5.5, via bootstrap, priority 0, hash value 1362971077
    RP 150.1.8.8, via bootstrap, priority 0, hash value 718054422

Rack1R4#show ip pim rp-hash 239.1.1.2
  RP 150.1.8.8 (?), v2
    Info source: 150.1.10.10 (?), via bootstrap, priority 0, holdtime
202
         Uptime: 00:08:14, expires: 00:03:03
  PIMv2 Hash Value (mask 255.255.255.254)
    RP 150.1.5.5, via bootstrap, priority 0, hash value 443334807
    RP 150.1.8.8, via bootstrap, priority 0, hash value 1364246456
```

Note

Now configure speakers on both systems to join these groups.

R4:

```
interface Loopback0
 ip pim sparse-mode
 ip igmp join-group 239.1.1.1
 ip igmp join-group 239.1.1.2
```

SW3:

```
interface Loopback0
 ip pim sparse-mode
 ip igmp join-group 239.1.1.1
 ip igmp join-group 239.1.1.2
```

Copyright © 2010 Internetwork Expert

> **Note**
>
> Initially, every router joins the shared tree in its own domain.

```
Rack1SW3#show ip mroute 239.1.1.1
IP Multicast Routing Table
<snip>

(*, 239.1.1.1), 00:19:49/00:02:22, RP 150.1.7.7, flags: SJCL
  Incoming interface: Vlan79, RPF nbr 155.1.79.7
  Outgoing interface list:
    Loopback0, Forward/Sparse, 00:17:45/00:02:22

Rack1SW3#show ip mroute 239.1.1.2
IP Multicast Routing Table
<snip>

(*, 239.1.1.2), 00:02:07/00:02:24, RP 150.1.7.7, flags: SJCL
  Incoming interface: Vlan79, RPF nbr 155.1.79.7
  Outgoing interface list:
    Loopback0, Forward/Sparse, 00:02:07/00:02:24

Rack1R4#show ip mroute 239.1.1.1
IP Multicast Routing Table
<snip>

(*, 239.1.1.1), 00:12:37/00:02:38, RP 150.1.5.5, flags: SJCL
  Incoming interface: Serial0/1/0, RPF nbr 155.1.45.5
  Outgoing interface list:
    Loopback0, Forward/Sparse, 00:12:37/00:02:38

Rack1R4#show ip mroute 239.1.1.2
IP Multicast Routing Table
<snip>

(*, 239.1.1.2), 00:03:25/00:02:28, RP 150.1.8.8, flags: SJCL
  Incoming interface: Serial0/1/0, RPF nbr 155.1.45.5
  Outgoing interface list:
    Loopback0, Forward/Sparse, 00:03:25/00:02:28
```

Copyright © 2010 Internetwork Expert

Note

Enable MSDP debugging on SW1 and start pinging group 239.1.1.1 from **SW4:**

```
Rack1SW1#debug ip msdp de
Rack1SW1#debug ip msdp detail
MSDP Detail debugging is on

Rack1SW4#ping 239.1.1.1 repeat 1000

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 239.1.1.1, timeout is 2 seconds:

Reply to request 0 from 155.1.45.4, 36 ms
Reply to request 0 from 155.1.79.9, 208 ms
Reply to request 0 from 155.1.79.9, 144 ms
Reply to request 0 from 155.1.79.9, 112 ms
Reply to request 0 from 155.1.45.4, 88 ms
Reply to request 0 from 155.1.45.4, 60 ms
Reply to request 1 from 155.1.45.4, 36 ms
Reply to request 1 from 155.1.79.9, 172 ms
Reply to request 1 from 155.1.79.9, 104 ms
Reply to request 1 from 155.1.79.9, 80 ms
Reply to request 1 from 155.1.45.4, 64 ms
Reply to request 1 from 155.1.45.4, 52 ms
Reply to request 2 from 155.1.45.4, 36 ms
Reply to request 2 from 155.1.79.9, 104 ms
```

Note

Notice that SW1 received Source Active messages for the sources on SW4. Since SW4 uses all of its PIM-enabled interfaces to source multicast, there are multiple SA messages for every registered source. The actual source is registered with the RP located in AS 200.

Copyright © 2010 Internetwork Expert

```
Rack1SW1#
MSDP(0): Received 120-byte TCP segment from 150.1.5.5
MSDP(0): Append 120 bytes to 0-byte msg 63 from 150.1.5.5, qs 1
MSDP(0): WAVL Insert SA Source 155.1.10.10 Group 239.1.1.1 RP 150.1.5.5
Successful
MSDP(0): Forward decapsulated SA data for (155.1.10.10, 239.1.1.1) on
Vlan79
MSDP(0): Received 120-byte TCP segment from 150.1.5.5
MSDP(0): Append 120 bytes to 0-byte msg 64 from 150.1.5.5, qs 1
MSDP(0): WAVL Insert SA Source 155.1.108.10 Group 239.1.1.1 RP
150.1.5.5 Successful
MSDP(0): Forward decapsulated SA data for (155.1.108.10, 239.1.1.1) on
Vlan79
MSDP(0): Received 120-byte TCP segment from 150.1.5.5
MSDP(0): Append 120 bytes to 0-byte msg 65 from 150.1.5.5, qs 1
MSDP(0): WAVL Insert SA Source 150.1.10.10 Group 239.1.1.1 RP 150.1.5.5
Successful
MSDP(0): Forward decapsulated SA data for (150.1.10.10, 239.1.1.1) on
Vlan79
MSDP(0): Received 3-byte TCP segment from 150.1.5.5
```

Note

Now check that SW3 has joined the SPTs towards the sources in different the AS. Notice that RPF information for these sources is taken from MBGP updates, not the unicast routing table.

```
Rack1SW3#show ip mroute 239.1.1.1
IP Multicast Routing Table
<snip>

(*, 239.1.1.1), 00:25:31/stopped, RP 150.1.7.7, flags: SJCL
  Incoming interface: Vlan79, RPF nbr 155.1.79.7
  Outgoing interface list:
    Loopback0, Forward/Sparse, 00:23:27/00:02:41

(150.1.10.10, 239.1.1.1), 00:03:09/00:02:58, flags: LJT
  Incoming interface: Vlan79, RPF nbr 155.1.79.7, Mbgp
  Outgoing interface list:
    Loopback0, Forward/Sparse, 00:03:09/00:02:41

(155.1.10.10, 239.1.1.1), 00:03:09/00:02:56, flags: LJT
  Incoming interface: Vlan79, RPF nbr 155.1.79.7, Mbgp
  Outgoing interface list:
    Loopback0, Forward/Sparse, 00:03:09/00:02:40

(155.1.108.10, 239.1.1.1), 00:03:09/00:02:57, flags: LJT
  Incoming interface: Vlan79, RPF nbr 155.1.79.7, Mbgp
  Outgoing interface list:
    Loopback0, Forward/Sparse, 00:03:09/00:02:40
```

Copyright © 2010 Internetwork Expert

Note

Now make sure the SPTs are built across the Frame-Relay link, as this is the preferred path for multicast traffic. Use the **show ip mroute** command to accomplish this.

```
Rack1SW1#show ip mroute 239.1.1.1
IP Multicast Routing Table
<snip>

(*, 239.1.1.1), 00:27:33/00:02:39, RP 150.1.7.7, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Vlan79, Forward/Sparse, 00:25:29/00:02:39

(150.1.10.10, 239.1.1.1), 00:05:11/00:03:21, flags: MT
  Incoming interface: FastEthernet0/3, RPF nbr 155.1.37.3, Mbgp
  Outgoing interface list:
    Vlan79, Forward/Sparse, 00:05:11/00:02:39

(155.1.10.10, 239.1.1.1), 00:05:11/00:02:48, flags: MT
  Incoming interface: FastEthernet0/3, RPF nbr 155.1.37.3, Mbgp
  Outgoing interface list:
    Vlan79, Forward/Sparse, 00:05:11/00:02:38

(155.1.108.10, 239.1.1.1), 00:05:11/00:03:29, flags: MT
  Incoming interface: FastEthernet0/3, RPF nbr 155.1.37.3, Mbgp
  Outgoing interface list:
    Vlan79, Forward/Sparse, 00:05:11/00:02:38

Rack1R3#show ip mroute 239.1.1.1
IP Multicast Routing Table
<snip>

(*, 239.1.1.1), 00:26:55/stopped, RP 150.1.7.7, flags: SP
  Incoming interface: FastEthernet0/0, RPF nbr 155.1.37.7
  Outgoing interface list: Null

(150.1.10.10, 239.1.1.1), 00:05:21/00:03:26, flags: T
  Incoming interface: Serial1/0.1, RPF nbr 155.1.0.5, Mbgp
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:05:21/00:02:33

(155.1.10.10, 239.1.1.1), 00:05:22/00:01:15, flags: T
  Incoming interface: Serial1/0.1, RPF nbr 155.1.0.5, Mbgp
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:05:22/00:02:32

(155.1.108.10, 239.1.1.1), 00:05:22/00:03:24, flags: T
  Incoming interface: Serial1/0.1, RPF nbr 155.1.0.5, Mbgp
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:05:22/00:02:32
```

Copyright © 2010 Internetwork Expert

Note

You may also use the **mtrace** command, to trace the multicast delivery tree from the leaf to the root. The first parameter is the source address and the second parameter is the destination group. This command queries the neighbors for the upstream multicast path and tells you the method used for RPF check at every router. Notice that inside AS 100 the RPF checks are performed using MBGP.

```
Rack1SW1#mtrace 150.1.10.10 239.1.1.1
Type escape sequence to abort.
Mtrace from 150.1.10.10 to 155.1.37.7 via group 239.1.1.1
From source (?) to destination (?)
Querying full reverse path...
 0  155.1.37.7
-1  155.1.37.7 PIM/MBGP Reached RP/Core [150.1.10.0/24]
-2  155.1.37.3 PIM/MBGP  [150.1.10.0/24]
-3  155.1.0.5 [AS 200] PIM Reached RP/Core [150.1.10.0/24]
-4  155.1.58.8 [AS 200] PIM  [150.1.10.0/24]
-5  155.1.108.10 [AS 200] PIM  [150.1.10.0/24]
```

Note

You may now repeat the tests for the group 239.1.1.2 and see that it works as well.

```
Rack1SW4#ping 239.1.1.2 repeat 1000

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 239.1.1.2, timeout is 2 seconds:

Reply to request 0 from 155.1.45.4, 36 ms
Reply to request 0 from 155.1.79.9, 140 ms
Reply to request 0 from 155.1.79.9, 104 ms
Reply to request 0 from 155.1.45.4, 84 ms
Reply to request 0 from 155.1.45.4, 56 ms
Reply to request 1 from 155.1.45.4, 36 ms
Reply to request 1 from 155.1.79.9, 244 ms
Reply to request 1 from 155.1.79.9, 176 ms
Reply to request 1 from 155.1.79.9, 120 ms
Reply to request 1 from 155.1.45.4, 88 ms
Reply to request 1 from 155.1.79.9, 80 ms
Reply to request 1 from 155.1.45.4, 76 ms
Reply to request 1 from 155.1.45.4, 60 ms
```

Copyright © 2010 Internetwork Expert

```
Rack1SW3#mtrace 150.1.10.10 239.1.1.2
Type escape sequence to abort.
Mtrace from 150.1.10.10 to 155.1.79.9 via group 239.1.1.2
From source (?) to destination (?)
Querying full reverse path...
 0  155.1.79.9
-1  155.1.79.9 PIM/MBGP  [150.1.10.0/24]
-2  155.1.79.7 PIM/MBGP Reached RP/Core [150.1.10.0/24]
-3  155.1.37.3 PIM/MBGP  [150.1.10.0/24]
-4  155.1.0.5 [AS 200] PIM  [150.1.10.0/24]
-5  155.1.58.8 [AS 200] PIM Reached RP/Core [150.1.10.0/24]
-6  155.1.108.10 [AS 200] PIM  [150.1.10.0/24]
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

8.31 Anycast RP

- Instead of relying on the BSR protocol to distribute the load between the RPs in AS 200, implement a solution that hides both RPs behind the same RP IP address 150.1.100.100.
- Every RP should inform the other one of the active sources registered with them.

Configuration

Note

Anycast RP is a special RP redundancy scenario, which allows using redundant RPs sharing the *same* IP address. Here Anycast means that groups of RPs use the same IP address used by all multicast routers in the domain to build shared trees. However, the PIM Joins are being sent to the closest RP, based on the unicast routing table. Thus, different routers might join shared trees rooted at different RPs. At the same time, different DRs will pick up different physical RPs based on the anycast address to register their local sources.

In order to maintain consistent source information, MSDP sessions should be configured between the RPs. This will ensure that all routers joining different RPs will still have full information about all potential sources in the domain. Thus, the following are the guidelines to configure Anycast RP:

1) Use the same IP address on all routers as the candidate RP IP address. Propagate this information via BSR or Auto-RP.
2) Using different IP addresses on every router, source MSDP sessions and link all candidate RPs in a mesh. Note that you might need to manually specify the MSDP originator ID to be different on every RP, or the MSDP sessions won't come up.

Anycast RP is a purely intra-domain solution, and does not deal with inter-domain multicast. Thus, it is a good example of using the inter-domain technology inside a single multicast region to achieve RP redundancy above the scheme used by PIM BSR.

In our scenario, we mix intra-domain MSDP with inter-domain connections. That is, a domain with Anycast RP peers in another domain with regular RPs. This results in a looped MSDP topology, which will successfully work due to MSDP RPF checks.

Copyright © 2010 Internetwork Expert

R5:

```
interface Loopback100
 ip address 150.1.100.100 255.255.255.255
 ip pim sparse-mode
!
router eigrp 100
 network 150.1.100.100 0.0.0.0
!
ip msdp originator-id Loopback 0
ip msdp peer 150.1.8.8 connect-source Loopback 0
no ip pim rp-candidate Loopback0
ip pim rp-candidate Loopback100
```

SW2:

```
interface Loopback100
 ip address 150.1.100.100 255.255.255.255
 ip pim sparse-mode
!
router eigrp 100
 network 150.1.100.100 0.0.0.0
!
ip msdp originator-id Loopback 0
ip msdp peer 150.1.5.5 connect-source Loopback 0
ip pim rp-candidate Loopback0
ip pim rp-candidate Loopback100
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> First, join receivers on R4 and SW3 to the multicast group 239.1.1.1. Check that R4 actually uses the Anycast RP IP address as its RP.

R4:

```
interface Loopback0
 ip pim sparse-mode
 ip igmp join-group 239.1.1.1
```

SW3:

```
interface Loopback0
 ip pim sparse-mode
 ip igmp join-group 239.1.1.1
```

```
Rack1R4#show ip mroute 239.1.1.1
IP Multicast Routing Table
<snip>

(*, 239.1.1.1), 00:39:32/00:02:38, RP 150.1.100.100, flags: SJCL
  Incoming interface: Serial0/1/0, RPF nbr 155.1.45.5
  Outgoing interface list:
    Loopback0, Forward/Sparse, 00:39:32/00:02:38
```

> **Note**
>
> Next source multicast from SW4 and confirm that it actually reaches the receivers.

```
Rack1SW4#ping 239.1.1.2 repeat 1000

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 239.1.1.2, timeout is 2 seconds:

Reply to request 0 from 155.1.45.4, 36 ms
Reply to request 0 from 155.1.79.9, 140 ms
Reply to request 0 from 155.1.79.9, 104 ms
Reply to request 0 from 155.1.45.4, 84 ms
Reply to request 0 from 155.1.45.4, 56 ms
Reply to request 1 from 155.1.45.4, 36 ms
```

Copyright © 2010 Internetwork Expert

Note

Look at the SA caches of R5 and SW1. Both of them should have been updated by SW2, as SW2 is the closest RP to the source (SW4) and the source registers with it.

```
Rack1R5#show ip msdp sa-cache
MSDP Source-Active Cache - 3 entries
(150.1.10.10, 239.1.1.1), RP 150.1.8.8, MBGP/AS 200, 00:00:26/00:05:34, Peer
150.1.8.8
(155.1.10.10, 239.1.1.1), RP 150.1.8.8, MBGP/AS 200, 00:00:26/00:05:34, Peer
150.1.8.8
(155.1.108.10, 239.1.1.1), RP 150.1.8.8, MBGP/AS 200, 00:00:25/00:05:34, Peer
150.1.8.8

Rack1SW1#show ip msdp sa-cache
MSDP Source-Active Cache - 3 entries
(150.1.10.10, 239.1.1.1), RP 150.1.8.8, MBGP/AS 200, 00:00:53/00:05:33, Peer
150.1.8.8
(155.1.10.10, 239.1.1.1), RP 150.1.8.8, MBGP/AS 200, 00:00:53/00:05:33, Peer
150.1.8.8
(155.1.108.10, 239.1.1.1), RP 150.1.8.8, MBGP/AS 200, 00:00:53/00:05:33, Peer
150.1.8.8
```

Note

Now source traffic from AS 100 and make sure both receivers are able to hear it. Check the SA caches of SW2 and R5 after that, to confirm that SW1 has actually updated them.

```
Rack1R6#ping 239.1.1.1 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 239.1.1.1, timeout is 2 seconds:

Reply to request 0 from 155.1.79.9, 8 ms
Reply to request 0 from 155.1.45.4, 69 ms
Reply to request 0 from 155.1.146.4, 8 ms
Reply to request 1 from 155.1.79.9, 4 ms
Reply to request 1 from 155.1.45.4, 48 ms
Reply to request 1 from 155.1.146.4, 4 ms
Reply to request 2 from 155.1.79.9, 4 ms
Reply to request 2 from 155.1.45.4, 44 ms
Reply to request 2 from 155.1.146.4, 4 ms
Reply to request 3 from 155.1.79.9, 4 ms
Reply to request 3 from 155.1.45.4, 44 ms
Reply to request 3 from 155.1.146.4, 4 ms
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip msdp sa-cache
MSDP Source-Active Cache - 4 entries
(150.1.10.10, 239.1.1.1), RP 150.1.8.8, MBGP/AS 200, 00:09:41/00:05:51, Peer
150.1.8.8
(155.1.10.10, 239.1.1.1), RP 150.1.8.8, MBGP/AS 200, 00:09:41/00:05:51, Peer
150.1.8.8
(155.1.67.6, 239.1.1.1), RP 150.1.7.7, MBGP/AS 100, 00:00:11/00:05:48,
Peer 150.1.7.7
(155.1.108.10, 239.1.1.1), RP 150.1.8.8, MBGP/AS 200, 00:09:40/00:05:51, Peer
150.1.8.8

Rack1SW2#show ip msdp sa-cache
MSDP Source-Active Cache - 1 entries
(155.1.67.6, 239.1.1.1), RP 150.1.7.7, MBGP/AS 100, 00:00:16/00:05:43, Peer
150.1.7.7
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Note

At this point, erase all devices running configs and load the *Basic IP Addressing* initial configuration files.

8.32 Catalyst IGMP Snooping

- Configure R6's Fa0/0 interface with the ip address 155.1.146.6/24
- Join the multicast group 239.1.1.100 on the VLAN 146 interfaces of R4 and R6.
- Configure the switches to assist in optimal traffic flooding across the switched topology for VLAN 146.
- Ensure a switch stops flooding out of a given port as soon as it receives an IGMPv2 report.

Configuration

Note

Multicast poses special difficulties in switched Ethernet networks. By default, IP Multicast addresses are mapped to Ethernet multicast MAC addresses per the following procedure:

1) The multicast Ethernet MAC address range starts at 01:00:5E:00:00:00 and goes through 01:00:5E:7F:FF:FF. The highest bit of the 4th byte in these addresses is fixed at 1, and thus only low-order 23 bits can vary. This allocation is historical.

2) Based on this range, the low-order 23 bits of the multicast IP address are mapped into the low-order 23 bits of the MAC address.

3) The high order 4 bits of the Layer 3 IP address is fixed to 1110 to indicate the Class D address space between 224.0.0.0 and 239.255.255.255. Thus, 28 bits remain for IP Multicast group numbering.

Copyright © 2010 Internetwork Expert

Therefore, there are 2^5 multicast groups mapped to the same multicast MAC address. However, this is only a part of the puzzle. As you remember, Ethernet switches flood multicast traffic out of all ports by default. In heavily loaded networks, this might result in excessive bandwidth usage. In order to overcome this issue, it's possible for switches to listen to IGMP and PIM messages received on Layer 2 ports. Based on the information snooped from these messages, switches may selectively prune some ports from unneeded multicast traffic. This procedure is called IGMP snooping and allows for selective multicast flooding in switched networks.

Notice that in order to work effectively, IGMP snooping must be implemented on Layer 3 capable switches. This is because Layer 2 devices cannot distinguish IGMP and PIM packets from any other multicast frames, and must process all multicast traffic at the CPU level. Thus, a layer 2 switch's performance might be severely impacted by intense multicast flows.

IGMP snooping is enabled by default on Catalyst multi-layer switches. Snooping is enabled globally and on a per-VLAN basis. If you want to disable IGMP snooping globally on all VLANs, use the command **`no ip igmp snooping`**. Use the command **`no ip igmp snooping vlan <VLAN-ID>`** to disable it for a single VLAN. Generally, switches automatically discover switchports connected to multicast-capable routers by listening to PIM and DVMRP messages, and flood all multicast groups on such ports. If you want to statically configure a port as connected to a multicast router, use the command **`ip igmp snooping vlan <vlan-id> mrouter interface <interface-id>`**.

If your switch has just one host connected to every layer 2 switch-port, you may want to enable the IGMP Snooping Immediate Leave feature using the command **`ip igmp snooping vlan <vlan-id> immediate-leave`**. When you enable this feature, the switch will immediately remove a port from the flood list for a given group once it receives the IGMPv2 Leave message on an interface that is part of a particular group.

Copyright © 2010 Internetwork Expert

R1:
```
ip multicast-routing
!
interface FastEthernet0/0
 ip address 155.1.146.1 255.255.255.0
 ip pim dense-mode
 ip igmp join-group 239.1.1.100
```

R4:
```
ip multicast-routing
!
interface FastEthernet0/1
 ip address 155.1.146.4 255.255.255.0
 ip pim dense-mode
 ip igmp join-group 239.1.1.100
```

R6:
```
ip multicast-routing
!
interface FastEthernet0/0
 ip address 155.1.146.6 255.255.255.0
 ip pim dense-mode
 ip igmp join-group 239.1.1.100
```

SW1:
```
ip igmp snooping vlan 146 immediate-leave
!
interface FastEthernet0/1
 switchport access vlan 146
```
SW2:
```
ip igmp snooping vlan 146 immediate-leave
!
interface FastEthernet0/6
 switchport access vlan 146
```

SW4:
```
ip igmp snooping vlan 146 immediate-leave
!
interface FastEthernet0/4
 switchport access vlan 146
!
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> First, check the IGMP Snooping general settings for VLAN 146. Notice that "IGMPv2 immediate leave" is enabled for this VLAN.

```
Rack1SW1#show ip igmp snooping vlan 146
Global IGMP Snooping configuration:
-----------------------------------
IGMP snooping              : Enabled
IGMPv3 snooping (minimal)  : Enabled
Report suppression         : Enabled
TCN solicit query          : Disabled
TCN flood query count      : 2
Last Member Query Interval : 1000

Vlan 146:
--------
IGMP snooping                          : Enabled
IGMPv2 immediate leave                 : Enabled
Explicit host tracking                 : Enabled
Multicast router learning mode         : pim-dvmrp
Last Member Query Interval             : 1000
CGMP interoperability mode             : IGMP_ONLY
```

> **Note**
>
> Now join R1's VLAN 146 interface to the IGMP group 239.1.1.100 and check that IGMP snooping actually processes the IGMP packets. From the commands output, you can see that there are two ports in the flood list for the group – one for the receiver, and the other for the multicast router (R4).

```
R1:
interface FastEthernet 0/0
 ip igmp join-group 239.1.1.100

Rack1SW1#show ip igmp snooping groups vlan 146
Vlan      Group             Type        Version     Port List
------------------------------------------------------------------
146        239.1.1.100        igmp         v2          Fa0/1, Fa0/19

Rack1SW1#show ip igmp snooping mrouter vlan 146
Vlan    ports
----    -----
 146     Fa0/1(dynamic), Fa0/19(dynamic)

Rack1SW4#show ip igmp snooping groups vlan 146
Vlan      Group             Version     Port List
------------------------------------------------------------
146       239.1.1.100       v2          Fa0/4, Fa0/13, Fa0/16
```

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show ip igmp snooping mrouter vlan 146
Vlan    ports
----    -----
 146     Fa0/4(dynamic), Fa0/13(dynamic), Fa0/16(dynamic)
```

Note

From R1 ping the multicast group:

R1:
Ping 239.1.1.100 repeat 1000

Multicast router information is learned via PIM messages. There are two multicast routers on VLAN 146 – R4 and R6. Confirm that R4 actually receives the IGMP report from R1.

```
Rack1R4#show ip igmp groups
IGMP Connected Group Membership
Group Address    Interface                Uptime    Expires   Last Reporter
Group Accounted
239.1.1.100      FastEthernet0/1          00:15:59  00:02:20
155.1.146.1
<snip>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

8.33 Catalyst Multicast VLAN Registration

- Configure SW1 so that multicast traffic sent by R1 on VLAN 146 is received by R5 on VLAN 58.
- Use the configuration that allows R1 to dynamically tracking sources in other VLANs.

Configuration

Note

Multicast VLAN registration (MVR) is a special type of multicast traffic delivery suited to the access layer of metro Ethernet networks, especially for ring topologies. In these networks, it is common to allocate a VLAN based on a network's geographic region. In this configuration, when you send a video multicast feed to receivers on multiple VLANs, the feed will be replicated for every VLAN across the ring topology, causing bandwidth over-utilization.

MVR uses a single dedicated VLAN across the whole ring to deliver a multicast feed to all receivers. The actual receivers reside in different VLANs, but the switches intercept their IGMP Join requests and pull the multicast feed from the MVR VLAN to the receiver VLAN. Thus, MVR allows multicast traffic to cross VLAN boundaries based on client IGMP reports. This function is similar to IGMP snooping, but the two work independently. You can have both features enable on the switch, but MVR will only inspect IGMP reports for groups explicitly configured to be supported by MVR.

There are two modes for the MVR feature: dynamic and compatible. Before we look into the differences, notice that that MVR and multicast routing are mutually exclusive. When MVR is enabled, the switch performs like a Layer 2 device with respect to multicast traffic, and multicast-routing should remain disabled. However, the switch still inspects IGMP messages and may forward them to the port where a multicast router is connected or not. Back to the modes - the first mode is called dynamic. This mode allows the multicast router connected to the switch ring to listen to the IGMP messages and dynamically create respective mroute states. That is, IGMP join messages from the clients are forwarded to the multicast source port, allowing the broadcast of unneeded channels to be stopped. The other mode, called compatible (the default in switches), inspects the IGMP messages but does not forward them to multicast router ports. This was the default mode of operation of the MVR feature in 3500XL switches. If you use compatible mode, make sure you configured static IGMP joins on the multicast router to feed the multicast streams down. The default mode assumes that the source constantly sends down all multicast channels, and does not process client IGMP joins.

Copyright © 2010 Internetwork Expert

Configure MVR by follow the steps below:

Step 1:
Enable MVR globally using the commands **mvr** and **mvr group <mcast-group> <count>.** This will enable the MVR feature for the particular group or for the number of groups specified by the **<count>** argument and starting at **<mcast-group>** address. Note that you cannot have more than 256 address configured for MVR, and they should never alias. Here aliasing means that two multicast IP addresses map to the same multicast MAC address (see the previous task). For example addresses 228.1.1.1 and 230.1.1.1 would alias to the same MAC address and would be rejected.

Step 2:
Define the MVR VLAN – this is the VLAN that will span multiple switches and carry the actual multicast traffic feed. This VLAN should be allowed on all trunks to permit multicast delivery. The command is **mvr vlan <vlan-id>**. You may also define the MVR mode by using the command **mvr {dynamic|compatible}**

Step 3:
Configure the source and receiver interfaces, using the interface-mode command **mvr type {source|receiver}**. Additionally, you may configure ports for immediate leave, using the command **mvr immediate**. When this command is enabled, any single IGMP Leave would cause the switch to prune the port from receiving multicast feeds. This is good for the ports that have only one host connected.

Step 4:
Optionally, configure static group joins using the command **mvr vlan <vlan-id> group <ip-address>** on receiver ports. This is similar to the static join command configured ona router, but it allows pulling multicast traffic from the MVR VLAN. Another optional command is **mvr querytime <1/10 of second>**. This command is similar to **the igmp query-max-response-time** configured on routers. The switch passively listens to IGMP general queries sent from multicast routers, and then starts the **querytime** timer, waiting for client replies. If no replies are received during the interval, the switch prunes the port from the list of output ports for the group.

Notice that you cannot configure trunk ports as MVR receivers.

Copyright © 2010 Internetwork Expert

SW1:

```
no ip multicast-routing distributed
mvr
mvr vlan 146
mvr group 239.1.1.100
mvr mode dynamic
!
interface FastEthernet 0/1
 mvr type source
!
interface FastEthernet 0/5
 mvr type receiver
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the MVR settings on SW1. Since the mode is dynamic, R1 should receive IGMP Joins from R5.

```
Rack1SW1#show mvr
MVR Running: TRUE
MVR multicast VLAN: 146
MVR Max Multicast Groups: 256
MVR Current multicast groups: 1
MVR Global query response time: 5 (tenths of sec)
MVR Mode: dynamic

Rack1SW1#show mvr interface
Port      Type        Status           Immediate Leave
----      ----        ------           ---------------
Fa0/1     SOURCE      ACTIVE/UP        DISABLED
Fa0/5     RECEIVER    ACTIVE/UP        DISABLED

Rack1SW1#show mvr members
MVR Group IP    Status          Members
------------    ------          -------
239.001.001.100 ACTIVE          Fa0/5(d)

Rack1R1#show ip igmp groups
IGMP Connected Group Membership
Group Address    Interface                Uptime    Expires   Last
Reporter   Group Accounted
239.1.1.100      FastEthernet0/0          00:18:04  00:02:56
155.1.58.5
224.0.1.40       FastEthernet0/0          00:29:42  00:02:50
155.1.146.1
```

> **Note**
>
> Now source the multicast feed from R1 and check that R5 actually receives the multicast packets by using the appropriate debug command. Notice that you will need to disable multicast route caching on R5 for this to work.

Copyright © 2010 Internetwork Expert

```
Rack1R1#ping 239.1.1.100 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 239.1.1.100, timeout is 2 seconds:
...

Rack1R5#debug ip mpacket
IP multicast packets debugging is on
Rack1R5#
IP(0): s=155.1.146.1 (FastEthernet0/0) d=239.1.1.100 id=89, ttl=253,
prot=1, len=114(100), RPF lookup failed for source or RP
Rack1R5#
IP(0): s=155.1.146.1 (FastEthernet0/0) d=239.1.1.100 id=90, ttl=253,
prot=1, len=114(100), RPF lookup failed for source or RP
Rack1R5#
IP(0): s=155.1.146.1 (FastEthernet0/0) d=239.1.1.100 id=91, ttl=253,
prot=1, len=114(100), RPF lookup failed for source or RP
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

8.34 Catalyst IGMP Profiles

- Configure SW4 to permit only the IGMP reports for groups in ranges 232.0.0.0/8 and 239.0.0.0/8 from R4.

Configuration

Note

Catalyst switches allow filtering of IGMP messages sent by directly connected hosts to multicast routers. This feature is similar to the **ip igmp access-group** command used on routers, but this command applies to transit IGMP messages. This functionality is accomplished by using IGMP profiles. IGMP profiles have global numbers, and every profile defines a list of ranges plus the accompanying operation – "permit" or "deny" (default). In the first case, the switch permits IGMP reports for the groups specified by the range and denies everything else. In the second case, the switch denies the group range configured and permits everything else. To configure the profile, use the command

```
ip igmp profile <global-number>
 range low-address1 [high-address1]
 range low-address2 [high-address2]
 ...
 [permit|deny]
```

The profile applies ingress to layer 2 ports only using the interface-level command **ip igmp filter <number>** and affects all IGMP reports sent by hosts connected to the particular port.

SW4:

```
ip igmp profile 1
  permit
  range 232.0.0.0 232.255.255.255
  range 239.0.0.0 239.255.255.255
!
interface FastEthernet 0/4
 ip igmp filter 1
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Join R4's VLAN 146 interface to a couple of groups. One group should match the profile criteria, while the other should not. After this, verify what IGMP groups are seen on the switch, using the IGMP snooping function.

R4:

```
interface FastEthernet0/1
 ip igmp join-group 239.4.4.4
 ip igmp join-group 230.4.4.4
```

```
Rack1SW4#show ip igmp snooping groups vlan 146
Vlan      Group               Version     Port List
------------------------------------------------------------
146       224.0.1.40          v2          Fa0/13
146       239.4.4.4           v2          Fa0/4
```

Copyright © 2010 Internetwork Expert

IPv6 Solutions

Note

Load the *IPv6* initial configurations, which provide basic IPv4 connectivity across the topology with RIP as the IGP.

9.1 IPv6 Link-Local Addressing

- Configure link-local IPv6 addresses on the Frame-Relay cloud between R1, R2, R3, R4, and R5.
- Use the IPv6 addressing format FE80::Y, where Y is the router number.

Configuration

```
R1:
interface Serial 0/0
 ipv6 address fe80::1 link-local
 frame-relay map ipv6 fe80::5 105 broadcast
 frame-relay map ipv6 fe80::2 105
 frame-relay map ipv6 fe80::3 105
 frame-relay map ipv6 fe80::4 105

R2:
interface Serial 0/0
 ipv6 address fe80::2 link-local
 frame-relay map ipv6 fe80::5 205 broadcast
 frame-relay map ipv6 fe80::1 205
 frame-relay map ipv6 fe80::3 205
 frame-relay map ipv6 fe80::4 205

R3:
interface Serial 1/0
 ipv6 address fe80::3 link-local
 frame-relay map ipv6 fe80::5 305 broadcast
 frame-relay map ipv6 fe80::1 305
 frame-relay map ipv6 fe80::2 305
 frame-relay map ipv6 fe80::4 305

R4:
interface Serial 0/0/0
 ipv6 address fe80::4 link-local
 frame-relay map ipv6 fe80::5 405 broadcast
 frame-relay map ipv6 fe80::1 405
 frame-relay map ipv6 fe80::2 405
 frame-relay map ipv6 fe80::3 405
```

Copyright © 2010 Internetwork Expert

```
R5:
interface Serial 0/0/0
 ipv6 address fe80::5 link-local
 frame-relay map ipv6 fe80::1 501 broadcast
 frame-relay map ipv6 fe80::2 502 broadcast
 frame-relay map ipv6 fe80::3 503 broadcast
 frame-relay map ipv6 fe80::4 504 broadcast
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Link-local IPv6 addresses are significant only within the context of a single link. This means that packets with link-local addresses cannot be routed between interfaces, and link-local addresses may overlap as long as they exist on different interfaces. The address format for IPv6 link-local addresses are FE80::/10, and they are synonymous with the range 169.254.0.0/16 in IPv4.
>
> Packets with link-local sources or destinations are mostly used by the router's control plane protocols, such as IPv6 routing protocols. For broadcast segments, such as Ethernet, link-local reachability is implicit due to automatic resolution through ICMP Neighbor Discovery (ICMPND). However, since multipoint NBMA interfaces do not support Inverse ICMPND, it is necessary to ensure link-local IPv6 address reachability over these links by configuring static mappings. Without these mappings, routing protocols may not be able to establish adjacencies or properly encapsulate packets.

```
Rack1R5#show frame-relay map
Serial0/0/0 (up): ipv6 FE80::2 dlci 502(0x1F6,0x7C60), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ipv6 FE80::3 dlci 503(0x1F7,0x7C70), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ipv6 FE80::4 dlci 504(0x1F8,0x7C80), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ip 155.1.0.1 dlci 501(0x1F5,0x7C50), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ip 155.1.0.2 dlci 502(0x1F6,0x7C60), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ip 155.1.0.3 dlci 503(0x1F7,0x7C70), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ip 155.1.0.4 dlci 504(0x1F8,0x7C80), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ipv6 FE80::1 dlci 501(0x1F5,0x7C50), static,
              broadcast,
              CISCO, status defined, active
```

Copyright © 2010 Internetwork Expert

When pinging link-local addresses it is necessary to tell the router which interface to send traffic out. Since the link-local address can exist on multiple interfaces the router, requires you to indicate the "Output Interface". You must specify the interface using the full interface name without spaces (e.g. Serial0/0/0)

```
Rack1R5#ping ipv6 fe80::1
Output Interface: Serial0/0/0
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FE80::1, timeout is 2 seconds:
Packet sent with a source address of FE80::5
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/61/80 ms

Rack1R5#ping ipv6 fe80::2
Output Interface: Serial0/0/0
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FE80::2, timeout is 2 seconds:
Packet sent with a source address of FE80::5
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms

Rack1R5#ping ipv6 fe80::3
Output Interface: Serial0/0/0
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FE80::3, timeout is 2 seconds:
Packet sent with a source address of FE80::5
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 60/60/61 ms

Rack1R5#ping ipv6 fe80::4
Output Interface: Serial0/0/0
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FE80::4, timeout is 2 seconds:
Packet sent with a source address of FE80::5
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/60 ms
```

Copyright © 2010 Internetwork Expert

9.2 IPv6 Unique Local Addressing

- Enable IPv6 processing on the links between R6 & SW1 and R3 & SW1.
- Use the ULA IPv6 prefixes FC00:X:0:67::Y/64 and FC00:X:0:37::Y/64 for the links between R1 & R6 and R3 & SW1 respectively.
- Here X is your rack number in decimal notation, (e.g. 10 for Rack10, not the hex 0A) and Y is your router number (7 for SW1).

Configuration

Note

It is necessary to change the SDM (Switch Database Manager) template for the 3560 switches in order to configure IPv6 addressing. Changes to the running SDM preferences will be stored in memory. Reload the switch for them to take effect.

```
R3:
interface FastEthernet 0/0
 ipv6 address FC00:1:0:37::3/64

R6:
interface FastEthernet 0/0.67
 ipv6 address FC00:1:0:67::6/64

SW1:
sdm prefer dual-ipv4-and-ipv6 routing
!
interface Vlan 67
 ipv6 address FC00:1:0:67::7/64
!
interface FastEthernet 0/3
 ipv6 address FC00:1:0:37::7/64
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Unique Local IPv6 Unicast Addressing (ULA), defined in RFC 4193, deprecates the previously used Site-Local (FEC0::/10) addressing. ULA addresses in IPv6 are synonymous with the RFC 1918 private addresses found in IPv4, i.e. the 10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16 prefixes. Considered private, ULA addresses are not publicly routable prefixes on the Internet.
>
> The format of the ULA is:
>
> FC00 (7 bits) + Unique ID (41 bits) + Link ID (16 bits) + Interface ID (64 bits).
>
> The randomly generated Unique ID helps avoid address collisions. Other than the addressing format, ULA addressing exhibits no other unique behavior when compared to normal publicly routable IPv6 addresses.

```
Rack1SW1#ping fc00:1:0:37::3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FC00:1:0:37::3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/1/9 ms

Rack1SW1#ping fc00:1:0:67::6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FC00:1:0:67::6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/1/9 ms
```

9.3 IPv6 Global Aggregatable Addressing

- Configure globally aggregatable IPv6 addresses in the format 2001:X:0:1234::Y/64 on the Frame-Relay interfaces of R1, R2, R3, R4, and R5.

  Here X is your rack number in decimal notation, (e.g. 10 for Rack 10, not the hex 0A) and Y is your router number (7 for SW1).

Configuration

```
R1:
interface Serial 0/0
 ipv6 address 2001:1:0:1234::1/64
 frame-relay map ipv6 2001:1:0:1234::5 105
 frame-relay map ipv6 2001:1:0:1234::2 105
 frame-relay map ipv6 2001:1:0:1234::3 105
 frame-relay map ipv6 2001:1:0:1234::4 105

R2:
interface Serial 0/0
 ipv6 address 2001:1:0:1234::2/64
 frame-relay map ipv6 2001:1:0:1234::5 205
 frame-relay map ipv6 2001:1:0:1234::1 205
 frame-relay map ipv6 2001:1:0:1234::3 205
 frame-relay map ipv6 2001:1:0:1234::4 205

R3:
interface Serial 1/0
 ipv6 address 2001:1:0:1234::3/64
 frame-relay map ipv6 2001:1:0:1234::5 305
 frame-relay map ipv6 2001:1:0:1234::1 305
 frame-relay map ipv6 2001:1:0:1234::2 305
 frame-relay map ipv6 2001:1:0:1234::4 305

R4:
interface Serial 0/0/0
 ipv6 address 2001:1:0:1234::4/64
 frame-relay map ipv6 2001:1:0:1234::5 405
 frame-relay map ipv6 2001:1:0:1234::1 405
 frame-relay map ipv6 2001:1:0:1234::2 405
 frame-relay map ipv6 2001:1:0:1234::3 405

R5:
interface Serial 0/0/0
 ipv6 address 2001:1:0:1234::5/64
 frame-relay map ipv6 2001:1:0:1234::1 501
 frame-relay map ipv6 2001:1:0:1234::2 502
 frame-relay map ipv6 2001:1:0:1234::3 503
 frame-relay map ipv6 2001:1:0:1234::4 504
```

Copyright © 2010 Internetwork Expert

Verification

Note

Globally Aggregatable IPv6 addresses, AKA global unicast addresses, are publicly allocated and routable on the Internet. Global unicast addresses start with the binary prefix 001 (2000::/3), and therefore encompass the range 2000:: - 3FFF::. By design the allocation of these addresses through Internet registries such as ARIN and APNIC are hierarchical, unlike IPv4 allocation. RIPE-450 (*IPv6 Address Allocation and Assignment Policy)* defines this hierarchical design methodology.

Although this range is extremely large – 1/8th of the total IPv6 address space – generally only the prefix 2001::/16 is currently used for allocation. Other ranges, such as the 6to4 Tunnel range of 2002::/16, are generally used for special purposes.

Once assigned, the behavior of a global unicast address is identical to that of a ULA address, but as its name implies it has global routing significance.

```
Rack1R5#ping 2001:1:0:1234::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:1:0:1234::1, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/61 ms

Rack1R5#ping 2001:1:0:1234::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:1:0:1234::2, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms

Rack1R5#ping 2001:1:0:1234::3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:1:0:1234::3, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#ping 2001:1:0:1234::4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:1:0:1234::4, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms

Rack1R5#show frame-relay map
Serial0/0/0 (up): ipv6 FE80::2 dlci 502(0x1F6,0x7C60), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ipv6 FE80::3 dlci 503(0x1F7,0x7C70), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ipv6 FE80::4 dlci 504(0x1F8,0x7C80), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ip 155.1.0.1 dlci 501(0x1F5,0x7C50), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ip 155.1.0.2 dlci 502(0x1F6,0x7C60), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ip 155.1.0.3 dlci 503(0x1F7,0x7C70), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ip 155.1.0.4 dlci 504(0x1F8,0x7C80), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ipv6 2001:1:0:1234::1 dlci 501(0x1F5,0x7C50), static,
              CISCO, status defined, active
Serial0/0/0 (up): ipv6 2001:1:0:1234::2 dlci 502(0x1F6,0x7C60), static,
              CISCO, status defined, active
Serial0/0/0 (up): ipv6 2001:1:0:1234::3 dlci 503(0x1F7,0x7C70), static,
              CISCO, status defined, active
Serial0/0/0 (up): ipv6 2001:1:0:1234::4 dlci 504(0x1F8,0x7C80), static,
              CISCO, status defined, active
Serial0/0/0 (up): ipv6 FE80::1 dlci 501(0x1F5,0x7C50), static,
              broadcast,
              CISCO, status defined, active
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

9.4 IPv6 EUI-64 Addressing

- Configure globally aggregatable IPv6 addresses in the format 2001:X:0:146::Y/64 on the VLAN 146 interfaces of R1, R4, and R6.
- Here X is your rack number in decimal notation, (e.g. 10 for Rack10, not the hex 0A) and the router will automatically derive Y based on the interface MAC address.

Configuration

```
R1:
interface FastEthernet 0/0
 ipv6 address 2001:1:0:146::/64 eui-64

R4:
interface FastEthernet 0/1
 ipv6 address 2001:1:0:146::/64 eui-64

R6:
interface FastEthernet 0/0.146
 ipv6 address 2001:1:0:146::/64 eui-64
```

Verification

Note

Extended Unique Identifiers (EUIs) are 64-bit values assigned to physical interfaces. EUI's are similar in many respects to IEEE MAC addresses, in that they identity a physical interface, but EUI's are designed to be used universally, not just with the hardware addresses.

IPv6 uses EUI-64 to construct a unique host address on a shared Ethernet segment automatically. When you use the **`eui-64`** keyword, the IOS uses the 48-bit hardware address of the Ethernet interface as the foundation to construct the unique 64-bit Interface identifier of the IPv6 address.

Specifically this is accomplished by inverting the 7th most significant bit of the Ethernet MAC address, called the Universal/Local (U/L) bit, and then inserting 16 bits of padding in the format FFFE in the middle.

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ipv6 interface fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::20D:65FF:FE84:6560
  Global unicast address(es):
    2001:1:0:146:20D:65FF:FE84:6560, subnet is 2001:1:0:146::/64 [EUI]
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF84:6560
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
```

From the above output we can see that the EUI-64 derived host address of R1's Fa0/0 interface is 20D:65FF:FE84:6560. This means that the MAC address of R1's Fa0/0 interface is 000D:6584:6560. Note that the link-local address of the interface automatically uses the format FE80::[EUI-64] unless manually modified.

Copyright © 2010 Internetwork Expert

9.5 IPv6 Auto-Configuration

- Use the unique local address FC00:X:0:58::5/64 for R5's VLAN 58 interface, where X is your rack number in decimal notation.
- Configure an additional IPv6 address of FC00:X:0:85::5/64 on the same interface of R5.
- Configure R5 to broadcast both prefixes using ICMPv6 ND RAs, but only allow the hosts to use the second prefix for auto-configuration.
- The valid and preferred lifetime for both prefixes should be set to 4 hours.
- R5 should advertise itself as the default router on the segment every 40 seconds with a lifetime interval of 60 seconds.
- SW2 should learn its IPv6 address on VLAN 58 automatically and use R5 as its default gateway.

Configuration

```
R5:
ipv6 unicast-routing
!
interface FastEthernet 0/0
 ipv6 address fc00:1:0:58::5/64
 ipv6 address fc00:1:0:85::5/64
 ipv6 nd prefix fc00:1:0:58::/64 14400 14400 no-autoconfig
 ipv6 nd prefix fc00:1:0:85::/64 14400 14400
 ipv6 nd ra-interval 40
 ipv6 nd ra-lifetime 60
 no ipv6 nd suppress-ra

SW2:
sdm prefer dual-ipv4-and-ipv6 routing
!
interface Vlan 58
 ipv6 address autoconfig default
```

Copyright © 2010 Internetwork Expert

Verification

Note

IPv6 has a special feature called auto-configuration. It replaces many functions served by DHCP in IPv4 networks (yet there is a special DHCPv6 edition of the DHCP protocol). With IPv6 auto-configuration, an IPv6 host may automatically learn the IPv6 prefixes assigned to the local segment, as well as determine the default routers on that segment. A special type of link-local IPv6 addressing and the ICMPv6 ND (Neighbor Discovery) protocol accomplish this.

Router Advertisements can be sent at a specified interval or the feature can be suppressed completely ("suppress-ra" is enabled by default on Ethernet interfaces. This default behavior can be changed with the no ipv6 nd suppress-ra) depending on how you configure your Cisco router. Note that you must enable IPv6 unicast routing in order to send router advertisements. The following commands control the IPv6 RA announcements:

`ipv6 nd ra-interval` – specifies the periodic interval to send RAs.
`ipv6 nd ra-lifetime` – specifies the validity interval of the router's IPv6 address
`ipv6 nd prefix` – manipulates the IPv6 network prefixes included into RA. By default, all prefixes are included.

You may adjust the interval that the prefix is valid and preferred with every command. Additionally, you may instruct the hosts not to use this prefix for auto-configuration, by using the **`no-autoconfig`** keyword.

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show ipv6 interface vlan 58
Vlan58 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::21B:D4FF:FED4:4D45
  Global unicast address(es):
    FC00:1:0:85:21B:D4FF:FED4:4D45, subnet is FC00:1:0:85::/64 [PRE]
      valid lifetime 14374 preferred lifetime 14374
  Joined group address(es):
    FF02::1
    FF02::1:FFD4:4D45
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
  Default router is FE80::212:FF:FE1C:DAA0 on Vlan58

Rack1R5#debug ipv6 nd
ICMP Neighbor Discovery events debugging is on
Rack1R5#
ICMPv6-ND: Sending RA to FF02::1 on FastEthernet0/0
ICMPv6-ND:     MTU = 1500
ICMPv6-ND:     prefix = FC00:1:0:58::/64 onlink
ICMPv6-ND:           14400/14400 (valid/preferred)
ICMPv6-ND:     prefix = FC00:1:0:85::/64 onlink autoconfig
ICMPv6-ND:           14400/14400 (valid/preferred)

Rack1R5#show ipv6 interface fastEthernet 0/0 prefix
IPv6 Prefix Advertisements FastEthernet0/0
Codes: A - Address, P - Prefix-Advertisement, O - Pool
       U - Per-user prefix, D - Default
       N - Not advertised, C - Calendar

PD   default [LA] Valid lifetime 2592000, preferred lifetime 604800
AP   FC00:1:0:58::/64 [L] Valid lifetime 14400, preferred lifetime
14400
AP   FC00:1:0:85::/64 [LA] Valid lifetime 14400, preferred lifetime
14400
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

9.6 RIPng

- Enable RIPng on the VLAN 146 segment connecting R1, R4, and R6 using the process name “RIPNG”.
- Create new Loopback 100 interfaces on R1, R4, and R6 with the IPv6 addresses FC00:X:0:Y::Y/64, where X is your rack number in decimal notation and Y is your router number.
- Advertise these Loopback interfaces into RIPng.

Configuration

```
R1:
ipv6 unicast-routing
!
interface FastEthernet 0/0
 ipv6 rip RIPNG enable
!
interface Loopback100
 ipv6 address fc00:1:0:1::1/64
 ipv6 rip RIPNG enable

R4:
ipv6 unicast-routing
!
interface FastEthernet 0/1
 ipv6 rip RIPNG enable
!
interface Loopback100
 ipv6 address fc00:1:0:4::4/64
 ipv6 rip RIPNG enable

R6:
ipv6 unicast-routing
!
interface FastEthernet 0/0.146
 ipv6 rip RIPNG enable
!
interface Loopback100
 ipv6 address fc00:1:0:6::6/64
 ipv6 rip RIPNG enable
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> To configure RIPng, simply enable it on the respective interfaces. Global RIPng parameters are configured in the routing process sub-configuration mode by issuing the **`ipv6 rip [pid]`** command in global configuration.
>
> With the exception of the ip keyword being replaced by ipv6 many of the routing verification commands are similar between IPv4 and IPv6.

```
Rack1R6#show ipv6 protocols
IPv6 Routing Protocol is "connected"
IPv6 Routing Protocol is "static"
IPv6 Routing Protocol is "rip RIPNG"
  Interfaces:
    Loopback100
    FastEthernet0/0.146
  Redistribution:
    None
```

```
Rack1R6#show ipv6 rip RIPNG
RIP process "RIPNG", port 521, multicast-group FF02::9, pid 207
     Administrative distance is 120. Maximum paths is 16
     Updates every 30 seconds, expire after 180
     Holddown lasts 0 seconds, garbage collect after 120
     Split horizon is on; poison reverse is off
     Default routes are not generated
     Periodic updates 5, trigger updates 1
  Interfaces:
    Loopback100
    FastEthernet0/0.146
  Redistribution:
    None
```

```
Rack1R6#show ipv6 route rip
IPv6 Routing Table - 10 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   FC00:1:0:1::/64 [120/2]
     via FE80::20D:65FF:FE84:6560, FastEthernet0/0.146
R   FC00:1:0:4::/64 [120/2]
     via FE80::20F:24FF:FE11:4961, FastEthernet0/0.146
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#debug ipv6 rip
RIP Routing Protocol debugging is on
Rack1R6#
RIPng: response received from FE80::20D:65FF:FE84:6560 on
FastEthernet0/0.146 for RIPNG
       src=FE80::20D:65FF:FE84:6560 (FastEthernet0/0.146)
       dst=FF02::9
       sport=521, dport=521, length=52
       command=2, version=1, mbz=0, #rte=2
       tag=0, metric=1, prefix=2001:1:0:146::/64
       tag=0, metric=1, prefix=FC00:1:0:1::/64
Rack1R6#
RIPng: response received from FE80::20F:24FF:FE11:4961 on
FastEthernet0/0.146 for RIPNG
       src=FE80::20F:24FF:FE11:4961 (FastEthernet0/0.146)
       dst=FF02::9
       sport=521, dport=521, length=52
       command=2, version=1, mbz=0, #rte=2
       tag=0, metric=1, prefix=2001:1:0:146::/64
       tag=0, metric=1, prefix=FC00:1:0:4::/64
Rack1R6#
RIPng: Sending multicast update on Loopback100 for RIPNG
       src=FE80::20C:85FF:FEC1:FC60
       dst=FF02::9 (Loopback100)
       sport=521, dport=521, length=92
       command=2, version=1, mbz=0, #rte=4
       tag=0, metric=1, prefix=2001:1:0:146::/64
       tag=0, metric=1, prefix=FC00:1:0:6::/64
       tag=0, metric=2, prefix=FC00:1:0:4::/64
       tag=0, metric=2, prefix=FC00:1:0:1::/64
RIPng: Sending multicast update on FastEthernet0/0.146 for RIPNG
       src=FE80::20C:85FF:FEC1:FC60
       dst=FF02::9 (FastEthernet0/0.146)
       sport=521, dport=521, length=52
       command=2, version=1, mbz=0, #rte=2
       tag=0, metric=1, prefix=2001:1:0:146::/64
Rack1R6#
       tag=0, metric=1, prefix=FC00:1:0:6::/64
RIPng: Process RIPNG received own response on Loopback100
```

Copyright © 2010 Internetwork Expert

9.7 RIPng over NBMA

- Enable RIPng on the Frame-Relay interfaces of R1, R4, and R5 using the process name “RIPNG”.
- Ensure that R1 and R4 can reach each other’s Loopback 100 subnets even if any of their VLAN 146 interfaces are down.

Configuration

```
R1:
interface Serial 0/0
 ipv6 rip RIPNG enable

R4:
interface Serial0/0/0
 ipv6 rip RIPNG enable

R5:
ipv6 unicast-routing
!
ipv6 router rip RIPNG
 no split-horizon
!
interface Serial 0/0/0
 ipv6 rip RIPNG enable
```

Verification

Note

NBMA networks pose two potential issues to IPv6 routing. First, unmapped link-local IPv6 addresses can cause incorrect processing of routing updates. These updates will not be processed correctly because in NBMA networks the next-hop is always going to be a link-local address. Do not forget to use the **broadcast** keyword with the IPv6 mapping statements on DLCIs that should switch routing updates, otherwise the router will not send any multicast packets.

Secondly, partial mesh connectivity might prevent some nodes from receiving routing updates. Commonly, this problem arises in Hub-and-Spoke topologies, where spokes do not receive routing information from other spokes. In order to deal with this issue, disable the split-horizon function in RIPng. You can only disable split-horizon globally, under the routing process, not on a per interface basis.

Alternatively, you may want to use either default routing or summarization at the hub router to prevent the loss of reachability from taking place. In our case, we will simply disable split-horizon on R5.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ipv6 route rip
IPv6 Routing Table - 12 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   2001:1:0:146::/64 [120/2]
     via FE80::4, Serial0/0/0
     via FE80::1, Serial0/0/0
R   FC00:1:0:1::/64 [120/2]
     via FE80::1, Serial0/0/0
R   FC00:1:0:4::/64 [120/2]
     via FE80::4, Serial0/0/0
R   FC00:1:0:6::/64 [120/3]
     via FE80::4, Serial0/0/0
     via FE80::1, Serial0/0/0

Rack1R5#debug ipv6 rip
RIPng: Sending multicast update on Serial0/0/0 for RIPNG
       src=FE80::5
       dst=FF02::9 (Serial0/0/0)
       sport=521, dport=521, length=112
       command=2, version=1, mbz=0, #rte=5
       tag=0, metric=1, prefix=2001:1:0:1234::/64
       tag=0, metric=2, prefix=2001:1:0:146::/64
       tag=0, metric=2, prefix=FC00:1:0:4::/64
       tag=0, metric=2, prefix=FC00:1:0:1::/64
       tag=0, metric=3, prefix=FC00:1:0:6::/64
Rack1R5#
```

Copyright © 2010 Internetwork Expert

9.8 RIPng Summarization

- Create a new Loopback 100 interface on R5 with the IPv6 address FC00:X:0:5::5/64 where X is your rack number in decimal notation and advertise this interface into RIPng.
- Ensure that R1 and R4 advertise only the summary prefix for the Loopback 100 interfaces of R1, R4, and R5 to R6 across the VLAN 146 segment.

Configuration

```
R1:
interface FastEthernet 0/0
 ipv6 rip RIPNG summary fc00:1::/61

R4:
interface FastEthernet 0/1
 ipv6 rip RIPNG summary fc00:1::/61

R5:
interface Loopback 100
 ipv6 address fc00:1:0:5::5/64
 ipv6 rip RIPNG enable
```

Verification

Note

In order to summarize RIPng routes, use the interface level command **`ipv6 rip [pid] summary`**. Use the standard rules of summarization, converting IPv6 prefixes from hex format into binary format. In our case we have three prefixes:

FC00:X:0:5::/64
FC00:X:0:1::/64
FC00:X:0:4::/64

The prefixes differ in the 4th position (remember each block of the IPv6 address is 16 bits):

0000 0000 0000 0**101**
0000 0000 0000 0**001**
0000 0000 0000 0**100**

Copyright © 2010 Internetwork Expert

In order to summarize, move the prefix length 3 bit positions to the left. The resulting prefix is:

FC00:X::/61

```
Rack1R6#show ipv6 route rip
IPv6 Routing Table - 10 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   2001:1:0:1234::/64 [120/2]
     via FE80::20D:65FF:FE84:6560, FastEthernet0/0.146
     via FE80::20F:24FF:FE11:4961, FastEthernet0/0.146
R   FC00:1::/61 [120/2]
     via FE80::20F:24FF:FE11:4961, FastEthernet0/0.146
     via FE80::20D:65FF:FE84:6560, FastEthernet0/0.146

Rack1R4#show ipv6 route rip
IPv6 Routing Table - 9 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   FC00:1::/61 [120/2]
     via FE80::211:92FF:FE0E:80C0, FastEthernet0/1
R   FC00:1:0:1::/64 [120/3]
     via FE80::5, Serial0/0/0
R   FC00:1:0:5::/64 [120/2]
     via FE80::5, Serial0/0/0
R   FC00:1:0:6::/64 [120/2]
     via FE80::21A:6CFF:FE30:87A2, FastEthernet0/1

Rack1R1#show ipv6 route rip
IPv6 Routing Table - 11 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   FC00:1::/61 [120/2]
     via FE80::21A:2FFF:FEFE:419, FastEthernet0/0
R   FC00:1:0:4::/64 [120/3]
     via FE80::5, Serial0/0
R   FC00:1:0:5::/64 [120/2]
     via FE80::5, Serial0/0
R   FC00:1:0:6::/64 [120/2]
     via FE80::21A:6CFF:FE30:87A2, FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

9.9 RIPng Prefix Filtering

- Configure R5 to block the IPv6 prefix corresponding to R6's Loopback 100 interface from entering the local routing table.
- Permit any other IPv6 prefixes

Configuration

```
R5:
ipv6 prefix-list FILTER_R6_LO100 deny fc00:1:0:6::/64
ipv6 prefix-list FILTER_R6_LO100 permit ::/0 le 128
!
ipv6 router rip RIPNG
 distribute-list prefix-list FILTER_R6_LO100 in
```

Verification

Note

RIPng uses IPv6 prefix-lists to filter routing updates. You apply prefix-lists either inbound or outbound under the RIPng process configuration mode. You may choose to associate an interface with the distribute-list, or apply it to all interfaces simultaneously by not specifying an interface.

Copyright © 2010 Internetwork Expert

Before applying the filter:

```
Rack1R5#show ipv6 route rip
IPv6 Routing Table - 13 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   2001:1:0:146::/64 [120/2]
     via FE80::4, Serial0/0
     via FE80::1, Serial0/0
R   FC00:1::/61 [120/3]
     via FE80::4, Serial0/0
     via FE80::1, Serial0/0
R   FC00:1:0:1::/64 [120/2]
     via FE80::1, Serial0/0
R   FC00:1:0:4::/64 [120/2]
     via FE80::4, Serial0/0
R   FC00:1:0:6::/64 [120/3]
     via FE80::4, Serial0/0
     via FE80::1, Serial0/0
```

Once the filter is applied, it may take some time for the routes to flush out of the routing table. This process can be sped up by clearing the rip process with the **`clear ipv6 rip [pid]`** command. The route will no longer appear once the table ages out the old prefix.

```
Rack1R5#sh ipv6 route rip
IPv6 Routing Table - Default - 13 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   2001:1:0:146::/64 [120/2]
     via FE80::4, Serial0/0/0
     via FE80::1, Serial0/0/0
R   FC00:1::/61 [120/3]
     via FE80::4, Serial0/0/0
     via FE80::1, Serial0/0/0
R   FC00:1:0:1::/64 [120/2]
     via FE80::1, Serial0/0/0
R   FC00:1:0:4::/64 [120/2]
     via FE80::4, Serial0/0/0
```

Copyright © 2010 Internetwork Expert

9.10 RIPng Metric Manipulation

- Enable IPv6 and RIPng on the Serial link connecting R4 and R5.
- Ensure that R4 prefers the Serial link to reach the Loopback 100 interface of R5 and uses the Frame-Relay link ONLY as a backup.
- Do not use summarization to accomplish this task.

Configuration

```
R4:
interface Serial 0/1/0
 ipv6 enable
ipv6 rip RIPNG enable
!
interface Serial 0/0/0
 ipv6 rip RIPNG metric-offset 2

R5:
interface Serial 0/1/0
 ipv6 enable
 ipv6 rip RIPNG enable
```

Verification

Note

RIPng does not have granular metric manipulation mechanisms. As an example, we can only apply metric offsets to all routes received via a particular interface, thus allowing us to define the outgoing interface preference. Note that by specifying a metric offset that results in a total metric of 16 or higher we can also filter routes using this feature..

This task also demonstrates that you only need link-local IPv6 addresses on a link to activate IPv6 routing. By issuing the **`ipv6 enable`** command you configure the link for automatic link-local IPv6 address allocation.

Before the metric-offset:

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ipv6 route fc00:1:0:5::/64
Routing entry for FC00:1:0:5::/64
  Known via "rip RIPNG", distance 120, metric 2
  Route count is 2/2, share count 0
  Routing paths:
    FE80::21A:6CFF:FED5:4A4, Serial0/1/0
      Last updated 00:17:47 ago
    FE80::5, Serial0/0/0
      Last updated 00:14:58 ago
```

After the metric offset is applied:

```
Rack1R4#show ipv6 route fc00:1:0:5::/64
Routing entry for FC00:1:0:5::/64
  Known via "rip RIPNG", distance 120, metric 2
  Route count is 1/1, share count 0
  Routing paths:
    FE80::21A:6CFF:FED5:4A4, Serial0/1/0
      Last updated 00:20:38 ago
```

Copyright © 2010 Internetwork Expert

9.11 RIPng Default Routing

- Configure R6 to advertise a default route into the RIPng routing domain with an initial metric of 5.

Configuration

```
R6:
interface FastEthernet 0/0.146
 ipv6 rip RIPNG default-information originate metric 5
```

Verification

> **Note**
>
> RIPng allows the sending of a default route out of any router interface. Additionally, you may specify the default route's metric. Adding the keyword **only** you may further configure the router to send just the default route exclusively out the particular interface, and filter all other subnet advertisements.

```
Rack1R4#show ipv6 route rip
IPv6 Routing Table - 10 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   ::/0 [120/6]
     via FE80::21A:6CFF:FE30:87A2, FastEthernet0/1
R   FC00:1::/61 [120/2]
     via FE80::211:92FF:FE0E:80C0, FastEthernet0/1
R   FC00:1:0:1::/64 [120/3]
     via FE80::219:30FF:FEE7:CBB6, Serial0/1/0
R   FC00:1:0:5::/64 [120/2]
     via FE80::219:30FF:FEE7:CBB6, Serial0/1/0
R   FC00:1:0:6::/64 [120/2]
     via FE80::21A:6CFF:FE30:87A2, FastEthernet0/1
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

9.12 EIGRPv6

- Enable the EIGRPv6 routing process on R4, R5, R6 and SW2.
- Enable EIGRPv6 on all interfaces connecting R4, R5, R6 and SW2.
- Create new Loopback 100 interfaces on the mentioned routers with the IPv6 addresses FC00:X:0:Y::Y/64, where X is your rack number in decimal notation and Y is the router number.
- Advertise these Loopback interfaces into EIGRPv6.
- Enable EIGRPv6 on the Frame-Relay interfaces of R4, and R5.
- Authenticate the EIGRPv6 adjacency over VLAN 146 using the password value of CISCO.

Configuration

Note

EIGRPv6 is an IPv6 routing protocol that was added to Cisco IOS starting with version 12.4T. It mimics the functionality and convergence algorithm of the classic IPv4 EIGRP protocol and many configuration commands are similar to their IPv4 counterparts. Even the administrative distances for IPv6 EIGRP internal and external routes are the same. In order to enable the EIGRPv6 routing process use the following global commands:

```
ipv6 router eigrp N
 no shutdown
```

The IPv6 version of EIGRP uses AS numbers to identify routing processes. The second command is necessary to bring the process up – this feature is not found in other IPv6 routing protocols. After enabling the process you may select the interfaces that you want to advertise into EIGRPv6 by using the command **`ipv6 eigrp N`** where N is the AS number you configured on the routing process. Just like with IPv4 EIGRP you may use the interface commands **`ipv6 hold-time`** and **`ipv6 hello-interval`** to change the EIGRPv6 interface timers.

Similar to IPv4 EIGRP, enabling authentication on the interface is done in three stages: first create a key-chain, second set the interface authentication mode and third apply the key-chain:

Copyright © 2010 Internetwork Expert

```
key chain EIGRPV6
 key 1
  key-string CISCO
!
interface FastEthernet 0/1
 ipv6 eigrp 100
 ipv6 authentication mode eigrp 100 md5
 ipv6 authentication key-chain eigrp 100 EIGRPV6
```

Notice that EIGRPv6 is still a distance-vector protocol, this means the classic split horizon rule still applies. When deployed in hub-and-spoke topologies e.g. on a Frame-Relay interface, you may want to use the command **no ipv6 split-horizon eigrp 100** to disable the split-horizon rule on a particular interface. This is unlike RIPng that only allows disabling split-horizon globally. It is interesting to note that EIGRPv6 supports the interface-level command "**no ipv6 next-hop-self eigrp**" as well. This command, used on the hub router, explicitly sets the next-hop field in the relayed EIGRPv6 updates to the spoke router's IP address. DMVPN deployments commonly use this feature.

R4:

```
ipv6 unicast-routing
ipv6 router eigrp 100
 no shutdown
!
key chain EIGRPV6
 key 1
  key-string CISCO
!
interface FastEthernet 0/1
 ipv6 eigrp 100
 ipv6 authentication mode eigrp 100 md5
 ipv6 authentication key-chain eigrp 100 EIGRPV6
!
interface Loopback100
 ipv6 address fc00:1:0:4::4/64
 ipv6 eigrp 100
!
interface Serial0/0/0
 ipv6 eigrp 100
!
interface Serial0/1/0
 ipv6 eigrp 100
```

Copyright © 2010 Internetwork Expert

R5:

```
ipv6 unicast-routing
ipv6 router eigrp 100
 no shutdown
!
interface Serial 0/0/0
 ipv6 eigrp 100
!
interface Serial0/1/0
 ipv6 eigrp 100

!
interface FastEthernet 0/0
 ipv6 eigrp 100
!
interface Loopback100
 ipv6 address fc00:1:0:5::5/64
 ipv6 eigrp 100
```

R6:

```
ipv6 unicast-routing
ipv6 router eigrp 100
 no shutdown
!
key chain EIGRPV6
 key 1
  key-string CISCO
!
interface FastEthernet 0/0.146
 ipv6 eigrp 100
 ipv6 authentication mode eigrp 100 md5
 ipv6 authentication key-chain eigrp 100 EIGRPV6
!
interface Loopback100
 ipv6 address fc00:1:0:6::6/64
 ipv6 eigrp 100
```

SW2:

```
sdm prefer dual-ipv4-and-ipv6 routing
ipv6 unicast-routing
!
ipv6 unicast-routing
ipv6 router eigrp 100
 no shutdown
!
!
interface Vlan 58
 ipv6 eigrp 100
!
interface Loopback 100
 ipv6 address fc00:1:0:8::8/64
 ipv6 eigrp 100
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Start by checking the IPv6 EIGRP process settings and protocol adjacencies:

```
Rack1R5#show ipv6 eigrp 100 interfaces
IPv6-EIGRP interfaces for process 100

                     Xmit Queue   Mean   Pacing Time   Multicast    Pending
Interface        Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer   Routes
Se0/0/0            1        0/0       102       0/15          475           0
Fa0/0              1        0/0        19       0/1           128           0
Lo100              0        0/0         0       0/1             0           0
Se0/1/0            1        0/0        26       0/15          131           0

Rack1R5#show ipv6 protocols
IPv6 Routing Protocol is "connected"
IPv6 Routing Protocol is "rip RIPNG"
  Interfaces:
    Serial0/1/0
    Loopback100
    Serial0/0/0
  Redistribution:
    None
IPv6 Routing Protocol is "eigrp 100"
  EIGRP metric weight K1=1, K2=0, K3=1, K4=0, K5=0
  EIGRP maximum hopcount 100
  EIGRP maximum metric variance 1
  Interfaces:
    FastEthernet0/0
    Serial0/0/0
    Serial0/1/0
    Loopback100
  Redistribution:
    None
  Maximum path: 16
  Distance: internal 90 external 170

Rack1R5#show ipv6 eigrp neighbors
IPv6-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q
Seq
                                            (sec)         (ms)
Cnt Num
2   Link-local address:     Fa0/0             10 00:25:20   19   200  0
9
    FE80::218:73FF:FE8D:9E45
1   Link-local address:     Se0/1/0           12 00:27:14   26   200  0
13
    FE80::215:62FF:FE41:FD9C
0   Link-local address:     Se0/0/0          142 00:28:03  102   612  0
14
    FE80::4
```

Note

You should repeat this procedure on all routers running IPv6 EIGRP. Now check the authenticated adjacency between R4 and R6:

```
Rack1R4#show ipv6 eigrp neighbors fastEthernet 0/1
IPv6-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q
Seq
                                            (sec)         (ms)
Cnt Num
1   Link-local address:     Fa0/1             14 00:30:51  816  4896  0
4
    FE80::21A:2FFF:FE78:4678

Rack1R4#show ipv6 eigrp interfaces detail fastEthernet 0/1
IPv6-EIGRP interfaces for process 100

                        Xmit Queue   Mean   Pacing Time   Multicast
Pending
Interface        Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer
Routes
Fa0/1              1        0/0       816       0/1          4080
0
  Hello interval is 5 sec
  Next xmit serial <none>
  Un/reliable mcasts: 0/3  Un/reliable ucasts: 4/3
  Mcast exceptions: 1  CR packets: 1  ACKs suppressed: 0
  Retransmissions sent: 0  Out-of-sequence rcvd: 1
  Authentication mode is md5,  key-chain is "EIGRPV6"
  Use multicast
Rack1R4#
```

Copyright © 2010 Internetwork Expert

Now check the routing information propagation and end-to-end connectivity across the EIGRPv6 domain:

```
Rack1SW2#show ipv6 route eigrp
IPv6 Routing Table - Default - 12 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       R - RIP, D - EIGRP, EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
D   2001:1:0:146::/64 [90/2172928]
     via FE80::213:1AFF:FE68:B04C, Vlan58
D   2001:1:0:1234::/64 [90/2170368]
     via FE80::213:1AFF:FE68:B04C, Vlan58
D   FC00:1:0:4::/64 [90/2298368]
     via FE80::213:1AFF:FE68:B04C, Vlan58
D   FC00:1:0:5::/64 [90/131072]
     via FE80::213:1AFF:FE68:B04C, Vlan58
D   FC00:1:0:6::/64 [90/2300928]
     via FE80::213:1AFF:FE68:B04C, Vlan58
D   FC00:1:0:58::/64 [90/28672]
     via FE80::213:1AFF:FE68:B04C, Vlan58

Rack1SW2#ping fc00:1:0:6::6 source loopback 100

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FC00:1:0:6::6, timeout is 2 seconds:
Packet sent with a source address of FC00:1:0:8::8
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 33/36/42 ms
```

Copyright © 2010 Internetwork Expert

9.13 EIGRPv6 Summarization

- Ensure that R5 advertises an optimal summary prefix for the IPv6 addresses of R5's and SW2's Loopback 100 interfaces.

Configuration

Note

Thanks to EIGRPv6's distance-vector behavior, you may summarize routing information at any point in the network. More specifically, you may configure the routing process to summarize prefixes advertised out any interface using the command **`ipv6 summary-address eigrp N`**. EIGRPv6 follows the same summarization rules we used in the previous RIPng scenarios. In our case, we have two prefixes:

FC00:1:0:5::/64 and FC00:1:0:8::/64

Breaking down 5 and 8 into binary digit notation (remember every component is 16 bits):

0000 0000 0000 **00101** = 5
0000 0000 0000 **01000** = 8

We can see that in order to remove the "dissimilar" parts we need to shift the prefix length four bits to the left. Thus the summary prefix is:

FC00:1::/60

Remember to configure summarization on both interfaces connecting R5 to R4 so that no specific routing information leaks to R4. Notice that unlike IPv4 EIGRP, you cannot associate a leak-map with the summary prefix and allow specific routing information to leak out.

```
R5:
interface Serial 0/0/0
 ipv6 summary-address eigrp 100 FC00:1::/60
!
interface Serial 0/1/0
 ipv6 summary-address eigrp 100 FC00:1::/60
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check R4's routing table for EIGRP routes and notice that individual prefixes for R5's and SW2's Loopback 100 interfaces are not present.

```
Rack1R4#show ipv6 route eigrp
IPv6 Routing Table - Default - 15 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
D   FC00:1::/60 [90/2297856]
     via FE80::5, Serial0/0/0
     via FE80::213:1AFF:FE68:B04C, Serial0/1/0
D   FC00:1:0:6::/64 [90/156160]
     via FE80::21A:2FFF:FE78:4678, FastEthernet0/1
D   FC00:1:0:58::/64 [90/2172416]
     via FE80::5, Serial0/0/0
     via FE80::213:1AFF:FE68:B04C, Serial0/1/0
D   FC00:1:0:85::/64 [90/2172416]
     via FE80::5, Serial0/0/0
     via FE80::213:1AFF:FE68:B04C, Serial0/1/0
```

Copyright © 2010 Internetwork Expert

9.14 EIGRPv6 Prefix Filtering

- Configure R5 to block the IPv6 prefix corresponding to R6's Loopback 100 interface from entering the local routing table.
- Permit any other IPv6 prefixes.

Configuration

> **Note**
>
> As with any other distance-vector protocol, you may filter EIGRPv6 routing information at any point in the network. Use the routing process-command **`distribute-list prefix-list <NAME> in|out [Interface]`** to filter incoming or outgoing updates respectively. You may apply the list to an interface or configure it to filter in/out all directions, by omitting the interface name.
>
> Notice that due to its distance-vector nature, a filtered prefix will not appear on any router that is learning the route from the filtering node solely.

```
R5:
no ipv6 prefix-list R6_LOOP100
!
! Notice the last line in the prefix-list which permits all
! IPv6 prefixes
!
ipv6 prefix-list R6_LOOP100 seq 10 deny FC00:1:0:6::/64
Ipv6 prefix-list R6_LOOP100 seq 20 permit ::/0 le 128
!
ipv6 router eigrp 100
 distribute-list prefix-list R6_LOOP100 in
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Verify that EIGRPv6 no longer learns about R6's Loopback 100, and ensure that other prefixes are not affected.

```
Rack1R5#show ipv6 route eigrp
IPv6 Routing Table - Default - 16 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
D   2001:1:0:146::/64 [90/2172416]
     via FE80::215:62FF:FE41:FD9C, Serial0/1/0
     via FE80::4, Serial0/0/0
D   FC00:1::/60 [5/128256]
     via Null0, directly connected
D   FC00:1:0:4::/64 [90/2297856]
     via FE80::215:62FF:FE41:FD9C, Serial0/1/0
     via FE80::4, Serial0/0/0
D   FC00:1:0:8::/64 [90/156160]
     via FE80::218:73FF:FE8D:9E45, FastEthernet0/0

Rack1R5#show ipv6 route FC00:1:0:6::
Routing entry for FC00:9::/61
  Known via "rip RIPNG", distance 120, metric 3
  Route count is 3/3, share count 0
  Routing paths:
    FE80::4, Serial0/0/0
      Last updated 02:24:49 ago
    FE80::1, Serial0/0/0
      Last updated 02:24:47 ago
    FE80::21A:2FFF:FEFE:418, Serial0/1/0
      Last updated 02:20:56 ago
```

Copyright © 2010 Internetwork Expert

9.15 EIGRPv6 Metric Manipulation

- Allow EIGRPv6 to perform load balancing between multiple paths with unequal metrics so long as the metrics are within the range of the optimal metric scaled three times.
- Configure R4's Serial link to R5 to have a delay that is twice good as R4's Frame-Relay link.

Configuration

> **Note**
>
> EIGRPv6 uses the same automatic metric computation as EIGRP for IPv4. Thus, you may tune interface bandwidth and delay to affect the metric for the incoming updates. You may change the K-values vector by using the routing process command **`metric weight.`** Where IPv4 EIGRP supports unequal-cost load balancing, IPv6 EIGRP can only implement equal-cost load balancing as a result of existing IPv6 CEF limitations. In the future, these limitations may be lifted. Right now, however you can instruct EIGRPv6 to take in account paths with unequal metrics, but it will only load balance across them equally. This is accomplished by using the routing-process **`variance`** command to allow the process to install the routes with unequal costs into the routing table. Just as usual, paths selected for load balancing will include any feasible successor path with a metric less than the minimal metric scaled by the variance parameter, up to the maximum number of allowed paths.

R4, R5, R6, SW2:
```
ipv6 router eigrp 100
 metric weight 0 0 0 1 0 0
 variance 3
```

R4:
```
interface Serial 0/0/0
 delay 2000
!
interface Serial 0/1/0
 delay 1000
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the K-values and the variance setting.

```
Rack1R4#show ipv6 protocols
IPv6 Routing Protocol is "connected"
IPv6 Routing Protocol is "static"
IPv6 Routing Protocol is "rip RIPNG"
  Interfaces:
    Serial0/1/0
    Serial0/0/0
    Loopback100
    FastEthernet0/1
  Redistribution:
    None
IPv6 Routing Protocol is "eigrp 100"
  EIGRP metric weight K1=0, K2=0, K3=1, K4=0, K5=0
  EIGRP maximum hopcount 100
  EIGRP maximum metric variance 3
  Interfaces:
    FastEthernet0/1
    Serial0/0/0
    Serial0/1/0
    Loopback100
  Redistribution:
    None
  Maximum path: 16
  Distance: internal 90 external 170
```

> **Note**
>
> Check for multipath routing towards the prefix we summarized previously:

```
Rack1R4#show ipv6 route FC00:1::/60
Routing entry for FC00:1::/60
  Known via "eigrp 100", distance 90, metric 640000, type internal
  Route count is 2/2, share count 0
  Routing paths:
    FE80::5, Serial0/0/0
      Last updated 00:01:26 ago
    FE80::213:1AFF:FE68:B04C, Serial0/1/0
      Last updated 00:01:26 ago
```

Copyright © 2010 Internetwork Expert

Confirm that the IPv6 CEF table has 16 buckets equally allocated to both paths, even though their metrics differ:

```
Rack1R4# show ipv6 eigrp topology FC00:1::/60
IPv6-EIGRP (AS 100): Topology entry for FC00:1::/60
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is
384000
  Routing Descriptor Blocks:
  FE80::213:1AFF:FE68:B04C (Serial0/1/0), from
FE80::213:1AFF:FE68:B04C, Send flag is 0x0
      Composite metric is (384000/128000), Route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 15000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
  FE80::5 (Serial0/0/0), from FE80::5, Send flag is 0x0
      Composite metric is (640000/128000), Route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 25000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ipv6 cef FC00:1::/60 internal
FC00:1::/60, epoch 0, RIB[I], refcount 4, per-destination sharing
  sources: RIB
  feature space:
   IPRM: 0x00038000
  ifnums:
   Serial0/0/0(5): FE80::5
   Serial0/1/0(6): FE80::213:1AFF:FE68:B04C
  path 65E7BFC8, path list 65F9332C, share 1/1, type attached nexthop,
for IPv6
  nexthop FE80::5 Serial0/0/0, adjacency IPV6 adj out of Serial0/0/0,
addr FE80::5 65E7AAE0
  path 65E7BC28, path list 65F9332C, share 1/1, type attached nexthop,
for IPv6
  nexthop FE80::213:1AFF:FE68:B04C Serial0/1/0, adjacency IPV6 adj out
of Serial0/1/0 65E79E60
  output chain:
    loadinfo 6655FD68, per-session, 2 choices, flags 0005, 8 locks
    flags: Per-session, for-rx-IPv6
    16 hash buckets
      < 0 > IPV6 adj out of Serial0/0/0, addr FE80::5 65E7AAE0
      < 1 > IPV6 adj out of Serial0/1/0 65E79E60
      < 2 > IPV6 adj out of Serial0/0/0, addr FE80::5 65E7AAE0
      < 3 > IPV6 adj out of Serial0/1/0 65E79E60
      < 4 > IPV6 adj out of Serial0/0/0, addr FE80::5 65E7AAE0
      < 5 > IPV6 adj out of Serial0/1/0 65E79E60
      < 6 > IPV6 adj out of Serial0/0/0, addr FE80::5 65E7AAE0
      < 7 > IPV6 adj out of Serial0/1/0 65E79E60
      < 8 > IPV6 adj out of Serial0/0/0, addr FE80::5 65E7AAE0
      < 9 > IPV6 adj out of Serial0/1/0 65E79E60
      <10 > IPV6 adj out of Serial0/0/0, addr FE80::5 65E7AAE0
      <11 > IPV6 adj out of Serial0/1/0 65E79E60
      <12 > IPV6 adj out of Serial0/0/0, addr FE80::5 65E7AAE0
      <13 > IPV6 adj out of Serial0/1/0 65E79E60
      <14 > IPV6 adj out of Serial0/0/0, addr FE80::5 65E7AAE0
      <15 > IPV6 adj out of Serial0/1/0 65E79E60
    Subblocks:
     None
```

Copyright © 2010 Internetwork Expert

9.16 EIGRPv6 Default Routing

- Configure R6 to advertise a default route into the EIGRPv6 routing domain.

Configuration

> **Note**
>
> There are two ways to advertise a default route into a IPv6 EIGRP routing domain – using redistribution or using summarization. The first technique will originate the route as EIGRP an external route with an AD of 170 (one of the least preferred). The second technique will summarize all routes, as the default route encompasses all of them, and there is no leak-map in EGRPv6 as of yet to "unsuppress" certain prefixes. Thus, every technique has its drawbacks. In addition, notice, that using redistribution allows for specifying the default route metric explicitly, while summarization computes the metric automatically.
>
> In this task we use the summarization approach and summarize all prefixes under ::/0 on R6's interfaces towards R4. This ensures that the EIGRPv6 default route will preempt RIPng's default route.

```
R6:
interface FastEthernet0/0.146
 ipv6 summary-address eigrp 100 ::/0 5
```

Verification

> **Note**
>
> Make sure the default route propagates throughout the EIGRPv6 topology:

```
Rack1R5#show ipv6 route eigrp
IPv6 Routing Table - Default - 18 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
D   ::/0 [90/517120]
     via FE80::215:62FF:FE41:FD9C, Serial0/1/0
<Snip>
```

Copyright © 2010 Internetwork Expert

Notice that SW2 uses the static default route learned via auto-configuration because of its better AD:

```
Rack1SW2#show ipv6 route
IPv6 Routing Table - Default - 12 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       R - RIP, D - EIGRP, EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
S   ::/0 [1/0]
     via FE80::213:1AFF:FE68:B04C, Vlan58
<Snip>
```

Copyright © 2010 Internetwork Expert

9.17 OSPFv3

- Configure OSPFv3 as the routing protocol on the links connecting R3, SW1, and R6.
- Manually configure the router identifiers for all OSPFv3 processes to match the IPv4 Loopback 0 addresses on these routers.
- Use Area 0 on the link between SW1 and R6 and use Area 37 on the link between R3 and SW1.
- Change the network type to point-to-point on the link between R3 and SW1.
- Make the default hello/dead intervals on the segment connecting R6 and SW1 10 times less than the default.

Configuration

```
R3:
ipv6 unicast-routing
!
ipv6 router ospf 1
 router-id 150.1.3.3
!
interface FastEthernet 0/0
 ipv6 ospf 1 area 37
 ipv6 ospf network point-to-point

R6:
ipv6 router ospf 1
 router-id 150.1.6.6
!
interface FastEthernet 0/0.67
 ipv6 ospf hello-interval 1
 ipv6 ospf 1 area 0

SW1:
ipv6 unicast-routing
!
ipv6 router ospf 1
 router-id 150.1.7.7
!
interface Vlan 67
 ipv6 ospf hello-interval 1
 ipv6 ospf 1 area 0
!
interface FastEthernet 0/3
 ipv6 ospf 1 area 37
 ipv6 ospf network point-to-point
```

Copyright © 2010 Internetwork Expert

Verification

Note

OSPFv3 for IPv6 retains many concepts of OSPFv2 for IPv4; they have practically the same set of LSA types plus the concept of areas, ABRs, area types and so on. Even the router ID for OSPFv3 is the same 32-bit value, derived from the highest Loopback IPv4 address.

Most OSPFv3 configurations are done at the interface level. For instance instead of using the **network** command, under the OSPF process, you simply assign individual interfaces to the OSPFv3 process.

Like RIPng, many of the OSPFv3 verification commands use the same format as their OSPFv2 counterparts, with the exception of the **ipv6** keyword in place of the **ip** keyword.

```
Rack1SW1#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID  Interface
150.1.6.6         1   FULL/DR         00:00:03    16            Vlan67
150.1.3.3         1   FULL/  -        00:00:38    4             FastEthernet0/3

Rack1SW1#show ipv6 ospf 1
 Routing Process "ospfv3 1" with ID 150.1.7.7
 It is an area border router
 SPF schedule delay 5 secs, Hold time between two SPFs 10 secs
 Minimum LSA interval 5 secs. Minimum LSA arrival 1 secs
 LSA group pacing timer 240 secs
 Interface flood pacing timer 33 msecs
 Retransmission pacing timer 66 msecs
 Number of external LSA 0. Checksum Sum 0x000000
 Number of areas in this router is 2. 2 normal 0 stub 0 nssa
    Area BACKBONE(0)
        Number of interfaces in this area is 1
        SPF algorithm executed 3 times
        Number of LSA 7. Checksum Sum 0x02FE9B
        Number of DCbitless LSA 0
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0
    Area 37
        Number of interfaces in this area is 1
        SPF algorithm executed 3 times
        Number of LSA 7. Checksum Sum 0x0401D4
        Number of DCbitless LSA 0
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0
```

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show ipv6 ospf interface fastEthernet 0/3
FastEthernet0/3 is up, line protocol is up (connected)
  Link Local Address FE80::212:1FF:FE31:41, Interface ID 1021
  Area 37, Process ID 1, Instance ID 0, Router ID 150.1.7.7
  Network Type POINT_TO_POINT, Cost: 1
  Transmit Delay is 1 sec, State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:01
  Index 1/1/2, flood queue length 0
  Next 0x0(0)/0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.3.3
  Suppress hello for 0 neighbor(s)

Rack1SW1#show ipv6 ospf interface Vlan 67
Vlan67 is up, line protocol is up
  Link Local Address FE80::212:1FF:FE31:43, Interface ID 2135
  Area 0, Process ID 1, Instance ID 0, Router ID 150.1.7.7
  Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State BDR, Priority 1
  Designated Router (ID) 150.1.6.6, local address
FE80::20C:85FF:FEC1:FC60
  Backup Designated router (ID) 150.1.7.7, local address
FE80::212:1FF:FE31:43
  Timer intervals configured, Hello 1, Dead 4, Wait 4, Retransmit 5
    Hello due in 00:00:00
  Index 1/1/1, flood queue length 0
  Next 0x0(0)/0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 2
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.6.6  (Designated Router)
  Suppress hello for 0 neighbor(s)
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

9.18 OSPFv3 over NBMA

- Configure OSPFv3 area 0 on the Frame-Relay links of R5, R2, and R3.
- Use a network type that does not elect a DR, and ensure that OSPF does not send broadcast hello packets.

Configuration

```
R2:
ipv6 unicast-routing
!
ipv6 router ospf 1
 router-id 150.1.2.2
!
interface Serial 0/0
 ipv6 ospf 1 area 0
 ipv6 ospf network point-to-multipoint non-broadcast
 ipv6 ospf neighbor fe80::5

R3:
interface Serial 1/0
 ipv6 ospf 1 area 0
 ipv6 ospf network point-to-multipoint non-broadcast
 ipv6 ospf neighbor fe80::5

R5:
ipv6 router ospf 1
 router-id 150.1.5.5
!
interface Serial 0/0/0
 ipv6 ospf 1 area 0
 ipv6 ospf network point-to-multipoint non-broadcast
 ipv6 ospf neighbor fe80::3
 ipv6 ospf neighbor fe80::2
```

Copyright © 2010 Internetwork Expert

Verification

Note

The task wording prompts us to use the multi-point non-broadcast network type. Just like in OSPFv2, this network type supports multiple neighbors on the link, sends unicast updates, and does not perform the DR/BDR election. Note that in OSPFv3 you configure the neighbors at the interface level. As usual, properly mapping the link-local IPv6 addresses is required so that OSPF can exchange packets. Note the use of link-local IPv6 addresses with the **`neighbor`** command.

With the multi-point network-type you can also specify the neighbor cost, like with OSPFv2.

```
Rack1R5#show ipv6 ospf neighbor
Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
150.1.2.2         1   FULL/  -        00:01:55    5               Serial0/0/0
150.1.3.3         1   FULL/  -        00:01:45    6               Serial0/0/0

Rack1R5#show ipv6 ospf int se 0/0/0
Serial0/0 is up, line protocol is up
  Link Local Address FE80::5, Interface ID 6
  Area 0, Process ID 1, Instance ID 0, Router ID 150.1.5.5
  Network Type POINT_TO_MULTIPOINT, Cost: 64
  Transmit Delay is 1 sec, State POINT_TO_MULTIPOINT,
  Timer intervals configured, Hello 30, Dead 120, Wait 120,Retransmit 5
    Hello due in 00:00:24
  Index 1/1/1, flood queue length 0
  Next 0x0(0)/0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 3
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 2, Adjacent neighbor count is 2
    Adjacent with neighbor 150.1.2.2
    Adjacent with neighbor 150.1.3.3
  Suppress hello for 0 neighbor(s)
```

Copyright © 2010 Internetwork Expert

9.19 OSPFv3 Virtual Links

- Restore the discontiguous OSPFv3 Area 0 by provisioning a virtual link between R3 and SW1.

Configuration

```
R3:
ipv6 router ospf 1
 area 37 virtual-link 150.1.7.7

SW1:
ipv6 router ospf 1
 area 37 virtual-link 150.1.3.3
```

Verification

> **Note**
>
> The concept of the virtual-link remains the same in OSPFv3 as it does in OSPFv2. Using it, you may repair a discontiguous backbone, attach a non-zero area to the backbone across a transit area, or perform traffic engineering.
>
> We configure Virtual-links under the global OSPFv3 process configuration mode. Check the virtual-link health and verify that you can see Area 0 link prefixes on R6.

```
Rack1R3#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID  Interface
150.1.7.7         1   FULL/  -        00:00:36    2033          OSPFv3_VL0
150.1.5.5         1   FULL/  -        00:01:59    6             Serial1/0
150.1.7.7         1   FULL/  -        00:00:38    1021          FastEthernet0/0

Rack1R3#show ipv6 ospf virtual-links
Virtual Link OSPFv3_VL0 to router 150.1.7.7 is up
  Interface ID 17, IPv6 address FC00:1:0:37::7
  Run as demand circuit
  DoNotAge LSA allowed.
  Transit area 37, via interface FastEthernet0/0, Cost of using 1
  Transmit Delay is 1 sec, State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Adjacency State FULL (Hello suppressed)
    Index 1/2/3, retransmission queue length 0, number of
retransmission 0
    First 0x0(0)/0x0(0)/0x0(0) Next 0x0(0)/0x0(0)/0x0(0)
    Last retransmission scan length is 0, maximum is 0
    Last retransmission scan time is 0 msec, maximum is 0 msec
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ipv6 route ospf
IPv6 Routing Table - 16 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:1:0:1234::2/128 [110/847]
     via FE80::212:1FF:FE31:43, FastEthernet0/0.67
O   2001:1:0:1234::3/128 [110/2]
     via FE80::212:1FF:FE31:43, FastEthernet0/0.67
O   2001:1:0:1234::5/128 [110/783]
     via FE80::212:1FF:FE31:43, FastEthernet0/0.67
OI  FC00:1:0:37::/64 [110/2]
     via FE80::212:1FF:FE31:43, FastEthernet0/0.67
OI  FC00:1:0:37::1/128 [110/2]
     via FE80::212:1FF:FE31:43, FastEthernet0/0.67
OI  FC00:1:0:37::7/128 [110/1]
     via FE80::212:1FF:FE31:43, FastEthernet0/0.67
```

Copyright © 2010 Internetwork Expert

9.20 OSPFv3 Summarization

- Configure OSPFv3 Area 58 on the link between SW2 and R5.
- Create two new interfaces, Loopback 100 and Loopback 101 on SW2 with IPv6 addresses FC00:X:0:8::8/64 and FC00:X:0:88::88/64 respectively.
- Advertise both Loopbacks into OSPF Area 58 on SW2.
- Configure R5 to summarize the two subnets to an optimal prefix.

Configuration

```
R5:
ipv6 router ospf 1
 area 58 range fc00:1::/56
!
interface FastEthernet 0/0
 ipv6 ospf 1 area 58

SW2:
ipv6 unicast-routing
!
interface Vlan 58
 ipv6 ospf 1 area 58
!
interface Loopback100
 ipv6 address fc00:1:0:8::8/64
 ipv6 ospf 1 area 58
!
interface Loopback101
 ipv6 address fc00:1:0:88::88/64
 ipv6 ospf 1 area 58
```

Verification

Note

Summarization concepts are the same for OSPFv3 as for OSPFv2. Even the commands remain the same; you just need to apply them under the OSPFv3 process context.

When summarizing the prefixes fc00:1:0:8::/64 and fc001:0:88::/64 remember that 8 and 88 are in hex. Convert them to binary format using the Windows calculator application.

88 = 0000 0000 **1000 1000**
8 = 0000 0000 **0000 1000**

Therefore, to summarize, shrink the prefix length by 8, to remove the mismatched parts. The resulting prefix is: fc00:1:0::/56.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID  Interface
150.1.2.2         1   FULL/  -        00:01:37    5             Serial0/0
150.1.3.3         1   FULL/  -        00:01:50    6             Serial0/0
150.1.8.8         1   FULL/DR         00:00:35    2126          FastEthernet0/0

Rack1R5#show ipv6 route ospf
IPv6 Routing Table - 20 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:1:0:1234::2/128 [110/64]
     via FE80::2, Serial0/0
O   2001:1:0:1234::3/128 [110/64]
     via FE80::3, Serial0/0
O   FC00:1::/56 [110/0]
     via ::, Null0
O   FC00:1:0:8::8/128 [110/1]
     via FE80::21B:D4FF:FED4:4D45, FastEthernet0/0
OI  FC00:1:0:37::/64 [110/65]
     via FE80::3, Serial0/0
OI  FC00:1:0:37::1/128 [110/64]
     via FE80::3, Serial0/0
OI  FC00:1:0:37::7/128 [110/65]
     via FE80::3, Serial0/0
O   FC00:1:0:67::/64 [110/66]
     via FE80::3, Serial0/0
O   FC00:1:0:88::88/128 [110/1]
     via FE80::21B:D4FF:FED4:4D45, FastEthernet0/0

Rack1R3#show ipv6 route ospf
IPv6 Routing Table - 12 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:1:0:1234::2/128 [110/845]
     via FE80::5, Serial1/0
O   2001:1:0:1234::5/128 [110/781]
     via FE80::5, Serial1/0
OI  FC00:1::/56 [110/782]
     via FE80::5, Serial1/0
O   FC00:1:0:37::7/128 [110/1]
     via FE80::212:1FF:FE31:41, FastEthernet0/0
O   FC00:1:0:67::/64 [110/2]
     via FE80::212:1FF:FE31:41, FastEthernet0/0

Rack1SW1#show ipv6 route ospf
IPv6 Routing Table - 11 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:1:0:1234::2/128 [110/846]
     via FE80::20C:CEFF:FE4B:A2A0, FastEthernet0/3
O   2001:1:0:1234::3/128 [110/1]
     via FE80::20C:CEFF:FE4B:A2A0, FastEthernet0/3
O   2001:1:0:1234::5/128 [110/782]
```

Copyright © 2010 Internetwork Expert

```
     via FE80::20C:CEFF:FE4B:A2A0, FastEthernet0/3
OI   FC00:1::/56 [110/783]
     via FE80::20C:CEFF:FE4B:A2A0, FastEthernet0/3
O    FC00:1:0:37::1/128 [110/1]
     via FE80::20C:CEFF:FE4B:A2A0, FastEthernet0/3

Rack1R6#show ipv6 route ospf
IPv6 Routing Table - 17 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O    2001:1:0:1234::2/128 [110/847]
     via FE80::212:1FF:FE31:43, FastEthernet0/0.67
O    2001:1:0:1234::3/128 [110/2]
     via FE80::212:1FF:FE31:43, FastEthernet0/0.67
O    2001:1:0:1234::5/128 [110/783]
     via FE80::212:1FF:FE31:43, FastEthernet0/0.67
OI   FC00:1::/56 [110/784]
     via FE80::212:1FF:FE31:43, FastEthernet0/0.67
OI   FC00:1:0:37::/64 [110/2]
     via FE80::212:1FF:FE31:43, FastEthernet0/0.67
OI   FC00:1:0:37::1/128 [110/2]
     via FE80::212:1FF:FE31:43, FastEthernet0/0.67
OI   FC00:1:0:37::7/128 [110/1]
     via FE80::212:1FF:FE31:43, FastEthernet0/0.6
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

9.21 IPv6 Redistribution

- Perform mutual redistribute between OSPFv3, RIPng and EIGRPv6 on R5.
- Ensure each protocol prefers to reach native prefixes using internal paths.

Configuration

```
R5:
ipv6 router ospf 1
 redistribute rip RIPNG metric 8
 redistribute eigrp 100 metric 8
 redistribute connected metric 8
!
ipv6 router rip RIPNG
 redistribute eigrp 100 include-connected metric 8
 redistribute ospf 1 include-connected metric 8
!
ipv6 router eigrp 100
 redistribute rip RIPNG include-connected metric 1000 0 255 1 1500
 redistribute ospf 1 include-connected metric 1000 0 255 1 1500
!

R6:
ipv6 router ospf 1
 distance ospf external 171
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> When redistributing between two IPv6 IGPs, remember that the command **`redistribute <protocol>`** will not redistribute the locally connected interfaces advertised into <protocol>. For example, on R6, if you configure **`redistribute rip RIPNG`**, only the routes learned by R6 via RIPng will be redistributed (not the local Loopback 100, which is being advertised into RIPng).
>
> In order to account for directly connected subnets, use the separate command **`redistribute connected`** or use the **`include-connected`** keyword when performing redistributing.
>
> Since OSPFv3 has a better administrative distance than RIPng or EIGRPv6, adjust the OSPFv3 external distance on R6 to make sure that external OSPFv3 prefixes are not preferred over internal RIPng prefixes.
>
> The rest of the redistribution rules match those used when redistributing between IPv4 routing protocols. In this task, we also change the AD for OSPFv3 external routes on R6, as this is where all three domains connect and the OSPF external routes will "preempt" RIPng internals. We must tune the individual OSPF external route administrative distances to avoid this.

```
Rack1R6#show ipv6 route rip
IPv6 Routing Table - Default - 27 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   FC00:1::/61 [120/2]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0.146
     via FE80::213:80FF:FEAD:7A80, FastEthernet0/0.146
R   FC00:1:0:8::/64 [120/10]
     via FE80::213:80FF:FEAD:7A80, FastEthernet0/0.146
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0.146
R   FC00:1:0:8::8/128 [120/10]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0.146
     via FE80::213:80FF:FEAD:7A80, FastEthernet0/0.146
R   FC00:1:0:88::88/128 [120/10]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0.146
     via FE80::213:80FF:FEAD:7A80, FastEthernet0/0.146
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ipv6 route eigrp
IPv6 Routing Table - Default - 27 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
D   ::/0 [5/2560]
     via Null0, directly connected
D   2001:1:0:1234::/64 [90/514560]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0.146
D   FC00:1::/60 [90/258560]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0.146
D   FC00:1:0:4::/64 [90/130560]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0.146
D   FC00:1:0:58::/64 [90/261120]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0.146
D   FC00:1:0:85::/64 [90/261120]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0.146

Rack1R6#show ipv6 route ospf
IPv6 Routing Table - Default - 27 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:1:0:1234::2/128 [110/847]
     via FE80::211:BBFF:FE0A:C8C3, FastEthernet0/0.67
O   2001:1:0:1234::3/128 [110/2]
     via FE80::211:BBFF:FE0A:C8C3, FastEthernet0/0.67
O   2001:1:0:1234::5/128 [110/783]
     via FE80::211:BBFF:FE0A:C8C3, FastEthernet0/0.67
OI  FC00:1::/56 [110/784]
     via FE80::211:BBFF:FE0A:C8C3, FastEthernet0/0.67
OE2 FC00:1:0:1::/64 [171/8]
     via FE80::211:BBFF:FE0A:C8C3, FastEthernet0/0.67
OE2 FC00:1:0:5::/64 [171/8]
     via FE80::211:BBFF:FE0A:C8C3, FastEthernet0/0.67
OI  FC00:1:0:37::/64 [110/2]
     via FE80::211:BBFF:FE0A:C8C3, FastEthernet0/0.67
OI  FC00:1:0:37::3/128 [110/2]
     via FE80::211:BBFF:FE0A:C8C3, FastEthernet0/0.67
OI  FC00:1:0:37::7/128 [110/1]
     via FE80::211:BBFF:FE0A:C8C3, FastEthernet0/0.67
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

9.22 IPv6 Filtering

- Configure R3's Frame-Relay interface so that only FTP and HTTP traffic from the users on VLAN 67 can access the interface.
- Allow DNS queries and responses from the same subnet, and ensure IPv6 routing will not be affected.
- Enable HTTP Server on R5

Configuration

```
R3:
ipv6 access-list FILTER_OUT
 permit tcp fc00:1:0:67::/64 any eq 80
 permit tcp fc00:1:0:67::/64 any range 20 21
 permit udp fc00:1:0:67::/64 any eq 43
!
ipv6 access-list FILTER_IN
 permit tcp any eq 80 fc00:1:0:67::/64
 permit tcp any range 20 21 fc00:1:0:67::/64
 permit udp any eq 43 fc00:1:0:67::/64
 permit 89 any any
!
interface Serial 1/0
 ipv6 traffic-filter FILTER_OUT out
 ipv6 traffic-filter FILTER_IN in

R5
ip http server
```

Verification

Note

Cisco IOS supports only extended IPv6 access-lists. The syntax remains the same, except the source/destination specifications. Instead of using subnets and wildcard masks, you specify IPv6 prefixes. Aside from that, the logic of the access-lists remains the same. The access-lists are applied as traffic-filters to the interfaces.

Note that IPv6 access-lists do not have a keyword to permit OSPFv3 explicitly, so you need to use its protocol number, which is 89.

Copyright © 2010 Internetwork Expert

Generate some traffic to test the access-lists from SW1.

```
Rack1SW1#ping 2001:1:0:1234::3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:1:0:1234::3, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/0 ms

Rack1SW1#ping 2001:1:0:1234::5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:1:0:1234::5, timeout is 2
seconds:
AAAAA
Success rate is 0 percent (0/5)
```

Check that OSPFv3 adjacencies are OK on R3.

```
Rack1R3#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID  Interface
150.1.7.7         1   FULL/  -          -         2033          OSPFv3_VL0
150.1.5.5         1   FULL/  -        00:01:48    6             Serial1/0
150.1.7.7         1   FULL/  -        00:00:31    1021          FastEthernet0/0
```

Verify that the HTTP filtering is working:

```
Rack1SW1#telnet 2001:1:0:1234::5 80
Trying 2001:1:0:1234::5, 80 ...
% Destination unreachable; gateway or host down

Rack1SW1#telnet 2001:1:0:1234::5 80 /source-interface vlan 67
Trying 2001:1:0:1234::5, 80 ... Open
```

Copyright © 2010 Internetwork Expert

9.23 IPv6 NAT-PT

- Configure R6 so that SW1 can access the IPv4 address 150.X.4.4 as 2000::9601:404.
- SW1 should source IPv6 packets off its VLAN 67 interface and R6 should translate it to the IPv4 address 155.X.146.7.
- Create a static route on SW1 to ensure traffic for 2000::/96 does not transit SW1's Fa0/3 to reach R6.

Configuration

```
R6:
interface FastEthernet 0/0.67
 ipv6 nat
!
interface FastEthernet 0/0.146
 ipv6 nat
!
ipv6 nat v6v4 source fc00:1:0:67::7 155.1.146.7
ipv6 nat v4v6 source 150.1.4.4 2000::9601:0404
!
ipv6 nat prefix 2000::/96
!
ipv6 router rip RIPNG
 redistribute connected

SW1
!
ipv6 route 2000::/96 fc00:1:0:67::6
```

Verification

Note

IPv6 NAT-PT allows IPv6 and IPv4 domains to communicate by rewriting packet headers and replacing addresses. In contrast to IPv4 NAT, both the source and destination addresses of every packet must be rewritten. Specifically, IPv6 NAT-PT works as follows:

A block of IPv6 addresses will represent the IPv4 address space. This block usually has a /96 prefix length, to cover all 2^{32} IPv4 addresses, but otherwise it could be arbitrary. If needed, we can map selected IPv4 addresses to respective IPv6 addresses, using static translation rules. When an IPv6 host needs to communicate with an IPv4 host, it sends an IPv6 packet to the address within the /96 block. The translating router intercepts the packet going to the selected /96 prefix, and replaces the IPv6 header with an IPv4 header, rewriting both the source and the destination IPv6 addresses with IPv4 addresses. When a response IPv4 packet returns, the router performs the reverse translation.

Copyright © 2010 Internetwork Expert

In order to configure IPv6 NAT-PT, you have to specify the following:

1) Rules to translate IPv4 source addresses to IPv6 addresses
2) Rules to translate IPv6 source addresses to IPv4 addresses
3) The /96 prefix to map the IPv4 address space to

To verify this configuration, enable IPv6 NAT-PT debugging on R6 and send packets from SW1 to R4.

```
Rack1R6#debug ipv6 nat detailed
IPv6 NAT-PT detailed debugging is on

Rack1SW1#ping ipv6 2000::9601:0404

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2000::9601:404, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/4/8 ms
Rack1SW1#

Rack1R6#
IPv6 NAT: ipv6nat_find_entry_v4tov6:
         ref_count = 1,
                                usecount = 0, flags = 64, rt_flags = 0,
                                more_flags = 0

IPv6 NAT: icmp src (FC00:1:0:67::7) -> (155.1.146.7), dst
(2000::9601:404) -> (150.1.4.4)
IPv6 NAT: ipv6nat_find_entry_v4tov6:
         ref_count = 1,
                                usecount = 0, flags = 64, rt_flags = 0,
                                more_flags = 0

IPv6 NAT: icmp src (150.1.4.4) -> (2000::9601:404), dst (155.1.146.7) -
> (FC00:1:0:67::7)
IPv6 NAT: ipv6nat_find_entry_v4tov6:
         ref_count = 1,
                                usecount = 0, flags = 64, rt_flags = 0,
                                more_flags = 0
```

9.24 IPv6 MP-BGP

- Configure R1 and R5 in BGP autonomous systems 100 and 500 respectively, and form a peering relationship over the Frame-Relay cloud.
- Create two new loopback interfaces on R1 with IPv6 addresses 2003:X:0:1::1/64 and 2003:X:0:11::11/61, and advertise them into BGP.
- Ensure R5 sees only an optimal summary route encompassing both IPv6 prefixes.

Configuration

```
R1:
interface Loopback 101
 ipv6 address 2003:1:0:1::1/64
!
interface Loopback 102
 ipv6 address 2003:1:0:11::11/61
!
router bgp 100
 address-family ipv6 unicast
  neighbor 2001:1:0:1234::5 remote-as 500
  neighbor 2001:1:0:1234::5 activate
  network 2003:1:0:1::/64
  network 2003:1:0:10::/61
  aggregate-address 2003:1::/59 summary-only

R5:
router bgp 500
 address-family ipv6 unicast
  neighbor 2001:1:0:1234::1 remote-as 100
  neighbor 2001:1:0:1234::1 activate
```

Copyright © 2010 Internetwork Expert

Verification

Note

Multi-protocol extensions for BGP adds supports for the IPv6 address family, which essentially means that IPv6 prefixes and attributes can be exchanged over a normal TCP based BGP peering. We configure IPv6 peers under the **`address-family ipv6 unicast`** configuration mode inside of BGP, and then they must be "activated", or enabled. Technically this is the same process used in IPv4 peering, but the normal global BGP process automatically refers to the **`address-family ipv4 unicast`**, and the command **`bgp default ipv4-unicast`** is enabled by default, which means that all unicast IPv4 peers are automatically "activated".

The configuration of all IPv6 related options takes place under the address-family, such as the network statement, or applying attributes onto neighbors like route-maps, prefix-lists, filter-lists, etc. Keep in mind that all other BGP operations stay the same, such as advertisement rules, route reflection, confederation, best path selection, etc. Mainly the only thing that has changed is the format of the prefix and next-hop advertised inside of the BGP update messages.

Note that for EBGP peerings the next-hop values recurse to link-local addresses in the routing table, so when peering over multi-point NBMA, we must map the link-local IPv6 addresses as with the other IPv6 protocols we have discussed.

Prefix summarization uses the same command used in IPv4 and the same prefix length calculations. In our case, take the mismatched parts of the prefixes (remember, 11 is in hex, thus it is 17 in decimal).

0000 0000 000**0 0001**
0000 0000 000**1 0**

(note that the second prefix length is 61, not 64). To build the summary, shift the prefix length left by 5 bits. The resulting prefix is:

2003:1::/59

Copyright © 2010 Internetwork Expert

```
Rack1R1#show bgp ipv6 unicast summary
BGP router identifier 150.1.1.1, local AS number 100
BGP table version is 6, main routing table version 6
3 network entries using 447 bytes of memory
3 path entries using 228 bytes of memory
3/2 BGP path/bestpath attribute entries using 372 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 1047 total bytes of memory
BGP activity 3/0 prefixes, 3/0 paths, scan interval 60 secs

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down
State/PfxRcd
2001:1:0:1234::5
                4   500       4       5        6    0    0 00:00:03
0

Rack1R1#show bgp ipv6 unicast
BGP table version is 6, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 2003:1::/59      ::                                32768 i
s> 2003:1:0:1::/64  ::                         0      32768 i
s> 2003:1:0:10::/61 ::                         0      32768 i

Rack1R5#show bgp ipv6 unicast
BGP table version is 8, local router ID is 150.1.5.5
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 2003:1::/59      2001:1:0:1234::1
                                               0             0 100 i
```

Copyright © 2010 Internetwork Expert

9.25 IPv6 PIM and MLD

- Enable IPv6 Multicast routing on R1, R3, R4, R5 and R6.
- Ensure PIM does not run on the Frame-Relay link between R4 and R5.
- Configure R5 to join the multicast group FF08::8/128 on its VLAN 58 interface.
- R5 should only accept MLD Reports for the group range FF08::/64 and should limit the maximum number of multicast states on the interface towards SW2 to 10.
- Send the general queries every 10 seconds on R5's VLAN 58 interface.

Configuration

Note

In IPv4, there is a specific range of addresses reserved for multicast use (224.0.0.0 to 239.255.255.255). In IPv6, multicast addresses are all addresses inside the prefix FF00::/8. In other words, an IPv6 address is a multicast address if and only if the first byte is FF (255 in decimal). An IPv6 address is 128-bits long and these bits are divided into groups that are used to signify specific things. After the first 8 bits (which define a multicast address), there are four-bits each for flags and scopes. This leaves 112-bits left to specify a group ID, which due to the expansive size of the IPv6 address space is massive compared to IPv4 addressing.

Now we are going to look at the flag bits. Currently, the first three of these four bits are unused (and subsequently set to 0). The forth flag bit is called the transient bit. Its purpose is to indicate whether an address is permanent or temporary. If the address is permanently assigned (issued by IANA), this bit is set to zero. However, if the transient bit is a one then the address is temporary (or transient). Perfect examples of permanently assigned addresses would be FF02::2 (All Routers) or FF02::6 (OSPF DR Routers).

The remaining four bits of interest to us in a multicast address are known as the Scope ID bits. There are 16 different combinations than can be created using these 4-bits. Not all 16 available values are currently in use at this time, but seven of them are used to determine an address' scope. As an example, if the address scope is 1110 or "global", then that address is valid across the entire Internet. These are the currently defined scope ID bits:

Decimal Value	Binary Value	Address Scope
0	0000	Reserved
1	0001	Node-Local Scope
2	0010	Link-Local Scope
5	0101	Site Local Scope

Copyright © 2010 Internetwork Expert

8	1000	Organization Local Scope
14	1110	Global Scope
15	1111	Reserved

It needs to be noted that enforcement of flags and flooding scopes does not take place in software in any way. In addition to that, flooding scope enforcement must be configured administratively using multicast filtering. The prefixes just use the field for ease of administrative management.

Most of IPv6 multicast functionality is similar to IPv4 multicast, with some modifications. Therefore, we highly recommend you complete the **Multicast** section of this workbook before continuing further into IPv6 multicasting.

The following protocols should be understood thoroughly before continuing since they are the fundamental core of IPv6 multicasting:

PIMv2 for IPv6 is very similar to standard PIMv2 in its functionality, but it supports only Sparse-Mode operation. There is no Dense Mode flooding feature in IPv6 multicast protocol. Many of IPv4 PIMv2's commands and concepts transition to its IPv6 counterpart, including Designated Router, Assert, and Rendezvous Point etc. IPv6 multicast uses the unicast routing table (IPv6 RIB) for RPF checks against multicast sources. This means that multicast traffic propagation for IPv6 takes place in the same fashion as IPv4 Multicasting.

As soon as you type in the command **ipv6 multicast-routing** PIMv6 becomes active on all IPv6 capable interfaces. You may disable this on an interface-by-interface basis by using the command **no ipv6 pim**. As usual, all PIM-enabled interfaces are capable of flooding multicast traffic, provided there is a multicast tree built.

MLD – Multicast Listener Discovery protocol, based on ICMPv6, has replaced IGMP. IGMP is the membership signaling protocol used in IPv4 multicasting. MLDv1 translates to IGMPv2 and MLDv2 translates to IGMPv3, allowing for Source Specific IPv6 multicast.

MLD supports three message types very similar to IGMP's messages: Query, Report and Done, which is equivalent to IGMPv2's Leave. Configuring MLD is similar to configuring IGMP – you may define the maximum number of groups joined by the hosts on the interface using the command **ipv6 mld limit**, or join a group by using the command ipv6 mld join-group. You may also set various MLD query-related parameters using the commands **ipv6 mld query-interval**, **ipv6 mld query-timeout**, **ipv6 mld query-max-reponse-time**. The MLD timers are the same as those used with IGMPv2 or IGMPv3.

Copyright © 2010 Internetwork Expert

Notice the use of an IPv6 access-list below to control the groups that a host may join. The access-list entry **{permit|deny} ipv6 <part1> <part2>** allows for both SSM and normal multicast group filtering. If you want to filter just MLDv1 joins, leave **<part1>** as "any" this specifies any source. The parameter configured in **<part2>** is responsible for the multicast-group being joined. So it is possible with this one filter to permit or deny what specific multicast groups can be joined by specific hosts.

```
R1, R3, R4, R5, R6:
ipv6 multicast-routing

R1:
interface Serial 0/1
 no ipv6 pim

R3:
interface Serial 1/2
 no ipv6 pim

R4:
interface Serial 0/0/0
 no ipv6 pim

R5:
ipv6 access-list MLD_FILER
 permit ipv6 any ff08::/64
!
interface FastEthernet 0/0
 ipv6 mld access-group MLD_FILTER
 ipv6 mld join-group ff08::8
 ipv6 mld query-interval 10
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Verify PIM interfaces and neighbors on every router. Here we show just R5's status. Notice that R4 does not show up on the list of the neighbors over the Frame-Relay interface.

```
Rack1R5#show ipv6 pim interface
Interface          PIM  Nbr   Hello  DR
                        Count Intvl  Prior

Loopback0          off  0     30     1
    Address: ::
    DR     : not elected
FastEthernet0/0    on   0     30     1
    Address: FE80::213:1AFF:FE68:B04C
    DR     : this system
FastEthernet0/1    off  0     30     1
    Address: ::
    DR     : not elected
Serial0/0/0        on   3     30     1
    Address: FE80::5
    DR     : this system
Serial0/1/0        on   1     30     1
    Address: FE80::213:1AFF:FE68:B04C
    DR     : FE80::215:62FF:FE41:FD9C
SSLVPN-VIF0        off  0     30     1
    Address: ::
    DR     : not elected
Loopback100        on   0     30     1
    Address: FE80::213:1AFF:FE68:B04C
    DR     : this system
Loopback101        on   0     30     1
    Address: FE80::213:1AFF:FE68:B04C
    DR     : this system
Tunnel0            off  0     30     1
    Address: FE80::213:1AFF:FE68:B04C
    DR     : not elected

Rack1R5#show ipv6 pim neighbor
PIM Neighbor Table
Mode: B - Bidir Capable, G - GenID Capable
Neighbor Address           Interface          Uptime    Expires  Mode
DR pri

FE80::1                    Serial0/0/0        00:21:34  00:01:27 B G
1
FE80::3                    Serial0/0/0        00:21:34  00:01:30 B G
1
FE80::215:62FF:FE41:FD9C   Serial0/1/0        00:21:34  00:01:31 B G
DR 1
```

Copyright © 2010 Internetwork Expert

9.26 IPv6 PIM BSR

- Configure R6 as the RP for the multicast domain and R4 as the BSR to propagate this information.
- R3 should be able to ping the multicast group joined by R5.

Configuration

Note

PIMv6 RP and BSR are very similar to their IPv4 counterparts. However, there are a number of small configuration differences. The command to configure a router to announce itself as a RP via the BSR protocol is **ipv6 pim bsr candidate rp <IPv6 Address>** (not the interface name). The command to configure a router to start distributing RP bootstrap information and participate in BSR elections is **ipv6 pim bsr candidate bsr <IPv6 Address>.** As usual, you can configure a single router as both a RP and a BSR. One additional feature of an IPv6 BSR is that you may statically program a BSR with a list of candidate RPs using the command **ipv6 pim bsr announced rp <IPv6 Address>** and thus eliminate the need for dynamic RP advertisements by the candidates.

The only significant difference between the IPv4 PIM and IPv6 PIM is the source registration process. When a source starts multicasting, the DR on the segment is supposed to register it with the RP. This is normally performed by sending PIM register messages to the RP and encapsulating the original data packet inside them. As soon as a multicast IPv6 router learns the RP address it creates a tunnel interface connecting the router to the RP router. This tunnel is automatically enabled for multicasting, so multicast traffic sent down the shared tree is simply forwarded out the tunnel towards the RP. The tunnel is used only for the duration of the registration process, and then the receivers switch to the optimal path, which will not transit the tunnel. The use of this tunnel allows consistent multicast packet flooding behavior using the same code and simple tunneling of PIM registration messages. You may see a list of all currently active tunnels using the command **show ipv6 pim tunnel**

In the solution below, you will notice a static multicast route configured on R1. This is needed because R1 learns BSR messages from the Loopback 0 interface of R5 via its Ethernet interface, which breaks the RPF rule. You may observe this by issuing the command debug ipv6 pim bsr:

```
IPv6 BSR: Received BSR message from FE80::215:62FF:FE41:FD9D for
FC00:1:0:4::4, BSR priority 100 hash mask length 126
```

Copyright © 2010 Internetwork Expert

```
IPv6 BSR: BSR message from FE80::215:62FF:FE41:FD9D/FastEthernet0/0 for
FC00:1:0:4::4 RPF failed, dropped
```

The static multicast route fixes this problem and allows R1 to learn the RP information. The next-hop IPv6 address is the link-local IPv6 address of R4 on the VLAN 146 segment. A similar static route is configured on R5 for R4's Loopback 100 interface, to make it accept the BSR messages as well.

R6:

```
ipv6 pim bsr candidate rp fc00:1:0:6::6 priority 100
```

R4:

```
ipv6 pim bsr candidate bsr fc00:1:0:4::4 priority 100
```

R5:

```
ipv6 route fc00:1:0:4::/64 Serial 0/1/0 multicast
```

R1:

```
ipv6 route FC00:1:0:4::/64 FastEthernet0/0 FE80::215:62FF:FE41:FD9D
multicast
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Use the commands below to display the BSR election information and the RP to group mappings. You need to know that all routers have learned the BSR's information prior to starting any tests.

```
Rack1R1#show ipv6 pim bsr election
PIMv2 BSR information

BSR Election Information
  Scope Range List: ff00::/8
     BSR Address: FC00:1:0:4::4
     Uptime: 00:34:51, BSR Priority: 100, Hash mask length: 126
     RPF: FE80::215:62FF:FE41:FD9D,FastEthernet0/0
     BS Timer: 00:01:24

Rack1R5#show ipv6 pim bsr election
PIMv2 BSR information

BSR Election Information
  Scope Range List: ff00::/8
     BSR Address: FC00:1:0:4::4
     Uptime: 00:26:12, BSR Priority: 100, Hash mask length: 126
     RPF: FE80::215:62FF:FE41:FD9C,Serial0/1/0
     BS Timer: 00:02:02
```

The following are the mappings of RPs to the multicast group ranges:

```
Rack1R1#show ipv6 pim range-list
Static SSM Exp: never Learnt from : ::
  FF33::/32 Up: 00:36:24
  FF34::/32 Up: 00:36:24
  FF35::/32 Up: 00:36:24
  FF36::/32 Up: 00:36:24
  FF37::/32 Up: 00:36:24
  FF38::/32 Up: 00:36:24
  FF39::/32 Up: 00:36:24
  FF3A::/32 Up: 00:36:24
  FF3B::/32 Up: 00:36:24
  FF3C::/32 Up: 00:36:24
  FF3D::/32 Up: 00:36:24
  FF3E::/32 Up: 00:36:24
  FF3F::/32 Up: 00:36:24
BSR SM RP: FC00:1:0:6::6 Exp: 00:02:17 Learnt from : FC00:1:0:4::4
  FF00::/8 Up: 00:04:13
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#sh ipv6 pim range-list
Static SSM Exp: never Learnt from : ::
  FF33::/32 Up: 00:50:47
  FF34::/32 Up: 00:50:47
  FF35::/32 Up: 00:50:47
  FF36::/32 Up: 00:50:47
  FF37::/32 Up: 00:50:47
  FF38::/32 Up: 00:50:47
  FF39::/32 Up: 00:50:47
  FF3A::/32 Up: 00:50:47
  FF3B::/32 Up: 00:50:47
  FF3C::/32 Up: 00:50:47
  FF3D::/32 Up: 00:50:47
  FF3E::/32 Up: 00:50:47
  FF3F::/32 Up: 00:50:47
BSR SM RP: FC00:1:0:6::6 Exp: 00:01:46 Learnt from : FC00:1:0:4::4
  FF00::/8 Up: 00:09:45
```

Now make sure R5 has the (*,G) state for the multicast group joined locally:

```
Rack1R5#show ipv6 mroute
Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group,
       C - Connected, L - Local, I - Received Source Specific Host
Report,
       P - Pruned, R - RP-bit set, F - Register flag, T - SPT-bit set,
       J - Join SPT
Timers: Uptime/Expires
Interface state: Interface, State

(*, FF08::8), 00:53:43/never, RP FC00:1:0:6::6, flags: SCLJ
  Incoming interface: Serial0/1/0
  RPF nbr: FE80::215:62FF:FE41:FD9C
  Immediate Outgoing interface list:
    FastEthernet0/0, Forward, 00:53:43/never
```

Ping the group from R1 via its Ethernet interface.

```
Rack1R1#ping ff08::8 repeat 10000
Output Interface: FastEthernet0/0
Type escape sequence to abort.
Sending 10000, 100-byte ICMP Echos to FF08::8, timeout is 2 seconds:
Packet sent with a source address of 2001:1:0:146:213:80FF:FEAD:7A80

Reply to request 0 received from FC00:1:0:58::5, 36 ms
Reply to request 1 received from FC00:1:0:58::5, 16 ms
Reply to request 2 received from FC00:1:0:58::5, 36 ms
Reply to request 3 received from FC00:1:0:58::5, 16 ms
Reply to request 4 received from FC00:1:0:58::5, 36 ms
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/28/36 ms
5 multicast replies and 0 errors.
```

Copyright © 2010 Internetwork Expert

Check the multicast routing states on R4 and R5 and notice the (S,G) entries:

```
Rack1R4#show ipv6 mroute
Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group,
       C - Connected, L - Local, I - Received Source Specific Host
Report,
       P - Pruned, R - RP-bit set, F - Register flag, T - SPT-bit set,
       J - Join SPT
Timers: Uptime/Expires
Interface state: Interface, State

(*, FF08::8), 00:16:14/00:03:13, RP FC00:1:0:6::6, flags: S
  Incoming interface: FastEthernet0/1
  RPF nbr: FE80::21A:2FFF:FE78:4678
  Immediate Outgoing interface list:
    Serial0/1/0, Forward, 00:16:14/00:03:13

(2001:1:0:146:213:80FF:FEAD:7A80, FF08::8), 00:02:13/00:01:16, flags:
ST
  Incoming interface: FastEthernet0/1
  RPF nbr: FE80::213:80FF:FEAD:7A80
  Immediate Outgoing interface list:
    Serial0/1/0, Forward, 00:02:13/00:03:13

Rack1R5#show ipv6 mroute
Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group,
       C - Connected, L - Local, I - Received Source Specific Host
Report,
       P - Pruned, R - RP-bit set, F - Register flag, T - SPT-bit set,
       J - Join SPT
Timers: Uptime/Expires
Interface state: Interface, State

(*, FF08::8), 00:56:34/never, RP FC00:1:0:6::6, flags: SCLJ
  Incoming interface: Serial0/1/0
  RPF nbr: FE80::215:62FF:FE41:FD9C
  Immediate Outgoing interface list:
    FastEthernet0/0, Forward, 00:56:34/never

(2001:1:0:146:213:80FF:FEAD:7A80, FF08::8), 00:02:18/00:01:10, flags:
SJT
  Incoming interface: Serial0/1/0
  RPF nbr: FE80::215:62FF:FE41:FD9C
  Inherited Outgoing interface list:
    FastEthernet0/0, Forward, 00:56:34/never
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

9.27 IPv6 Embedded RP

- Configure R6 to stop announcing itself as an RP.
- Configure R5 to join the group FF76:0640:2001:CC1E::8 on its VLAN 58 interface.
- Make sure R1 can still ping this group without any router in the domain learning the RP information via the BSR.

Configuration

Note

There is one feature that is available in IPv4 multicasting that is not available in IPv6 at this time. Auto-RP is a technology where routers can be set up as candidate RP's, for all or selected multicast groups. They advertise themselves via multicast to 224.0.1.39. Auto-RP is not currently available in IPv6 Multicasting. Additionally, there is nothing like Multicast Source Discovery Protocol (MSDP) in IPv6 Multicasting either.

IPv6 Embedded RP is one solution we can use when we are required to support a PIM SM model across multiple domains. Earlier we discussed the Flags bits in an IPv6 address. We explained that under normal situations the first 3 Flags bits are always set to 000. When we use Embedded RP this situation changes, and the second, third and forth bits are set to 1s. We explained the function of the forth or transient bit earlier, now we are looking at the second and third bits. The second bit is the R bit and when it is set to a value of one, it indicates that the 4-bit RP interface-ID is actually embedded in the group address. The third bit is the P bit when it is set to 1 it indicates that the 64-bit RP prefix is now embedded in the multicast address. Now with the P, R and T bits set to one, instead of relying on the BSR for information propagation, senders and receivers agree to embed the RP IPv6 address into the multicast IPv6 address itself. This address takes the form:

FF7x:0yLL:PPPP:PPPP:PPPP:PPPP:<32-bit-group-id>

Where FF7 is the hexadecimal value of “1111 1111 0111”, and x is the 4-bit scope, y is the 4-bit RP interface-id, LL is an 8-bit prefix length, and PPPP:PPPP:PPPP:PPPP is the 64-bit RP prefix. The actual RP address is constructed by taking the prefix, masking it with LL ones bits and appending the RP Interface-ID like so:

PPPP:PPPP:PPPP:PPPP::y/LL

Copyright © 2010 Internetwork Expert

The interface ID cannot be zero, so it is any value from 0 to F (remember all numbers are in hex!). The router that is supposed to be the RP with this address must have a Loopback interface with this address advertised into the routing protocol so all routers in the multicast domain can reach it.

In our case, we've picked up the group address FF76:0640:2001:CC1E::8 which corresponds to the RP address 2001:CC1E::6/64 which we advertise from R6.

```
R5:
interface FastEthernet 0/0
 ipv6 mld join-group ff76:0640:2001:CC1E::8

R6:
no ipv6 pim bsr candidate rp FC00:1:0:6::6 priority 100
!
interface Loopback300
 ipv6 address 2001:CC1E::6/128
 ipv6 eigrp 100
 ipv6 rip RIPNG enable
```

Verification

Note

Check that R4 and R5 learn the RP-information embedded in the group address.

```
Rack1R5#show ipv6 pim range-list
Static SSM Exp: never Learnt from : ::
  FF33::/32 Up: 01:45:57
  FF34::/32 Up: 01:45:57
  FF35::/32 Up: 01:45:57
  FF36::/32 Up: 01:45:57
  FF37::/32 Up: 01:45:57
  FF38::/32 Up: 01:45:57
  FF39::/32 Up: 01:45:57
  FF3A::/32 Up: 01:45:57
  FF3B::/32 Up: 01:45:57
  FF3C::/32 Up: 01:45:57
  FF3D::/32 Up: 01:45:57
  FF3E::/32 Up: 01:45:57
  FF3F::/32 Up: 01:45:57
Embedded SM RP: 2001:CC1E::6 Exp: never Learnt from : ::
  FF76:640:2001:CC1E::/96 Up: 00:02:53
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ipv6 pim range-list
Static SSM Exp: never Learnt from : ::
  FF33::/32 Up: 01:48:18
  FF34::/32 Up: 01:48:18
  FF35::/32 Up: 01:48:18
  FF36::/32 Up: 01:48:18
  FF37::/32 Up: 01:48:18
  FF38::/32 Up: 01:48:18
  FF39::/32 Up: 01:48:18
  FF3A::/32 Up: 01:48:18
  FF3B::/32 Up: 01:48:18
  FF3C::/32 Up: 01:48:18
  FF3D::/32 Up: 01:48:18
  FF3E::/32 Up: 01:48:18
  FF3F::/32 Up: 01:48:18
Embedded SM RP: 2001:CC1E::6 Exp: never Learnt from : ::
  FF76:640:2001:CC1E::/96 Up: 00:05:09
```

Now ping the group from R1 and ensure you get a response:

```
Rack1R1#ping ipv6 FF76:640:2001:CC1E::8 repeat 1000
Output Interface: FastEthernet0/0
Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to FF76:640:2001:CC1E::8, timeout is
2 seconds:
Packet sent with a source address of 2001:1:0:146:213:80FF:FEAD:7A80

Reply to request 0 received from FC00:1:0:58::5, 60 ms
Reply to request 1 received from FC00:1:0:58::5, 36 ms
Reply to request 2 received from FC00:1:0:58::5, 16 ms
Reply to request 3 received from FC00:1:0:58::5, 48 ms
Success rate is 100 percent (4/4), round-trip min/avg/max = 16/40/60 ms
```

Look at the multicast routing table of R4 and notice the (*,G) state created for the RP 2001:CC1E::6. There is also an (S,G) state for the shortest path tree.

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ipv6 mroute
Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group,
       C - Connected, L - Local, I - Received Source Specific Host
Report,
       P - Pruned, R - RP-bit set, F - Register flag, T - SPT-bit set,
       J - Join SPT
Timers: Uptime/Expires
Interface state: Interface, State

(*, FF76:640:2001:CC1E::8), 00:12:11/00:03:17, RP 2001:CC1E::6, flags:
S
  Incoming interface: FastEthernet0/1
  RPF nbr: FE80::21A:2FFF:FE78:4678
  Immediate Outgoing interface list:
    Serial0/1/0, Forward, 00:12:11/00:03:17

(2001:1:0:146:213:80FF:FEAD:7A80, FF76:640:2001:CC1E::8),
00:01:36/00:01:53, flags: ST
  Incoming interface: FastEthernet0/1
  RPF nbr: FE80::213:80FF:FEAD:7A80
  Immediate Outgoing interface list:
    Serial0/1/0, Forward, 00:01:36/00:02:57
```

You may observe a similar output on R5.

Copyright © 2010 Internetwork Expert

9.28 IPv6 SSM

- Configure R5 to join the group FF36::8 but only accept the multicast streams sourced from R6's IPv6 address on VLAN 146.

Configuration

Note

SSM or Source Specific Multicast is the simplest form of multicasting. The receiver specifies the list of sources it will accept multicast flows from as well as the group address in an MLD Report message (alternatively, there could be an "exclude" report, listing the sources the host does not want to listen to, but this requires source discovery by using an RP and does not work with SSM). The multicast router builds shortest-path trees towards the source(s) specified in the report and the senders may start transmitting. There is no need for RPs with SSM. In fact, groups within the IPv6 SSM range will never have an RP mapped to them. The special SSM group range is FF3x::/96. This allows for 2^{32} multicast groups per sender. You may filter the sources that a listener may join using the MLD configuration command **`ipv6 mld access-group`**, where an access-list specifies the sources and the group ranges.

```
R5:
interface FastEthernet 0/0
 ipv6 mld join-group ff36::8 2001:1:0:146:21A:2FFF:FE78:4678
```

Verification

Note

Check the multicast routing table of R5 and R4 and confirm there are (S,G) entries installed in each.

```
Rack1R5#show ipv6 mroute
Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group,
       C - Connected, L - Local, I - Received Source Specific Host
Report,
       P - Pruned, R - RP-bit set, F - Register flag, T - SPT-bit set,
       J - Join SPT
Timers: Uptime/Expires
Interface state: Interface, State
```

Copyright © 2010 Internetwork Expert

```
(*, FF76:640:2001:CC1E::8), 01:24:31/never, RP 2001:CC1E::6, flags:
SCLJ
  Incoming interface: Serial0/1/0
  RPF nbr: FE80::215:62FF:FE41:FD9C
  Immediate Outgoing interface list:
    FastEthernet0/0, Forward, 01:24:31/never

(2001:1:0:146:21A:2FFF:FE78:4678, FF36::8), 00:00:03/never, flags: sLTI
  Incoming interface: Serial0/1/0
  RPF nbr: FE80::215:62FF:FE41:FD9C
  Immediate Outgoing interface list:
    FastEthernet0/0, Forward, 00:00:03/never

(*, FF08::8), 03:06:48/never, RP ::, flags: SCLJ
  Incoming interface: Null
  RPF nbr: ::
  Immediate Outgoing interface list:
    FastEthernet0/0, Forward, 03:06:48/never

Rack1R4#show ipv6 mroute
Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group,
       C - Connected, L - Local, I - Received Source Specific Host
Report,
       P - Pruned, R - RP-bit set, F - Register flag, T - SPT-bit set,
       J - Join SPT
Timers: Uptime/Expires
Interface state: Interface, State

(*, FF76:640:2001:CC1E::8), 01:24:23/00:03:05, RP 2001:CC1E::6, flags:
S
  Incoming interface: FastEthernet0/1
  RPF nbr: FE80::21A:2FFF:FE78:4678
  Immediate Outgoing interface list:
    Serial0/1/0, Forward, 01:24:23/00:03:05

(2001:1:0:146:21A:2FFF:FE78:4678, FF36::8), 00:02:21/00:03:05, flags:
sT
  Incoming interface: FastEthernet0/1
  RPF nbr: FE80::21A:2FFF:FE78:4678
  Immediate Outgoing interface list:
    Serial0/1/0, Forward, 00:02:21/00:03:05
```

Copyright © 2010 Internetwork Expert

Now generate multicast packets from R6 and confirm that you are getting the responses:

```
Rack1R6#ping ipv6 FF36::8 repeat 1000
Output Interface: FastEthernet0/0.146
Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to FF36::8, timeout is 2 seconds:
Packet sent with a source address of 2001:1:0:146:21A:2FFF:FE78:4678

Reply to request 0 received from FC00:1:0:58::5, 24 ms
Reply to request 1 received from FC00:1:0:58::5, 36 ms
Reply to request 2 received from FC00:1:0:58::5, 12 ms
Reply to request 3 received from FC00:1:0:58::5, 36 ms
Reply to request 4 received from FC00:1:0:58::5, 12 ms
Reply to request 5 received from FC00:1:0:58::5, 36 ms
Reply to request 6 received from FC00:1:0:58::5, 12 ms
Reply to request 7 received from FC00:1:0:58::5, 36 ms
Success rate is 100 percent (8/8), round-trip min/avg/max = 12/25/36 ms
8 multicast replies and 0 errors.
Rack1R6#
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

> **Note**
>
> Clear all router configurations and load the *IPv6* initial configurations.

9.29 IPv6 Tunneling

- Create new Loopback interfaces on R2, R5, and R6 with the IP addresses 2001:X:0:Y::Y/64 where Y is the router number and X is your rack number in decimal notation (e.g. 10 for rack 10, 11 for rack 11 etc.)
- Create tunnels between R5 & R6 and R2 & R6. Use the IPv6 addresses 2001:X:0:56::Y/64 for the tunnel between R5 and R6 and the IPv6 address 2001:X:0:26::Y/64 for the tunnel between R2 and R6.
- The tunnel connecting R2 and R6 should use encapsulation suitable for transporting different Layer 3 protocols.
- The tunnel connecting R5 and R6 should use the encapsulation specifically designed for transporting IPv6 over IPv4.
- Use static routing to obtain connectivity.

Configuration

```
R2:
ipv6 unicast-routing
!
interface Loopback100
 ipv6 address 2001:1:0:2::2/64
!
interface tunnel 26
 tunnel source Loopback0
 tunnel destination 150.1.6.6
 ipv6 address 2001:1:0:26::2/64
!
ipv6 route ::/0 Tunnel 26

R5:
ipv6 unicast-routing
!
interface Loopback100
 ipv6 address 2001:1:0:5::5/64
!
interface tunnel 56
 tunnel source Loopback0
 tunnel destination 150.1.6.6
 ipv6 address 2001:1:0:56::5/64
 tunnel mode ipv6ip
!
ipv6 route ::/0 Tunnel 56
```

Copyright © 2010 Internetwork Expert

```
R6:
ipv6 unicast-routing
!
interface Loopback100
 ipv6 address 2001:1:0:6::6/64
!
interface tunnel 26
 tunnel source Loopback0
 tunnel destination 150.1.2.2
 ipv6 address 2001:1:0:26::6/64
!
interface tunnel 56
 tunnel source Loopback0
 tunnel destination 150.1.5.5
 ipv6 address 2001:1:0:56::6/64
 tunnel mode ipv6ip
!
ipv6 route 2001:1:0:5::/64 Tunnel 56
ipv6 route 2001:1:0:2::/64 Tunnel 26
```

Verification

> **Note**
>
> IPv6 can be transported across IPv4 clouds using various tunneling techniques. The most common are static point-to-point tunnels, using either GRE (generic routing encapsulation), or IPv6 in IPv4 encapsulation (a special protocol type designed to carry only IPv6 packets). Designed to carry a multi-protocol payload, GRE has a slightly larger overhead than IPv6 in IPv4 encapsulation.

```
Rack1R6#show interfaces tunnel 26
Tunnel26 is up, line protocol is up
  Hardware is Tunnel
  MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 150.1.6.6 (Loopback0), destination 150.1.2.2
  Tunnel protocol/transport GRE/IP
    Key disabled, sequencing disabled
    Checksumming of packets disabled
  Tunnel TTL 255
<snip>
```

```
Rack1R6#show interfaces tunnel 56
Tunnel56 is up, line protocol is up
  Hardware is Tunnel
  MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 150.1.6.6 (Loopback0), destination 150.1.5.5
  Tunnel protocol/transport IPv6/IP
  Tunnel TTL 255
<snip>

Rack1R2#ping 2001:1:0:5::5 source loopback 100

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:1:0:5::5, timeout is 2 seconds:
Packet sent with a source address of 2001:1:0:2::2
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 156/167/205
ms
```

Pitfall

IPv6 IP tunnels use IP protocol number 41 for transport. This protocol number does not have a keyword shortcut in extended IP access-lists in IOS. Therefore to permit or deny an IPv6IP tunnel, the syntax **`access-list 100 [permit|deny] 41 any any`** is required.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

9.30 Automatic 6to4 Tunneling

- Using the IPv4 Loopback 0 interfaces create automatic 6to4 tunnels connecting R3, R4, and R5.
- Create additional Loopback interfaces on every router with the subnet number 0 and the prefix length of 64 under the respective 6to4 /48 prefix.
- Use static routing to obtain connectivity between the newly allocated subnets.

Configuration

```
R3:
interface Tunnel 345
 tunnel source Loopback0
 tunnel mode ipv6ip 6to4
 ipv6 address 2002:9601:303::3/64
!
ipv6 route 2002::/16 Tunnel 345
!
interface Loopback200
 ipv6 address 2002:9601:303:1::3/64

R4:
interface Tunnel 345
 tunnel source Loopback0
 tunnel mode ipv6ip 6to4
 ipv6 address 2002:9601:404::5/64
!
ipv6 route 2002::/16 Tunnel 345
!
interface Loopback200
 ipv6 address 2002:9601:404:1::4/64

R5:
interface Tunnel 345
 tunnel source Loopback0
 tunnel mode ipv6ip 6to4
 ipv6 address 2002:9601:505::5/64
!
ipv6 route 2002::/16 Tunnel 345
!
interface Loopback200
 ipv6 address 2002:9601:505:1::5/64
```

Copyright © 2010 Internetwork Expert

Verification

Note

Automatic 6to4 tunnels are multipoint by design. The idea is to allow automatic routing across an IPv4 cloud based on a part of the destinations IPv6 address. Specifically, the format of the 6to4 IPv6 address is as follows:

2002 (16 bits):IPv4 address (32 bits):Subnet ID(16 bits):Interface ID (64 bits)

When a packet is routed across the 6to4 tunnel, the router extracts the IPv4 address embedded in the IPv6 address and uses it to build the IPv4 destination address of the tunnel header. The receiving router strips the header, extracts the IPv6 packet, and routes it based on the IPv6 routing table.

As you can imagine, 6to4 subnets have some addressing restrictions. First, you need to use the 16-bit prefix 2002, as it is the common reservation for all 6to4 deployments. Second, you need to select the public IPv4 address used to create the /48 prefix. It is common to pick any interface of the border router and then allocate the /64 subnets to other devices on the network, as long as the address is publicly routable.

6to4 tunnels are a transition mechanism for hosts that do not have native IPv6 connectivity, allowing them to reach other nodes that have full connectivity to the IPv6 Internet. Due to the multipoint nature of the 6to4 tunnels, only static routing is possible with this technology. However, it is common to simply route the whole 2002::/16 prefix to the 6to4 tunnel.

In our case, the use of Loopback 0 subnets results in the following IPv6 6to4 prefixes:

R3: 150.1.3.3 = 2002:9601:303::/48
R4: 150.1.4.4 = 2002:9601:404::/48
R5: 150.1.5.5 = 2002:9601:505::/48

“9601” in hex corresponds to “150.1” in decimal, and on R3 “303” corresponds to “3.3”.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show interfaces tunnel 345
Tunnel345 is up, line protocol is up
  Hardware is Tunnel
  MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 150.1.5.5 (Loopback0)
  Tunnel protocol/transport IPv6 6to4

  Fast tunneling enabled
  Tunnel transmit bandwidth 8000 (kbps)
  Tunnel receive bandwidth 8000 (kbps)
  Last input never, output 00:00:13, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/0 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     11 packets output, 1176 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out

Rack1R5#ping ipv6 2002:9601:404:1::4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2002:9601:404:1::4, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 44/45/52 ms

Rack1R5#ping ipv6 2002:9601:303:1::3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2002:9601:303:1::3, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/70/72 ms
```

Copyright © 2010 Internetwork Expert

9.31 ISATAP Tunneling

- Remove the 6to4 tunnels connecting R3, R4, and R5.
- Using their Loopback 0 IPv4 addresses connect R1, SW1, and SW2 via ISATAP tunnels.
- Use the /64 IPv6 prefix 2001:1:0:345::/64 to allocate IPv6 addresses to the tunnel endpoints.
- Create additional IPv6 Loopback interfaces on all three routers with the IPv6 addresses 2001:1:0:Y::Y/64, and use static routing to obtain full connectivity.

Configuration

```
R3:
no interface Tunnel345
!
interface Tunnel345
 ipv6 address 2001:1:0:345::/64 eui-64
 tunnel source Loopback0
 tunnel mode ipv6ip isatap
!
interface Loopback100
 ipv6 address 2001:1:0:3::3/64
!
ipv6 route 2001:1:0:4::/64 2001:1:0:345:0:5efe:9601:404
ipv6 route 2001:1:0:5::/64 2001:1:0:345:0:5efe:9601:505

R4:
no interface Tunnel345
!
interface Tunnel345
 ipv6 address 2001:1:0:345::/64 eui-64
 tunnel source Loopback0
 tunnel mode ipv6ip isatap
!
interface Loopback100
 ipv6 address 2001:1:0:4::4/64
!
ipv6 route 2001:1:0:3::/64 2001:1:0:345:0:5efe:9601:303
ipv6 route 2001:1:0:5::/64 2001:1:0:345:0:5efe:9601:505
```

Copyright © 2010 Internetwork Expert

```
R5:
no interface Tunnel345
!
interface Tunnel345
 ipv6 address 2001:1:0:345::/64 eui-64
 tunnel source Loopback0
 tunnel mode ipv6ip isatap
!
interface Loopback100
 ipv6 address 2001:1:0:5::5/64
!
ipv6 route 2001:1:0:4::/64 2001:1:0:345:0:5efe:9601:404
ipv6 route 2001:1:0:3::/64 2001:1:0:345:0:5efe:9601:303
```

Verification

Note

ISATAP (Intra-Site Automatic Tunnel Addressing Protocol) is a technology that contrasts 6to4 automatic tunnels. While 6to4 automatically generates a /48 prefix out of an IPv4 address, ISATAP uses the IPv4 domain as a "multi-access" media with IPv4 addresses being the rough equivalent of Ethernet MAC addresses.

Specifically, ISATAP constructs the interface identifier (last 64 bits) of the IPv6 address based on the IPv4 address of a host using the EUI-64 rules. Thus, given that you already have a /64 IPv6 prefix, you can allocate IPv6 addresses to tunnel endpoints performing simple manipulations over the IPv4 endpoint addresses. Specifically, the interface ID is constructed as follows:

EUI-64 = 0000 (16 bits) + 5EFE (16 bits) + IPv4 Address (32 bits).

Therefore, if you select the prefix 2001:1:0:345::/64, then R3 will have the IP address:

2001:1:0:345:0:5efe:9601:0303/64

Where 9601 is for 150.1 and 0303 is for last two octets of the IP address. When configuring an IOS router for ISATAP, you can automatically generate interface ID's with the **`eui-64`** keyword.

Unlike the 6to4 tunnels, ISATAP tunnels cannot automatically extract the destination. Therefore, you must use static routes that point to the exact IPv6 endpoint on the other end of the tunnel.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show interfaces tunnel 345
Tunnel345 is up, line protocol is up
  Hardware is Tunnel
  MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 150.1.5.5 (Loopback0), destination UNKNOWN
  Tunnel protocol/transport IPv6 ISATAP

  <snip>
```

```
Rack1R5#show ipv6 interface tunnel 345
Tunnel345 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::5EFE:9601:505
  Global unicast address(es):
    2001:1:0:345:0:5EFE:9601:505, subnet is 2001:1:0:345::/64 [EUI]
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF01:505
  MTU is 1480 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ND DAD is not supported
  ND reachable time is 30000 milliseconds
  Hosts use stateless autoconfig for addresses.
```

```
Rack1R5#ping 2001:1:0:345:0:5efe:9601:404

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:1:0:345:0:5EFE:9601:404, timeout
is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 44/44/45 ms

Rack1R5#ping 2001:1:0:345:0:5efe:9601:303

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:1:0:345:0:5EFE:9601:303, timeout
is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/69/73 ms

Rack1R5#ping 2001:1:0:4::4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:1:0:4::4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 44/44/45 ms

Rack1R5#ping 2001:1:0:3::3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:1:0:3::3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/69/72 ms
```

Copyright © 2010 Internetwork Expert

Quality of Service Solutions

> **Note**
>
> Load the *QoS* initial configurations prior to starting.

10.1 Hold-Queue and Tx-Ring

- Configure R1's connection to VLAN 146 to have an input software queue length of 10 packets, and an output software queue length of 30 packets.
- Set the output hardware queue size to 15 packets.

Configuration

```
R1:
interface FastEthernet0/0
 tx-ring-limit 15
 hold-queue 10 in
 hold-queue 30 out
```

Verification

```
Rack1R1#show interfaces fastEthernet 0/0 | include queue
  Input queue: 0/10/0/0 (size/max/drops/flushes); Total output drops: 0
  Output queue: 0/30 (size/max)

Rack1R1#show controllers fastEthernet 0/0 | include tx
 rx ring entries=64, tx ring entries=128
 txring=0x7DC5800, txr shadow=0x84C0A6D4, tx_head=12, tx_tail=12, tx_count=0
 tx_one_col_err=0, tx_more_col_err=0, tx_no_enp=0, tx_deferred_err=0
 tx_underrun_err=0, tx_late_collision_err=0, tx_loss_carrier_err=6
 tx_exc_collision_err=0, tx_buff_err=0, fatal_tx_err=0
 tx_limited=0(15)
```

> **Note**
>
> Network interfaces on a router work asynchronously. This means the interface driver responds to hardware interrupts, notifies the switching process, and queues incoming packets if the central processor is not yet ready to start working on them. The same applies to outgoing packets. The IOS prepares the packet, puts it into a buffer, and notifies the interface driver of the new packet.

Copyright © 2010 Internetwork Expert

On the input direction if the processor is not too busy and the packet rate is not too high, the system will never actually use the input queue. Similarly, if the outgoing interface is not congested, the interface driver will work fast enough to stop the outgoing queue from growing.
To fully understand an end-to-end QoS model it's important to understand the key differences and similarities between the input and output queues. Both queue types consume memory chunks from buffer pools – interface buffer pools and public shared buffer pools. The buffer management scheme can be quite complicated, especially with particle buffer pools, but the idea remains the same. Queues consume I/O memory available via system buffers, which is a configurable parameter.

For input queueing there is just one queue per interface, which is always First-In-First-Out (FIFO), and has a size of 75 packets by default. The Interface driver assigns incoming packets directly into the incoming FIFO queue if the central processor has no time to switch them quickly (e.g. if sudden packet burst occurs). For more information about input queue management see the section dedicated to SPD in this document.

For output queueing there are two output queues by default. This value can be increased for interfaces that support Virtual Circuits (VCs), such as Frame Relay or ATM, when per-VC queuing is enabled. The first output queue is the "software queue", and defines the per-interface logical queuing strategy. This software queue can be a simple FIFO (hold-queue), or where more advanced "fancy queueing" methods are defined, such as Weighted Fair Queueing (WFQ) or Class-Based Weighted Fair Queueing (CBWFQ). Note that the FIFO hold-queue is the default and simplest queuing strategy on high-speed interfaces.

The second output queue sits right after the software queue, and is the hardware queue or "transmit ring" (TX-ring). The interface driver actually works with this memory space directly when looking for packets to send. This queue is typically smaller than the software queue, and is always FIFO.

Note that the software output queue only starts to fill up when the tx-ring is full. This is due to the fact that traffic does not need to be software queued unless the physical hardware queue is busy sending a packet. Therefore the tx-ring length defines how fast the system switches to a logical queuing strategy for interface scheduling. Triggering complicated software queues such as CBWFQ too often may degrade the overall router performance, such as when CBWFQ needs to make a full sorting run across all queues. This is why tuning the tx-ring and output queue size can be a complicated practice, especially when working with high delay and jitter sensitive traffic such as voice.

Copyright © 2010 Internetwork Expert

Note that for the software output queue is most commonly referred to as just the "output queue", while the hardware output queue is referred to as just the "transmit ring" (tx-ring). For the sake of differentiating the two these naming conventions are adhered to throughout the scope of this document.

To test the above configuration simulate a packet storm towards R1 and see how the input queue starts to fill up.

```
Rack1R6#ping 155.1.146.1 repeat 1000000 timeout 0 size 1000

Type escape sequence to abort.
Sending 1000000, 1000-byte ICMP Echos to 155.1.146.1, timeout is 0
seconds:
 ............................................................
 ...........................................................!.........
 ............................................................
 ............................................................
<snip>

Rack1R1#show interfaces fastEthernet 0/0 | inc queue
  Input queue: 11/10/133/0 (size/max/drops/flushes); Total output drops: 0
  Output queue: 0/30 (size/max)
```

Copyright © 2010 Internetwork Expert

10.2 Weighted Fair Queuing (WFQ)

- Configure the point-to-point Serial link between R4 and R5 with an interface clock rate and interface bandwidth of 128Kbps.
- Set the output hold-queue size to 256.
- Configure WFQ on the links with a length of 16 for the congestive discard threshold, 128 for the number of conversations, and 8 for RSVP reservable queues.
- Set the tx-ring size to the minimal value to engage the WFQ as fast as possible.
- Normalize the packet flows for the link by ajdusting the MTU so that each IP packet takes no more than 10ms to be sent.

Configuration

```
R4:
interface Serial0/1/0
 clock rate 128000
 bandwidth 128
 tx-ring-limit 1
 fair-queue 16 128 8
 hold-queue 256 out
 ip mtu 156

R5:
interface Serial0/1/0
clock rate 128000
 bandwidth 128
 tx-ring-limit 1
 fair-queue 16 128 8
 hold-queue 256 out
 ip mtu 156
```

Verification

Note

WFQ uses an intelligent congestion management solution that provides “fair” sharing of the interface bandwidth between multiple traffic *flows*. A traffic “flow” (or *conversation*) is a unidirectional sequence of packets, defined based on the protocol type, the source/destination IP addresses, the source/destination ports numbers (when available), and partially on the IPv4 ToS byte value. For example, an HTTP file transfer between two hosts represents one packet flow, while ICMP packets sent from one host to another represents a second.

Copyright © 2010 Internetwork Expert

The term "fair" in WFQ refers to the *max-min* fairness. The WFQ calculation procedure is as follows.

First, divide the interface bandwidth equally between all flows. For example if there are 10 flows, and 100Kbps of bandwidth, each flow gets 10Kbps. If a flow's demand is less than the "equal" share, e.g. a flow only needs 5Kbps instead of 10Kbps, then allocate the unneeded bandwidth equally among the remaining flows. If there are flows that demand more than their equal share, e.g. the equal share is 10Kbps, but two flows demand 25Kbps and 20Kbps respectively, they will each get the equal, maximum, possible shares (e.g 19Kbps and 19Kbps) but only if there are flows which demand less than the equal share.

In this context *max-min* means that all flows first get the bare minimum bandwidth, but the procedure tries to maximize each flow's share if possible. The concept of basic WFQ is very important to understand since many other congestion management techniques, like Round Robin scheduling, utilize it. To reiterate, the key thing to remember about WFQ is that *max-min* scheduling allows the sharing of a flow's unclaimed bandwidth between other flows.

An individual flow in the queue may be more or less demanding than other flows. The "demanding" flow will generate a higher bit-rate, either due to larger packet sizes or a higher packet per second rate. For example compare a bulk FTP file transfer against telnet sessions or VoIP RTP streams. Within the context of flow-based sharing, it is also important to understand that a single host may generate *multiple* flows, P2P file-sharing applications or download "accelerators" fall into this category. In this scenario, this particular host can obtain an "unfair" share of bandwidth compared to other hosts. Unfortunately, this is a limitation of a classification scheme that's based solely only on simple flows, such as WFQ, which has no notion of "flow mask".

To enhance its scheduling logic, WFQ may assign a *weight* value to a flow. The weight affects the minimum guaranteed share of bandwidth available to a flow. If there are *N* flows with weights w1, w2 ... w*N*, then a particular flow "*K*" will get the minimum share of bandwidth where s*K*=(w1+w2+… w*K*+… w*N*)/w*K* – inversely proportional to its weight. The shares are relative, in the sense that the scheduler divides the available bandwidth in proportions: s1:s2:…s*N*.

The IOS implementation of WFQ assigns weights automatically based on the IP Precedence (IPP) value in the packet's IP header. The formula is Weight=32384/(IPP+1), where IPP is the IP Precedence of the flow.

Copyright © 2010 Internetwork Expert

To understand the effect of the weight settings, imagine four flows with different IP Precedence values of zero, one, one and three.

Step 1:
Using the above formula for weight, we obtain:

Weight(1) = 32384/(0+1) = 32384
Weight(2) = 32384/(1+1) = 16192
Weight(3) = 32384/(1+1) = 16192
Weight(4) = 32384/(3+1) = 8096

Step 2:

Compute the sum of all weights:
Sum(Weight(i),1…4) = 32384+16192*2+8096 = 72864

Step 3:
Compute the shares:

Share(1) = 72864/Weight(1) = 72864/32384 = 2.25
Share(2) = 72864/Weight(2) = 72864/16192 = 4.5
Share(3) = 72864/Weight(3) = 72864/16192 = 4.5
Share(4) = 72864/Weight(4) = 72864/8096 = 9

The proportion is 2.25:4.5:4.5:9 = 1:2:2:4 – this is based on the "shifted" ip precedences: (IPP(i) + 1)

Note that the numerator "32384" is an arbitrary value in these computations, meaning that you can use any value. The important thing is the proportion of "shifted" IP precedences.

For the bandwidth sharing we have been describing to function correctly, all flows must be adaptive. Adaptive flows are those that respond to packet drops by slowing their sending rate. Non-adaptive, aggressive flows may defeat the bandwidth sharing logic of WFQ by claiming all available *buffer space* and starving other flows because when implementing WFQ all flows use shared buffer space (more on this later).

Copyright © 2010 Internetwork Expert

Specifically WFQ implements its fair sharing logic as follows. First, the scheduler creates a group of flow queues for a particular interface, e.g. 256 "conversations", based on a manual setting or an automatic calculation derived from the interface bandwidth. When a new packet arrives for output scheduling, the scheduler applies a special *hash* function to the packet's source/destination IP/Port values to yield the flow queue number. This is why the number of queues is always a power of 2, because the hash output value is somewhere between 0 and 2^N. This procedure also means that multiple flows may share the same queue, called "hash collision", when the number of flows is large.

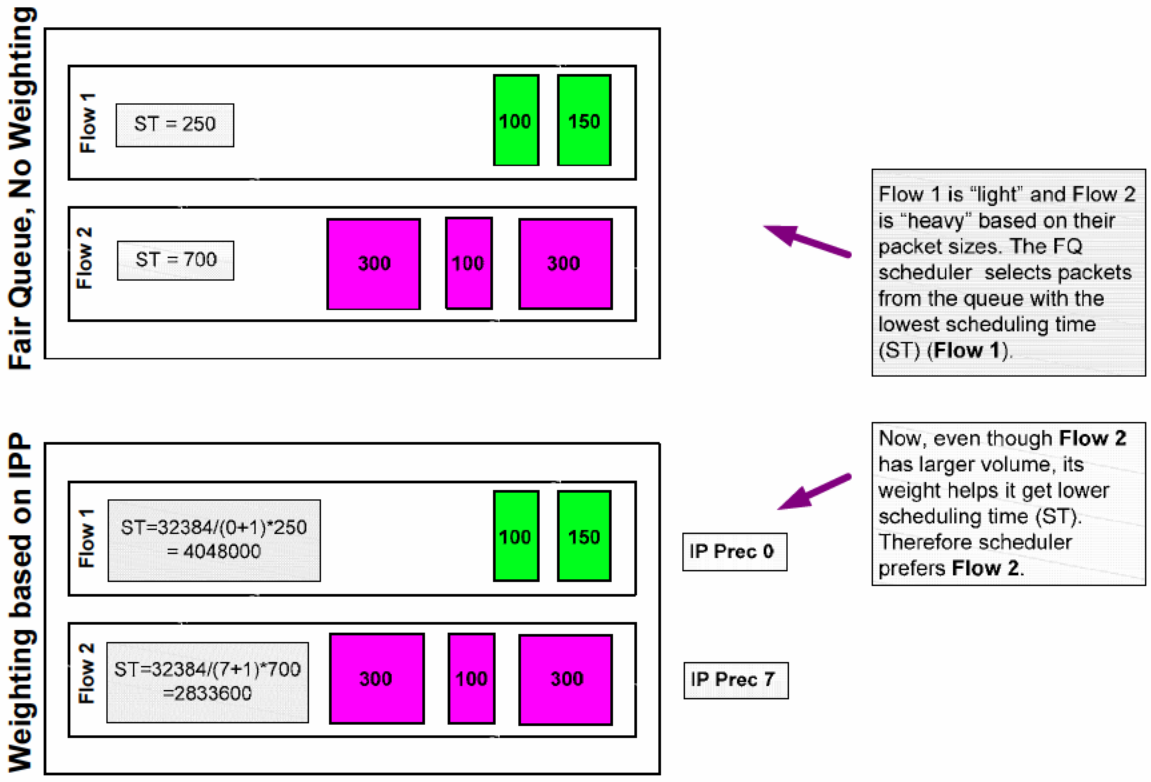
Each flow queue has a special "*virtual scheduling time*" assigned – the amount of "time" that it would take to serialize the packets in the queue across the output interface. This "virtual time" is actually the total size of all packets stored in the flow queue scaled by the flow computational weight.

Now imagine a new packet of size *packet_size* and IP precedence of *packet_ip_precedence* arrives at its respective flow queue (hash queue):

weight = 32384/(packet_ip_precedence +1)
*virtual_scheduling_time = queue_tail_time + packet_size*weight*
queue_tail_time = virtual_scheduling_time

The "*queue_tail_time*" variable stores the previous virtual scheduling time for the flow. Note that the weight is inversely proportional to IP precedence, and thus more important packets have smaller virtual scheduling time value (WFQ thinks that it can serialize them "faster"). It would be more appropriate to call this computational weight the "scaling factor" to avoid confusion with the classic meaning of weight.

Copyright © 2010 Internetwork Expert



Every time WFQ needs to take a packet out of a queue, it sorts all queues based on their virtual scheduling time (sorting is time-consuming). The scheduler selects the flow queue with the *minimum* virtual scheduling time, sends a packet out and then decreases the respective *queue_tail_time* variable by the packet's size. In essence, this procedure tries to pull the short (smaller packets) flow's traffic first.

However, the following question needs to be answered: if a new flow queue has just started, how should we initialize its *queue_tail_time* variable? Should we do it based on just the size of the first packets? That would result in every new flow having a better chance of being scheduled than older ones with larger *queue_tail_times*! To prevent this from happening WFQ uses a special variable called the "*round_number" that is* shared by all queues. The *round_number* is equal to the *virtual_scheduling_time* of the last packet sent down the physical line. When a flow queue is first initialized, its *queue_tail_time* is set equal to the latest *round_number,* thus making it "comparable" with the existing flow queues, and not letting it monopolize the physical line. The idea is to penalize the new flow by the amount of time other flows have already waited so far.

Copyright © 2010 Internetwork Expert

The next important point is the congestive discard threshold (CDT), which is a unique congestion avoidance scheme available with WFQ. First you configure the total buffer space allocated to WFQ using the **`hold-queue <N> out`** command. The WFQ shares this buffer space between all flow queues. Any queue may grow up to the maximum free space, but as soon as its size reaches the CDT, WFQ drops a packet from the flow queue with the "*maximum*" virtual scheduling time (this may be some other queue, not the one that crossed the packet threshold). This way, WFQ penalizes the most aggressive flow and triggers a mechanism similar to random early detection's (RED) prevention of tail-drop. The fair-queue settings command is **`fair-queue <CDT> <N Flow Queues> <N Reservable Queues>.`**

The number of reservable conversations (queues) is the number of flow-queues available to RSVP reservations (if any). These flows have a very small weight value, and thus are preferred above any other flows. In addition to reserved flow queues, there are special "Link Queues". The number of queues is fixed to 8, and they are numbered right after the maximum dynamic queue (e.g. if there are 32 dynamic queues, "Link Queues" start at number 33). WFQ uses these queues to service routing protocol traffic and Layer 2 keepalives – everything that is critical to router operations and management. Each queue has a weight of 1024, which is lower than any dynamic queue can get, so control plane traffic has priority over regular data traffic.

Finally, note that the interface bandwidth setting does not influence the WFQ algorithm directly. It only prompts the WFQ command to pick up optimal parameters matching the interface bandwidth. The bandwidth, however, *is* used for admission control with RSVP flows.

Copyright © 2010 Internetwork Expert

For this particular task, verification can be performed as follows.

```
Rack1R4#show queueing interface serial 0/1/0
Interface Serial0/1/0 queueing strategy: fair
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 2044
  Queueing strategy: weighted fair
  Output queue: 3/256/16/2044 (size/max total/threshold/drops)
     Conversations  2/3/128 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 96 kilobits/sec

Rack1R4#show queueing fair
Current fair queue configuration:

  Interface             Discard    Dynamic  Reserved  Link     Priority
                        threshold  queues   queues    queues   queues
  Serial0/0/0           64          256      0         8        1
  Serial0/1/0           16          128      8         8        1
```

Changing the **ip mtu** for the interface in this task causes fragmentation at layer 3, and normalizes the packet sizes for further testing. With an additional 4 bytes of overhead for the layer 2 HDLC encapsulation, every frame sent out the link has a maximum size of 160 bytes (156 of IP plus 4 bytes of HDLC) . With a physical clocking rate of 128000 bits per second, this means a 160 byte packet will take a maximum time of 10ms to serialize. Adjusting serialization delay through fragmentation is covered in more detail later in this document.

To verify how WFQ shares the interface bandwidth the network is modified as follows. The HTTP service is enabled on R1 and R6 with the root directory to "flash:". R5's Frame Relay interface is shutdown to ensure traffic from VLAN 146 takes the path across the Serial link between R4 and R5. R6 is configured to mark traffic leaving the VLAN146 interface with an IP precedence of 1, and R1 is configured to mark traffic leaving VLAN146 with an IP precedence of 3. A policy-map is configured on R5 to match incoming IP precedence 1 and IP precedence 3 packets, and it is applied inbound to interface Serial0/1 to meter incoming traffic. Finally, the IOS images stored in flash memory on R1 and R6 are downloaded by SW2 and SW4's to "null:" using HTTP. The resulting configuration is as follows.

Copyright © 2010 Internetwork Expert

```
R1:
ip http server
ip http path flash:
ip tcp mss 100
!
policy-map MARK
 class class-default
  set ip precedence 3
!
interface FastEthernet 0/0
 service-policy output MARK

R5:
class-map match-all IP_PREC3
  match ip precedence 3
 class-map match-all IP_PREC1
  match ip precedence 1
!
 policy-map METER
  class IP_PREC1
  class IP_PREC3
!
interface Serial 0/1/0
 service-policy input METER
 load-interval 30
 clock rate 128000
!
interface Serial 0/0/0
 shutdown

R6:
ip http server
ip http path flash:
ip tcp mss 100
!
policy-map MARK
 class class-default
  set ip precedence 1
!
interface FastEthernet 0/0.146
 service-policy output MARK
```

```
Rack1SW2#copy http://admin:cisco@155.1.146.6/ c1841-adventerprisek9-
mz.124-24.T.bin null:

Rack1SW4#copy http://admin:cisco@155.1.146.1/c2600-adventerprisek9-
mz.124-10.bin null:
```

Copyright © 2010 Internetwork Expert

The configuration effectively generates two TCP flows across R4's connection to R5. Since TCP is adaptive, the flows should eventually balance and start sharing bandwidth using their weights: 1+1 and 3+1, which is 2:4. This means file transfers from R1 to SW4 should have twice as much bandwidth as the file transfer from R6 to SW2.

```
Rack1R5#show policy-map int serial 0/1/0
 Serial0/1/0

  Service-policy input: METER

    Class-map: IP_PREC1 (match-all)
      91618 packets, 17299811 bytes
      30 second offered rate 41000 bps
      Match: ip precedence 1

    Class-map: IP_PREC3 (match-all)
      330231 packets, 75523458 bytes
      30 second offered rate 83000 bps
      Match: ip precedence 3

    Class-map: class-default (match-any)
      58353 packets, 8839968 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

Check the flow queues on R4 to see their WFQ parameters. Note that the average packet length is the same, and the inverse weight proportion is 1/8096:1/16192 = 2:1 (the guaranteed shares of bandwidth for IP Precedence 1 and IP Precedence 0 traffic).

```
Rack1R4#show queueing interface serial 0/1/0
<snip>
  (depth/weight/total drops/no-buffer drops/interleaves) 7/16192/0/0/0
  Conversation 20, linktype: ip, length: 580
  source: 155.1.146.6, destination: 155.1.58.8, id: 0x6927, ttl: 254,
  TOS: 32 prot: 6, source port 80, destination port 11009

  (depth/weight/total drops/no-buffer drops/interleaves) 6/8096/0/0/0
  Conversation 26, linktype: ip, length: 580
  source: 155.1.146.1, destination: 155.1.108.10, id: 0x7BCF, ttl: 254,
  TOS: 96 prot: 6, source port 80, destination port 11001
```

Copyright © 2010 Internetwork Expert

As mentioned above, non-adaptive flows, such as UDP/ICMP traffic floods, may oversubscribe the shared WFQ buffer space. This is due to their "greedy" behavior - a single flow tries to monopolize the whole buffers space available to all queues, causing excessive packet drops, and dose not respond to congestive discards.

To see how this behavior can manifest itself originate two ICMP packet floods from R6 and R1 towards R5 using a packet size of 100 and a timeout of 0. The timeout value of zero means that the devices do not wait for an echo-reply before sending its next echo. Additionally configure R5 so that it does not respond to the echos so the return traffic doesn't affect the flow.

```
R5:
access-list 100 deny icmp any host 155.1.45.5
access-list 100 permit ip any any
!
interface Serial 0/1/0
 ip access-group 100 in
 no ip unreachables

Rack1R1#ping 155.1.45.5 repeat 100000000 timeout 0
Rack1R6#ping 155.1.45.5 repeat 100000000 timeout 0

Rack1R4#show queueing interface serial 0/1/0
<snip>
  (depth/weight/total drops/no-buffer drops/interleaves) 43/8096/7289/0/0
  Conversation 30, linktype: ip, length: 104
  source: 155.1.146.1, destination: 155.1.45.5, id: 0x05E1, ttl: 254, prot: 1

  (depth/weight/total drops/no-buffer drops/interleaves) 21/16192/25458/0/0
  Conversation 3, linktype: ip, length: 104
  source: 155.1.146.6, destination: 155.1.45.5, id: 0xC38A, ttl: 254, prot: 1

Rack1R5#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy input: METER

    Class-map: IP_PREC1 (match-all)
      104422 packets, 22233791 bytes
      30 second offered rate 47000 bps
      Match: ip precedence 1

    Class-map: IP_PREC3 (match-all)
      353680 packets, 85227338 bytes
      30 second offered rate 68000 bps
      Match: ip precedence 3
<snip>
```

Note the huge number of drops due to the queue getting full. Also, note that offered rates on R5 are *not* in a 1:2 proportion, even though the weights are set to share the bandwidth in 1:2 ratio and the packet lengths are the same.

Copyright © 2010 Internetwork Expert

10.3 Legacy RTP Reserved Queue

- Reset R4's Serial link to R5 to the default fair-queue values.
- Configure a single command on R4's interface so that 100% of the link bandwidth is reserved for RTP traffic in the UDP port range 16384 – 32767.

Configuration

```
R4:
interface Serial 0/1/0
 no hold-queue out
 fair-queue
 max-reserved-bandwidth 100
 ip rtp reserve 16384 16383 128
```

Verification

Note

Real Time Protocol (RTP) is the UDP based protocol used for audio and video transmissions, such as VoIP. **`ip rtp reserve`** was the first feature aimed at making WFQ aware of voice bearer traffic (RTP packets). By default, WFQ already prefers voice packets due to their small size and (usually) high IP Precedence value of 5. However as competing flows with the same weight appear in the WFQ, voice traffic is no longer given preferrential treatment.

The first solution for this problem was to enable automatic mapping of voice bearer traffic (RTP packets with a UDP port range 16384-32767) to a special flow queue. WFQ uses a Resource Reservation Protocol (RSVP) classifier and special RSVP conversation to queue the matching packets. The WFQ scheduler will treat this queue as a special "RSVP" reserved queue with a very low weight. Thus, only RSVP flows may compete with the special flow for the voice bearer packets. More details on RSVP and WFQ interaction are covered later.

The command format is **`ip rtp reserve`** ***`<Starting UDP Port> <Port Range> <Bandwidth>`***

For example **`ip rtp reserve 16384 16383 128`** means to reserve a special RSVP conversation for UDP traffic flows destined to ports in range 16384 – 32767 (16384+16383) – the standard RTP port range used by Cisco devices. The conversation number for IP RTP Reserve is equal to number_of_dynamic_WFQ_flows + 8 + 1, e.g. the first flow after the "Link Queues".

Copyright © 2010 Internetwork Expert

Configuring IP RTP Reserve reduces the "available interface bandwidth" counter by the amount of bandwidth reserved. This counter is needed for the purpose of admission control with RSVP flows, and user-defined classes in CBWFQ. By default, the amount of "available" bandwidth is 75% of total interface bandwidth. The idea is to "underestimate" the bandwidth, allowing some space for routing updates, Layer 2 keepalives, and any other data such as physical layer bit stuffing. This implies that the <*bandwidth*> value cannot be more than <*max-reserved-bandwidth*>*<*interface bandwidth*> - that is, the interface "available" bandwidth. In this task the **max-reserved-bandwidth 100** command is used to allow 128Kbps to be reserved.

The WFQ weight assigned to the special reserved queue is a fixed value of 128, and does not depend on the <*bandwidth*> setting. This value is relatively low to be able to preempt any data traffic flow with an IP precedence in the range of 0 through 7, since the best weight value for IP Precedence 7 traffic is 32384/(7+1) = 4048.

However, a special internal policer, with the rate equal to the configured bandwidth, applies to the matched traffic, and all exceeding packets will have a WFQ weight of 32384. This means that traffic in the RTP port range that exceeds the configured bandwidth is treated as if it had an IP Precedence of 0, which equates to best-effort forwarding. This is a special measure to make sure that the reserved conversation does not consume all of the interface bandwidth by using its low weight. Note that exceeding packets are not dropped, just handled by the WFQ scheduler as if they have less important weight.

```
Rack1R4#show queueing fair
Current fair queue configuration:

  Interface             Discard    Dynamic  Reserved  Link     Priority
                        threshold  queues   queues    queues   queues
  Serial0/0/0           64         256      0         8        1
  Serial0/1/0           64         32       1         8        1
```

Copyright © 2010 Internetwork Expert

To verify IP RTP reserve in action we will configure R6 to source G.729-like packet streams to R5 through the IP SLA feature, and measure jitter and round-trip time (RTT). R5 will be configured to respond to these probes. At the same time R1 will be configured as an HTTP server and allow other routers to download its IOS image, which will cause additional congestion in the output queue of R4. R6 will mark outgoing packets with an IP Precedence value of 1, and R1 will mark outgoing packets with an IP precedence value of 7 (the maximum priority and minimum WFQ weight). Note that the Frame-Relay interface of R5 is shutdown to make sure R1 prefers to reach R5 through R4, and vice versa.

```
R1:
ip http server
ip http path flash:
ip tcp mss 100
!
policy-map MARK
 class class-default
  set ip precedence 7
!
interface FastEthernet 0/0
 service-policy output MARK

R5:
interface Serial 0/0/0
 shutdown
!
ip sla responder

R6:
policy-map MARK
 class class-default
  set ip precedence 1
!
interface FastEthernet 0/0.146
 service-policy output MARK
!
ip sla monitor 1
 type jitter dest-ipaddr 155.1.45.5 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
!
ip sla monitor schedule 1 life forever start-time now
```

Copyright © 2010 Internetwork Expert

Disable RTP reserve on R4 and start transferring the IOS image file from R1 to SW4.

```
R4:
interface Serial 0/1/0
 no ip rtp reserve
```

```
Rack1SW4#copy http://admin:cisco@155.1.146.1/c2600-adventerprisek9-
mz.124-10.bin null:
```

Note the WWW file transfer has a minimum possible weight for a dynamic conversation (4048) based on its high IP precedence value of 7.

```
Rack1R4#show queueing interface serial 0/1/0
<snip>
  (depth/weight/total drops/no-buffer drops/interleaves) 19/4048/170/0/0
  Conversation 7, linktype: ip, length: 160
  source: 155.1.146.1, destination: 155.1.108.10, id: 0x3E96, ttl: 254, prot: 6

  (depth/weight/total drops/no-buffer drops/interleaves) 1/16192/22/0/0
  Conversation 2, linktype: ip, length: 64
  source: 155.1.146.6, destination: 155.1.45.5, id: 0x000F, ttl: 254,
  TOS: 32 prot: 17, source port 59144, destination port 16384
```

Note the RTT values for IP SLA probes and low mean opinion score (MOS) indicates poor voice quality.

```
Rack1R6#show ip sla monitor statistics 1
Round trip time (RTT) Index 1
       Latest RTT: 458 ms
Latest operation start time: 07:34:06.306 UTC Mon Aug 4 2008
Latest operation return code: OK
RTT Values
       Number Of RTT: 978
       RTT Min/Avg/Max: 26/468/701 ms
Latency one-way time milliseconds
       Number of one-way Samples: 0
       Source to Destination one way Min/Avg/Max: 0/0/0 ms
       Destination to Source one way Min/Avg/Max: 0/0/0 ms
Jitter time milliseconds
       Number of Jitter Samples: 955
       Source to Destination Jitter Min/Avg/Max: 1/5/24 ms
       Destination to Source Jitter Min/Avg/Max: 1/1/11 ms
Packet Loss Values
       Loss Source to Destination: 22            Loss Destination to Source: 0
       Out Of Sequence: 0   Tail Drop: 0  Packet Late Arrival: 0
Voice Score Values
       Calculated Planning Impairment Factor (ICPIF): 32
MOS score: 3.15
Number of successes: 81
Number of failures: 0
Operation time to live: Forever
```

Copyright © 2010 Internetwork Expert

With the current design the voice packets with IP precedence 1 cannot compete with the larger TCP packets that have an IP precedence of 7. This is due to the large mismatch in their weights. Normally VoIP packets would have an IP precedence of 5, which would help somewhat, but IP precedence 7 would still overtake it as a whole. Note the weight values and the average packet sizes.

Now re-enable the RTP reserve feature and compare the weights for both flows once more.

```
R4:
interface Serial 0/1/0
 ip rtp reserve 16384 16383 128

Rack1R4#show queueing interface serial 0/1/0
<snip>
  (depth/weight/total drops/no-buffer drops/interleaves) 1/128/0/0/0
  Conversation 41, linktype: ip, length: 64
  source: 155.1.146.6, destination: 155.1.45.5, id: 0x0046, ttl: 254,
  TOS: 32 prot: 17, source port 58177, destination port 16384

  (depth/weight/total drops/no-buffer drops/interleaves) 19/4048/170/0/0
  Conversation 7, linktype: ip, length: 160
  source: 155.1.146.1, destination: 155.1.108.10, id: 0x4497, ttl: 254, prot: 6
```

The weight value assigned to the VoIP traffic flow is now 128 due to the IP RTP Reserve, which is much better than the weight assigned to data transfer. Also, note the number of the conversation for this reserved flow is 41, which is 32+8+1, meaning that we have 32 dynamic flow queues on the interface.

Copyright © 2010 Internetwork Expert

Now check the IP SLA statistics to confirm the increase in voice quality. Note the improved MOS (Mean Opinion Score) and the decreased RTT value.

```
Rack1R1#show ip sla monitor statistics 1
Round trip time (RTT) Index 1
       Latest RTT: 18 ms
Latest operation start time: 07:35:33.300 UTC Mon Aug 4 2008
Latest operation return code: OK
RTT Values
       Number Of RTT: 1000
       RTT Min/Avg/Max: 12/18/34 ms
Latency one-way time milliseconds
       Number of one-way Samples: 0
       Source to Destination one way Min/Avg/Max: 0/0/0 ms
       Destination to Source one way Min/Avg/Max: 0/0/0 ms
Jitter time milliseconds
       Number of Jitter Samples: 999
       Source to Destination Jitter Min/Avg/Max: 1/4/9 ms
       Destination to Source Jitter Min/Avg/Max: 1/1/12 ms
Packet Loss Values
       Loss Source to Destination: 0            Loss Destination to Source: 0
       Out Of Sequence: 0   Tail Drop: 0  Packet Late Arrival: 0
Voice Score Values
       Calculated Planning Impairment Factor (ICPIF): 11
MOS score: 4.06
Number of successes: 85
Number of failures: 0
Operation time to live: Forever
```

10.4 Legacy RTP Prioritization

- Reset R4's Serial link to R5 to the default fair-queue values, and remove the RTP reservation.
- Configure legacy RTP prioritization for UDP ports in the range 16383 – 32767 up to 128Kbps.

Configuration

```
R4:
interface Serial 0/1/0
 bandwidth 128
 no hold-queue out
 fair-queue
 max-reserved-bandwidth 100
 no ip rtp reserve
 ip rtp priority 16384 16383 128
```

Verification

Note

The **ip rtp priority** feature was the next natural progression beyond the **ip rtp reserve** feature, as a strict priority queue replaced the RSVP conversation used to schedule the VoIP packets. The IP RTP Priority feature differs from the IP RTP Reserve in that the priority queue has a WFQ weight of zero, meaning that the WFQ always services it first. Packets leaving the priority queue are rate-limited using the configurable bandwidth setting, and use a burst size of 1 second (e.g. for the bandwidth 128Kbps the burst is 128Kbps*1 second = 16 Kbyte). Furthermore, the IP RTP Priority feature performs an additional check to ensure that only *even* UDP port numbers are matched, which increases the VoIP traffic classification quality.

The biggest advantage of the feature is that the voice conversation no longer has to compete with *any* flow, because of its prioritized nature. This is the reason that rate limiting is necessary, to prevent the other queues from being starved.

The command syntax is the same as with IP RTP Reserve, and reads **ip rtp priority *<Starting UDP Port> <Port Range> <Bandwidth>***. Here *<Bandwidth> means* the policer rate. As with RTP reserve, the *<Bandwidth>* cannot exceed *<max-reserved-bandwidth>*<interface bandwidth>*, which is why the **max-reserved-bandwidth** is changed to 100% in this particular task. Note that the IP RTP priority drops exceeding packets, while IP RTP Reserve simply changes their scheduling weight.

Copyright © 2010 Internetwork Expert

The flow queue number assigned to the priority queue is set right after the special "Link Queue" numbers (used to queue routing updates, L2 keepalives, CDP etc). If there are 2^N dynamic flow queues, the priority queue number is 2^{N+8}. For example, if there are 32 dynamic queues, the priority queue number is 40.

```
Rack1R4#show queueing fair
Current fair queue configuration:

  Interface             Discard    Dynamic  Reserved  Link     Priority
                        threshold  queues   queues    queues   queues
  Serial0/0/0           64         256      0         8        1
  Serial0/1/0           64         32       0         8        1
```

The verifications of this feature are the same as with RTP Reserve. R6 sources G.729-like packet streams to R5 through the IP SLA feature, and measure jitter and round-trip time (RTT). At the same time R1 will be configured as an HTTP server and allow other routers to download its IOS image, which will cause additional congestion in the output queue of R4. R6 will mark outgoing packets with an IP Precedence value of 1, and R1 will mark outgoing packets with an IP precedence value of 7 (the maximum priority and minimum WFQ weight). Note that the Frame-Relay interface of R5 is shutdown to make sure R1 prefers to reach R5 through R4, and vice versa.

```
R1:
ip http server
ip http path flash:
!
policy-map MARK
 class class-default
  set ip precedence 7
!
interface FastEthernet 0/0
 service-policy output MARK

R5:
interface Serial 0/0
 shutdown
!
ip sla responder

R6:
policy-map MARK
 class class-default
  set ip precedence 1
!
interface FastEthernet 0/0.146
 service-policy output MARK
!
ip sla monitor 1
 type jitter dest-ipaddr 155.1.45.5 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
!
ip sla monitor schedule 1 life forever start-time now
```

Copyright © 2010 Internetwork Expert

Start transferring the large file (IOS image) from R1 to SW4.

```
Rack1SW4#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-
mz.124-24.T.bin null:
```

Observe the queue of R4's Serial interface.

```
Rack1R4#show queue serial 0/1/0
<snip>

  (depth/weight/total drops/no-buffer drops/interleaves) 1/0/0/0/0
  Conversation 40, linktype: ip, length: 64
  source: 155.1.146.6, destination: 155.1.45.5, id: 0x01A0, ttl: 254,
  TOS: 32 prot: 17, source port 57143, destination port 16384

  (depth/weight/total drops/no-buffer drops/interleaves) 32/4048/0/0/0
  Conversation 7, linktype: ip, length: 160
  source: 155.1.146.1, destination: 155.1.108.10, id: 0x9280, ttl: 254, prot: 6

Rack1R4#show interfaces serial 0/1/0
Serial0/1/0 is up, line protocol is up
  Hardware is PowerQUICC Serial
  Internet address is 155.1.45.4/24
<snip>
  30 second input rate 38000 bits/sec, 84 packets/sec
  30 second output rate 121000 bits/sec, 139 packets/sec
```

Note the special "conversation 40", which has no weight value assigned. This is the IP RTP priority queue. The other flow is the HTTP traffic. Now verify the probe statistics on R6.

```
Rack1R6#show ip sla monitor statistics 1
Round trip time (RTT) Index 1
       Latest RTT: 18 ms
Latest operation start time: 08:15:49.305 UTC Mon Aug 4 2008
Latest operation return code: OK
RTT Values
       Number Of RTT: 1000
       RTT Min/Avg/Max: 13/18/33 ms
Latency one-way time milliseconds
       Number of one-way Samples: 0
       Source to Destination one way Min/Avg/Max: 0/0/0 ms
       Destination to Source one way Min/Avg/Max: 0/0/0 ms
Jitter time milliseconds
       Number of Jitter Samples: 999
       Source to Destination Jitter Min/Avg/Max: 1/5/10 ms
       Destination to Source Jitter Min/Avg/Max: 1/1/15 ms
Packet Loss Values
       Loss Source to Destination: 0            Loss Destination to Source: 0
       Out Of Sequence: 0   Tail Drop: 0  Packet Late Arrival: 0
Voice Score Values
       Calculated Planning Impairment Factor (ICPIF): 11
MOS score: 4.06
Number of successes: 29
Number of failures: 0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.5 Legacy Custom Queueing

- Remove R4's WFQ configuration on the Serial link connecting to R5.
- Configure legacy custom queueing on this link so that traffic is round-robin scheduled based on the following requirements:
  - RTP VoIP traffic should be guaranteed 30% of the link bandwidth.
  - File transfers from web servers in VLAN 146 should be guaranteed 60% of the link bandwidth.
  - The remaining 10% should be allocated to ICMP, but this traffic should not exceed 10 packets in the queue at any given time.
  - RIP routing packets should use the system priority queue.
- Clock the link at 128Kbps, and ensure that IP packets larger than 156 bytes are fragmented.
- Assume packet sizes of 60 bytes for RTP, 156 bytes for TCP, and 100 bytes for ICMP.
- Ensure to account for the layer 2 HDLC encapsulation overhead.

Configuration

```
R4:
access-list 100 permit tcp 155.1.146.0 0.0.0.255 eq www any
access-list 101 permit icmp any any
!
queue-list 1 protocol ip 0 udp 520
queue-list 1 protocol ip 1 lt 65
queue-list 1 protocol ip 2 list 100
queue-list 1 protocol ip 3 list 101
queue-list 1 default 3
queue-list 1 queue 3 limit 10
queue-list 1 queue 1 byte-count 320
queue-list 1 queue 2 byte-count 640
queue-list 1 queue 3 byte-count 104
!
interface Serial 0/1/0
 ip mtu 156
 custom-queue-list 1

R5:
interface Serial 0/1/0
 clock rate 128000
 ip mtu 156
```

Copyright © 2010 Internetwork Expert

Verification

Note

Legacy custom queueing is similar to weighted fair queueing in that it tries to share the bandwidth between packet flows in a max-min fashion. This means that each flow gets a guaranteed minimum share that is directly proportional to its weight, but any flow can claim the unused interface bandwidth if other flows are not using their guaranteed rate.

Unlike WFQ there are no dynamic flows, but instead there are 16 static queues that must be manually defined. The custom queue assigns a byte counter to every configured queue, with a 1500-byte default value, and services the queues in round-robin fashion. Every time a queue is serviced the de-queued packet decrements the byte count by the packet size until the counter drops to zero, at which time the next queue is serviced.

Custom Queuing also supports an additional system queue, number 0. The system queue is the priority queue, and is always serviced first, before all other regular queues. By default the system queue is used for layer 2 keepalives, but not for routing update packets (e.g. RIP, OSPF, or EIGRP). Therefore routing updates should be mapped manually to queue 0, unless the interface is running Frame Relay encapsulation, which uses a special broadcast queue to send multicast routing updates. All unclassified packets are assigned to queue 1 by default unless the default queue number is changed.

The limitation of round robin scheduling is that it cannot de-queue less than one packet each time a queue is serviced. Furthermore, since different queues can have different average packet sizes, e.g. voice packets of 60 bytes and TCP packets of 1500 bytes, this can lead to an undesirable bandwidth distribution ratio. For example if a queue's byte counter is 100 bytes, but a packet of 1500 bytes is in the queue, the packet is still sent since the counter is not zero. In order to make the distribution "fair", every queue's byte counter traditionally had to be proportional to the queue's average packet size.

In IOS 12.1 the behavior of the Custom Queue changed to fix this distribution problem by implementing a so-called Deficit Round Robin. The new deficit based algorithm tracks the amount of excessive bytes consumed by every queue, and takes that into account in the next scheduling rounds. In essence a queue can "borrow" credit in order to send a packet whose size exceeds the byte count left, but the debt is paid back the next time the queue is serviced. Based on this change it is no longer necessary to set the byte counts proportional to average packets sizes. However, as a general rule setting the byte counts higher will increase the scheduling delay, but will make the traffic distribution smoother.

Copyright © 2010 Internetwork Expert

In this example the traffic *is* classified based on the packet size, and the byte counts are computed based on this. Specifically in this task there are three traffic classes, RTP VoIP packets with a layer 3 size of 60 bytes (G.729 samples with 20 bytes per packet – 50pps), TCP packets with a layer 3 size of 156 bytes (due to the MTU adjustment forcing fragmentation), and ICMP packets with a layer 3 size of 100 bytes. Note that the custom queue byte counter does take Layer 2 overhead into consideration, so an additional 4 bytes is accounted for in the HDLC protocol overhead. The result is that our counters are based on frame sizes of 64 bytes, 160 bytes, and 104 bytes respectively. The desired distribution ratio of 30:60:10 is then calculated as follows.

Since the byte counters must be proportional to the packet size, the following should remain true: C1=a*64, C2=b*160; C3=c*104 where “a”, “b” and “c” are the multipliers we are looking for and C(i) is the byte-counter for queue “i”. From the proportion C1:C2:C3=30:60:10 we get a=30/64=0.47, b=60/160=0.38, and c=10/104=0.096. Now the problem is that those multiples are fractional, and we can’t hold a fractional number of packets in a queue.

To get an optimal whole number, we must first normalize the fractional numbers by dividing by the smallest one. In our case this is c=0.096. Therefore, now a1=a/c=4.89; b1=b/c=3.95; c1=1. It’s now possible to round up to the next integer and use this as the number of packets that should be sent every turn, but what if we had numbers like 3.5 or 4.2? Rounding them up we get integers far larger than the initial values, which results in a different packet distribution than desired. To avoid this we could scale (multiply) all numbers by some integer value (e.g. by 2 or 3) to obtain a value closer to a whole number.

In our case we could simply multiply by 100 to get exact whole numbers, but this would yield excessively high byte counts, and would result in considerable delays between round-robin scheduler runs. In addition to this when calculating the final byte counts we need to make sure the queue depths allow for large counters in case we still want to use them. For this example we simply round up to get multipliers a=5, b=4, and c=1. The resulting byte counters are C1=a*64=320, C2=b*160=640, C3=c*104=104. Now let’s calculate the weights each queue obtains with this configuration.

W1=C1/(C1+C2+C3)=320/(640+320+104)=320/1064=0.3
W2=C2/(C1+C2+C3)=640/1064=0.6
W3=C3/(C1+C2+C3)=104/1064=0.1

Copyright © 2010 Internetwork Expert

This is what we want – the proportions of 30:60:10. Also, the maximum byte counter is 640 bytes, so the maximum possible delay would be based on 4x160 byte packets, resulting in 40ms of maximum delay (not too VoIP friendly).

When traffic is mapped to a queue number, the system evaluates all statements sequentially, and the first one matched by the packet determines the queue where it goes. This is similar to how the modular quality of service policy-map matches classes in a top-down fashion. In this example packets with a size less than 65 bytes are matched in queue 1, per the statement **`queue-list 1 protocol ip 1 lt 65`**. This means that ICMP packets smaller than 65 bytes will be classified using the first statement, instead of the third. Also note that the size criteria with legacy QoS commands are based on the full layer 2 packet length, which means the layer 2 header, the layer 3 IP header, plus the packet payload. In this particular case the HDLC header is 4 bytes, and thus voice packets have a full layer 2 size of 64 bytes. In addition when you specify a port number to be matched for CQ classification, such as the UDP port 520 for RIP, it matches both source and destination ports.

```
Rack1R4#sh queueing custom
Current custom queue configuration:

List    Queue  Args
1       3      default
1       0      protocol ip          udp port rip
1       1      protocol ip          lt 65
1       2      protocol ip          list 100
1       3      protocol ip          list 101
1       1      byte-count 320
1       2      byte-count 640
1       3      byte-count 104 limit 10

Rack1R4#sh interfaces serial 0/1/0
Serial0/1/0 is up, line protocol is up
  <snip>
  Queueing strategy: custom-list 1
  Output queues: (queue #: size/max/drops)
     0: 0/20/0 1: 0/20/0 2: 0/20/0 3: 0/10/0 4: 0/20/0
     5: 0/20/0 6: 0/20/0 7: 0/20/0 8: 0/20/0 9: 0/20/0
     10: 0/20/0 11: 0/20/0 12: 0/20/0 13: 0/20/0 14: 0/20/0
     15: 0/20/0 16: 0/20/0
  5 minute input rate 0 bits/sec, 0 packets/sec
  <snip>
```

Copyright © 2010 Internetwork Expert

To further verify this configuration R6 is set to source a stream of G.729 packets to R5, R5 is set to respond, and the jitter and RTT are measured. The G.729 stream consumes slightly more than 24Kbps of bandwidth. At the same time R1 is configured as an HTTP server, and its IOS image is transferred to SW2. Finally note that the Frame Relay interface of R5 is disabled to force these traffic flows to route over the interface configured with the custom queueing parameters.

```
R1:
ip http server
ip http path flash:

R5:
interface Serial 0/0/0
 shutdown
!
ip sla responder

R6:
ip sla monitor 1
 type jitter dest-ipaddr 155.1.45.5 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
!
ip sla monitor schedule 1 life forever start-time now
```

Copyright © 2010 Internetwork Expert

Configure R5 to meter the incoming traffic rate with an MQC policy. The three traffic classes below match the packets generated by the SLA probe, the WWW traffic, and the ICMP packets. When applied to the Serial link we can view the traffic utilization on a per-class basis. Also note that the MQC matches the layer 3 packet size (e.g. 60 bytes for G.729) before any layer 2 overhead.

```
R5:
access-list 100 permit tcp 155.1.146.0 0.0.0.255 eq www any
access-list 101 permit icmp any any
!
class-map match-all ICMP
 match access-group 101
!
class-map match-all VOICE
 match packet length min 60 max 60
!
class-map match-all WWW
 match access-group 100
!
policy-map METER
 class VOICE
 class WWW
 class ICMP
!
interface Serial 0/1/0
 service-policy input METER
 load-interval 30
```

Start transferring R1's IOS to SW2, and send a barrage of ICMP packets from SW1 to R5.

```
Rack1SW2#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-
mz.124-24.T.bin null:

Rack1SW1#ping 155.1.45.5 size 100 repeat 100000000 timeout 0

Type escape sequence to abort.
Sending 100000, 100-byte ICMP Echos to 155.1.45.5, timeout is 1
seconds:
 ..............
<snip>
```

Copyright © 2010 Internetwork Expert

Now verify the custom queue settings and statistics on R4.

```
Rack1R4#show queueing custom
Current custom queue configuration:

List    Queue  Args
1       3      default
1       1      protocol ip          lt 65
1       2      protocol ip          list 100
1       3      protocol ip          list 101
1       0      protocol ip          udp port 520
1       1      byte-count 320
1       2      byte-count 640
1       3      byte-count 104 limit 10
```

Below, the current queue utilization are checked. Note that almost all queues have a non-zero depth, meaning packets are currently queued up, and the queue for ICMP packets has the maximum amount of packet drops. Also note that the voice packets (queue 1) are queued as well, but the drop count is very low.

```
Rack1R4#show interfaces serial 0/1/0
Serial0/1/0 is up, line protocol is up
<snip>
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops:
1332299
  Queueing strategy: custom-list 1
  Output queues: (queue #: size/max/drops)
      0: 0/20/0 1: 8/20/55 2: 11/20/24497 3: 10/10/1307747 4: 0/20/0
     5: 0/20/0 6: 0/20/0 7: 0/20/0 8: 0/20/0 9: 0/20/0
     10: 0/20/0 11: 0/20/0 12: 0/20/0 13: 0/20/0 14: 0/20/0
     15: 0/20/0 16: 0/20/0
  5 minute input rate 49000 bits/sec, 95 packets/sec
  5 minute output rate 121000 bits/sec, 143 packets/sec
<snip>
```

Copyright © 2010 Internetwork Expert

The current queue contents can be viewed with the command **show queue *<interface> <slot/port> <qNumber>***. For example, the second queue's contents are displayed below.

```
Rack1R4#show queue serial 0/1/0 2
Output queue for Serial0/1/0 is 5/20

Packet 1, linktype: ip, length: 160, flags: 0x88
  source: 155.1.146.1, destination: 155.1.58.8, id: 0x60A7, ttl: 254,
prot: 6
    data: 0x9577 0x93FD 0xB6FB 0xA7DC 0x7E54 0xBD08 0x74B7
          0xB79C 0x3EF4 0x27E1 0x75B2 0xC3B7 0x37F3 0xDC17

Packet 2, linktype: ip, length: 160, flags: 0x88
  source: 155.1.146.1, destination: 155.1.58.8, id: 0x60A8, ttl: 254,
  TOS: 0 prot: 6, source port 80, destination port 11015
    data: 0x0050 0x2B07 0x21FC 0xF1E0 0xA32B 0x3C56 0x5010
          0x0F62 0x7866 0x0000 0x3DBE 0xBE6D 0xE70D 0x3E27

Packet 3, linktype: ip, length: 160, flags: 0x88
  source: 155.1.146.1, destination: 155.1.58.8, id: 0x60A8, ttl: 254,
prot: 6
    data: 0x39FD 0xBEC1 0xC76A 0xF5F7 0x37FC 0xB731 0x4E09
          0xDEB5 0x761C 0xE6F6 0x7390 0xB3B3 0x791C 0x400A
<snip>
```

Check the SLA probe statistic on R6, and note the high RTT. This is because the voice packets are not in a priority queue, but are serviced as regular classes.

```
Rack1R6#show ip sla monitor statistics 1
Round trip time (RTT)     Index 1
      Latest RTT: 85 ms
Latest operation start time: 05:28:54.827 UTC Tue Aug 5 2008
Latest operation return code: OK
RTT Values
      Number Of RTT: 1000
      RTT Min/Avg/Max: 12/85/169 ms
Latency one-way time milliseconds
      Number of one-way Samples: 0
      Source to Destination one way Min/Avg/Max: 0/0/0 ms
      Destination to Source one way Min/Avg/Max: 0/0/0 ms
Jitter time milliseconds
      Number of Jitter Samples: 999
      Source to Destination Jitter Min/Avg/Max: 2/26/148 ms
      Destination to Source Jitter Min/Avg/Max: 1/2/19 ms
Packet Loss Values
      Loss Source to Destination: 0         Loss Destination to Source:
0
      Out Of Sequence: 0 Tail Drop: 0 Packet Late Arrival: 0
Voice Score Values
      Calculated Planning Impairment Factor (ICPIF): 11
<snip>
```

Copyright © 2010 Internetwork Expert

Look at R5's MQC policy statistics to see the resulting bandwidth allocation. Note that the voice class's bandwidth is close to 24Kbps, and does not exceed its allocated share of 38Kbps (0.3*128K). Therefore the other classes can claim the remaining bandwidth. This explains why WWW traffic has more bandwidth than its guaranteed share of 76Kbps (0.6*128K).

Note the ICMP packet flood gets 13Kbps of bandwidth, and does not seriously impair the other traffic flows like it did with WFQ configured. This is because each class has its separate FIFO queue, and traffic flows do not share the same buffer space (plus there are no congestive discard procedures). Furthermore, the aggressive ICMP traffic behavior does not seriously affect the other queues since the scheduler drops the exceeding traffic.

In total, all classes draw approximately 120Kbps of L3 bandwidth, which is naturally less than the configured 128Kbps line rate.

```
Rack1R5#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy input: METER

    Class-map: VOICE (match-all)
      878252 packets, 56208128 bytes
      1 minute offered rate 23000 bps
      Match: packet length min 60 max 60

    Class-map: WWW (match-all)
      211383 packets, 72688125 bytes
      1 minute offered rate 84000 bps
      Match: access-group 100

    Class-map: ICMP (match-all)
      68628 packets, 7137268 bytes
      1 minute offered rate 13000 bps
      Match: access-group 101

    Class-map: class-default (match-any)
      2573 packets, 630376 bytes
      1 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.6 Legacy Custom Queueing with Prioritization

- Modify R4's custom queueing configuration so that the RTP VoIP traffic is prioritized after the RIP packets.

Configuration

```
R4:
queue-list 1 lowest-custom 2
```

Verification

In this example queue 1 is next in line for prioritization after the system queue 0. The **lowest-custom** option tells the round robin scheduler where to start, meaning that all queues up to this number will use priority scheduling. Queue 0 is the most important, queue 1 is the next in importance, etc.

Note that even though this configuration allows for priority queueing and custom queueing at the same time, it is not recommended, because there is no way to limit the priority queues. Just like with the legacy priority queue, this means that other traffic classes can be effectively starved of bandwidth if there are consistently packets in the priority queues. In a practical design, voice traffic should use either the IP RTP Priority feature, or the MQC low-latency queue (LLQ), due to their built-in policer functions.

The above configuration can be verified as follows.

```
Rack1R4#clear counters Serial 0/1/0
Clear "show interface" counters on this interface [confirm]
Rack1R4#conf t
Rack1R4(config)#queue-list 1 lowest-custom 2

Rack1R4#show queueing custom
Current custom queue configuration:

List    Queue   Args
1       2       lowest custom queue
1       3       default
1       1       protocol ip          lt 65
1       2       protocol ip          list 100
1       3       protocol ip          list 101
1       0       protocol ip          udp port 520
1       1       byte-count 320
1       2       byte-count 640
1       3       byte-count 104 limit 10
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show interfaces serial 0/1/0
Serial0/1/0 is up, line protocol is up
<snip>
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops:
1053009
  Queueing strategy: custom-list 1
  Output queues: (queue #: size/max/drops)
     0: 0/20/0 1: 0/20/55 2: 7/20/20301 3: 7/10/1032653 4: 0/20/0
     5: 0/20/0 6: 0/20/0 7: 0/20/0 8: 0/20/0 9: 0/20/0
     10: 0/20/0 11: 0/20/0 12: 0/20/0 13: 0/20/0 14: 0/20/0
     15: 0/20/0 16: 0/20/0
  5 minute input rate 49000 bits/sec, 95 packets/sec
  5 minute output rate 121000 bits/sec, 144 packets/sec
<snip>
```

Now verify the SLA statistics for the RTP packet probe on R6. Note that RTT has dropped significantly since the packets now go through the priority queue.

```
Rack1R6#show ip sla monitor statistics 1
Round trip time (RTT)     Index 1
      Latest RTT: 19 ms
Latest operation start time: 05:44:00.485 UTC Tue Aug 5 2008
Latest operation return code: OK
RTT Values
      Number Of RTT: 1000
      RTT Min/Avg/Max: 12/19/35 ms
Latency one-way time milliseconds
      Number of one-way Samples: 0
      Source to Destination one way Min/Avg/Max: 0/0/0 ms
      Destination to Source one way Min/Avg/Max: 0/0/0 ms
Jitter time milliseconds
      Number of Jitter Samples: 999
      Source to Destination Jitter Min/Avg/Max: 1/4/10 ms
      Destination to Source Jitter Min/Avg/Max: 1/2/18 ms
Packet Loss Values
      Loss Source to Destination: 0         Loss Destination to Source:
0
      Out Of Sequence: 0 Tail Drop: 0 Packet Late Arrival: 0
Voice Score Values
      Calculated Planning Impairment Factor (ICPIF): 11
MOS score: 4.06
Number of successes: 91
Number of failures: 0
Operation time to live: Forever
```

Copyright © 2010 Internetwork Expert

Check bandwidth usage on R5. Note that bandwidth allocation did not change significantly since the voice traffic rate remained the same.

```
Rack1R5#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy input: METER

    Class-map: VOICE (match-all)
      916126 packets, 58632064 bytes
      1 minute offered rate 25000 bps
      Match: packet length min 60 max 60

    Class-map: WWW (match-all)
      277613 packets, 81443501 bytes
      1 minute offered rate 82000 bps
      Match: access-group 100

    Class-map: ICMP (match-all)
      82087 packets, 8537004 bytes
      1 minute offered rate 13000 bps
      Match: access-group 101

    Class-map: class-default (match-any)
      2881 packets, 675328 bytes
      1 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.7 Legacy Priority Queueing

- Remove R4's custom queueing configuration and replace it with legacy priority queueing per the following requirements:
  - All RIP packets going out the Serial link to R5 should be sent first.
  - If there are no RIP packets, RTP VoIP should be sent next.
  - If there are no RIP or VoIP packets, web traffic should be sent next.
  - As a last resort ICMP traffic should be sent.
- Set the queue-sizes for the high, medium, normal, and low queues to 5, 40, 60, and 80 packets respectively.

Configuration

```
R4:
access-list 102 permit udp any any range 16384 32767
access-list 103 permit icmp any any
!
priority-list 1 protocol ip high udp rip
priority-list 1 protocol http normal
priority-list 1 protocol ip medium list 102
priority-list 1 protocol ip low list 103
priority-list 1 queue-limit 5 40 60 80
!
interface Serial 0/1/0
 no custom-queue-list 1
 priority-group 1
```

Verification

Note

Unlike WFQ or legacy Custom Queuing (CQ), legacy Priority Queueing (PQ) does not provide a max-min sharing of interface bandwidth. Instead, it simply sends the most important traffic flow completely before it moves to the next important flow, and so on, from greatest to least important class. The PQ supports four static queues, unlike 16 in CQ, and a large number of dynamic queues in WFQ, and classification is based on static configuration, similar to Custom Queueing. Every queue has a limit which is based on number of packets, not bandwidth, with the lower priority queues having more packet space than the higher priority queues. The default values can be found by applying the **`priority-list`** command to an interface and issuing the **`show interface`** command.

Copyright © 2010 Internetwork Expert

The biggest design flaw with Priority Queueing is that the lower queues may starve if the higher queues are constantly busy. This is why legacy Priority Queueing is not very helpful for real VoIP deployments, because the VoIP traffic streams are continuous, e.g. they keep their queue busy almost consistently, and there is no way the PQ can restrict them. However, PQ is suitable for use in networks where sporadic traffic bursts of different priorities exist. For example, you may want to give OLTP database traffic the most priority and allow it to preempt other traffic types, but the traffic flow is only periodic.

Just like the legacy Custom Queue the port numbers specified in the **`priority-list`** command match the source or destination port, and the packet length match checks against the full layer 2 packet size.

To verify this configuration R6 is set to source a stream of G.729 packets to R5, R5 is set to respond, and the jitter and RTT are measured. The G.729 stream consumes slightly more than 24Kbps of layer 3 bandwidth. At the same time R1 is configured as an HTTP server, and its IOS image is transferred to SW2. Finally note that the Frame Relay interface of R5 is disabled to force these traffic flows to route over the interface configured with the priority qeueing parameters.

```
Rack9R4#sh interfaces s0/1/0
Serial0/1/0 is up, line protocol is up
  <snip>
  Queueing strategy: priority-list 1
  Output queue (queue priority: size/max/drops):
     high: 0/5/0, medium: 0/40/0, normal: 0/60/0, low: 0/80/0
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 1000 bits/sec, 1 packets/sec
     1559 packets input, 297222 bytes, 0 no buffer
  <snip>
```

Copyright © 2010 Internetwork Expert

To further verify this configuration R6 is set to source a stream of G.729 packets to R5, R5 is set to respond, and the jitter and RTT are measured. The G.729 stream consumes slightly more than 24Kbps of layer 3 bandwidth. At the same time R1 is configured as an HTTP server, and its IOS image is transferred to SW2. Finally note that the Frame Relay interface of R5 is disabled to force these traffic flows to route over the interface configured with the priority qeueing parameters.

```
R1:
ip http server
ip http path flash:

R5:
interface Serial 0/0
 shutdown
!
ip sla responder

R6:
ip sla monitor 1
 type jitter dest-ipaddr 155.1.45.5 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
!
ip sla monitor schedule 1 life forever start-time now
```

Start transferring R1's IOS image to SW2, and ping R5's serial interface from SW1. Note that the ping packets cannot make it through R4 while the file is transferring from R1 to SW2. This is because the higher queues completely block the low priority queue.

```
Rack1SW2#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-
mz.124-24.T.bin null:
Loading http://admin:cisco@155.1.146.6/c1841-adventerprisek9-mz.124-
24.T.bin !!!!!!

Rack1SW1#ping 155.1.45.5 repeat 100000 size 100

Type escape sequence to abort.
Sending 100000, 100-byte ICMP Echos to 155.1.45.5, timeout is 2
seconds:
 ...................
```

Copyright © 2010 Internetwork Expert

Verify the priority queue statistics on the serial interface of R4. Note the high queue depth for the low queue, which is used for ICMP packets. Also note that the system uses the "high" queue for RIP updates and layer 2 keepalives; this is why we see packet matches here.

```
Rack1R4#show interfaces serial 0/1/0
Serial0/1/0 is up, line protocol is up
  <snip>
  Queueing strategy: priority-list 1
  Output queue (queue priority: size/max/drops):
     high: 0/5/0, medium: 0/40/0, normal: 6/60/0, low: 43/80/0
  5 minute input rate 40000 bits/sec, 92 packets/sec
  5 minute output rate 123000 bits/sec, 71 packets/sec
     96338 packets input, 5550390 bytes, 0 no buffer
<snip>

Rack1R4#show queueing interface serial 0/1/0
Interface Serial0/1/0 queueing strategy: priority

Output queue utilization (queue/count)
      high/149 medium/53360 normal/23090 low/5105
```

Copyright © 2010 Internetwork Expert

To view the current queue contents issue the **show queue *<interface> <slot/port> <qNumber>***. Note the current queue sizes and the maximum queue depth fields.

```
Rack1R4#show queue serial 0/1/0 2
Output queue for Serial0/1 is 32/60

Packet 1, linktype: ip, length: 160, flags: 0x88
  source: 155.1.146.1, destination: 155.1.58.8, id: 0x4727, ttl: 254,
prot: 6
    data: 0x75B6 0x856D 0x986F 0x0278 0x4B11 0xE94C 0x69B0
          0x8E0B 0xE9D9 0xA556 0x9BBA 0xF9F3 0xC018 0xE443

Packet 2, linktype: ip, length: 36, flags: 0x88
  source: 155.1.146.1, destination: 155.1.58.8, id: 0x4727, ttl: 254,
prot: 6
    data: 0xE64E 0x415B 0xD71C 0x1BFA 0xEDC2 0xAFFB 0x02AA
          0x1C36 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD
Packet 3, linktype: ip, length: 160, flags: 0x88
  source: 155.1.146.1, destination: 155.1.58.8, id: 0x4728, ttl: 254,
  TOS: 0 prot: 6, source port 80, destination port 11010
    data: 0x0050 0x2B02 0xEAE5 0x5240 0x2235 0x7AA8 0x5010
          0x0F62 0x2D5F 0x0000 0x99E6 0x2C18 0x1FF6 0x3127

Packet 4, linktype: ip, length: 160, flags: 0x88
  source: 155.1.146.1, destination: 155.1.58.8, id: 0x4728, ttl: 254,
prot: 6
    data: 0x16E0 0xEB4B 0x91C7 0xA734 0xF13C 0x350A 0xE020
          0x97E3 0x00EF 0x589F 0x8CF7 0x8D03 0xFFB6 0xF0D8
0xA97C 0x0000 0x0000 0x03B4 0x0000 0xABCD 0xABCD
<snip>
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show queue serial 0/1/0 3
Output queue for Serial0/1 is 80/80

Packet 1, linktype: ip, length: 104, flags: 0x88
  source: 155.1.67.7, destination: 155.1.45.5, id: 0xD54F, ttl: 253,
prot: 1
    data: 0x0800 0x5F60 0x0012 0x0000 0x0000 0x0000 0x02AA
          0x1C2E 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD

Packet 2, linktype: ip, length: 104, flags: 0x88
  source: 155.1.67.7, destination: 155.1.45.5, id: 0xD550, ttl: 253,
prot: 1
    data: 0x0800 0x5F5F 0x0012 0x0001 0x0000 0x0000 0x02AA
          0x1C2E 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD

Packet 3, linktype: ip, length: 104, flags: 0x88
  source: 155.1.67.7, destination: 155.1.45.5, id: 0xD551, ttl: 253,
prot: 1
    data: 0x0800 0x5F5E 0x0012 0x0002 0x0000 0x0000 0x02AA
          0x1C2E 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD

Packet 4, linktype: ip, length: 104, flags: 0x88
  source: 155.1.67.7, destination: 155.1.45.5, id: 0xD552, ttl: 253,
prot: 1
    data: 0x0800 0x5F5D 0x0012 0x0003 0x0000 0x0000 0x02AA
          0x1C2E 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD
<snip>
```

Copyright © 2010 Internetwork Expert

Note that the file transfer does not affect voice packets.

```
Rack1R1#show ip sla monitor statistics 1
Round trip time (RTT)     Index 1
       Latest RTT: 18 ms
Latest operation start time: 05:07:10.702 UTC Tue Aug 5 2008
Latest operation return code: OK
RTT Values
       Number Of RTT: 1000
       RTT Min/Avg/Max: 12/18/43 ms
Latency one-way time milliseconds
       Number of one-way Samples: 0
       Source to Destination one way Min/Avg/Max: 0/0/0 ms
       Destination to Source one way Min/Avg/Max: 0/0/0 ms
Jitter time milliseconds
       Number of Jitter Samples: 999
       Source to Destination Jitter Min/Avg/Max: 1/5/26 ms
       Destination to Source Jitter Min/Avg/Max: 1/1/4 ms
Packet Loss Values
       Loss Source to Destination: 0         Loss Destination to Source:
0
       Out Of Sequence: 0 Tail Drop: 0 Packet Late Arrival: 0
Voice Score Values
       Calculated Planning Impairment Factor (ICPIF): 11
MOS score: 4.06
Number of successes: 161
Number of failures: 0
Operation time to live: Forever
```

Now stop the file transfer from R1 to SW2 and try pinging R5 from SW1 again.

```
Rack1SW1#ping 155.1.45.5 repeat 100000 size 100

Type escape sequence to abort.
Sending 100000, 100-byte ICMP Echos to 155.1.45.5, timeout is 2
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Note the queue depth for the "low" queue is zero now compared to the larger number before.

```
Rack1R4#show interfaces serial 0/1/0
Serial0/1/0 is up, line protocol is up
<snip>
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops:
320
  Queueing strategy: priority-list 1
  Output queue (queue priority: size/max/drops):
     high: 0/5/0, medium: 0/40/0, normal: 0/60/0, low: 0/80/320
<snip>
```

The voice stream quality basically remains the same:

```
Rack1R6#show ip sla monitor statistics 1
Round trip time (RTT)     Index 1
      Latest RTT: 12 ms
Latest operation start time: 05:05:25.421 UTC Tue Aug 5 2008
Latest operation return code: OK
RTT Values
      Number Of RTT: 1000
      RTT Min/Avg/Max: 11/12/31 ms
Latency one-way time milliseconds
      Number of one-way Samples: 0
      Source to Destination one way Min/Avg/Max: 0/0/0 ms
      Destination to Source one way Min/Avg/Max: 0/0/0 ms
Jitter time milliseconds
      Number of Jitter Samples: 999
      Source to Destination Jitter Min/Avg/Max: 1/1/19 ms
      Destination to Source Jitter Min/Avg/Max: 1/1/7 ms
Packet Loss Values
      Loss Source to Destination: 0         Loss Destination to Source:
0
      Out Of Sequence: 0 Tail Drop: 0 Packet Late Arrival: 0
Voice Score Values
      Calculated Planning Impairment Factor (ICPIF): 11
MOS score: 4.06
Number of successes: 156
Number of failures: 0
Operation time to live: Forever
```

Copyright © 2010 Internetwork Expert

10.8 Legacy Random Early Detection

- Configure R4's point-to-point Serial interface connecting to R5 to avoid tail drop behavior by randomly dropping packets before the output queue overflows.
- Set the hold-queue size to 10 packets, and the weighted averaging constant to 4.
- Ensure that routing updates are never randomly dropped.

Configuration

```
R4:
interface Serial0/1/0
 no priority-group 1
 random-detect
 random-detect exponential-weighting-constant 4
 random-detect precedence 6 11 12
 hold-queue 10 out
```

Verification

Note

Legacy QoS configuration supports WRED (Weighted Random Early Detection) only on main interfaces. Since WRED is essentially a FIFO queue with a modified drop strategy, you can tune its size using the **hold-queue** command. WRED parameters include the following:

1) Queue size averaging factor
2) Maximum and Minimum thresholds (per IP Precedence or per DSCP value)
3) Mark (drop) probability

Recall that IP Routing updates are sent from the router with an IP precedence of 6. In this example, the minimum threshold for IP precedence 6 is 11 packets, meaning that random dropping does not occur until this threshold is breached. However, since the tail drop threshold (hold-queue) is set to 10 packets, random dropping never occurs for these packet flows. A detailed explanation of the usage of the WRED feature is as follows.

Copyright © 2010 Internetwork Expert

A First-In First-Out (FIFO) queueing strategy is the simplest to implement, but it has several flaws critical to QoS implementations. One of them is the notion of *tail drop* used by FIFO queues. The *tail* of the output queue is where packets enter, while a packet at the *head* of the output queue is next in line for the transmit ring. Tail drop simply means that if the output queue is full, the system drops every new packet trying to enter the queue's tail, while packets already in the queue keep waiting to get to the head.

The tail drop behavior inherent to FIFO queues has undesirable effects on TCP flows, which are bursty in nature. A FIFO queue can quickly grow over its limit due to a traffic burst, even if the average sending rate is not very high. When this happens, packets are denied admission to the queue (tail drop). TCP interprets this packet loss as a signal of link congestion, and slows down its sending rate through a feature known as *slow start*. For more information on TCP congestion management techniques see RFC 2001, *TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms*.

Since tail drop usually affects many flows at one time, multiple flows will throttle down their sending rate (by going into slow start) simultaneously. Soon afterwards, these flows will synchronously increase their sending rates, resulting in another packet burst, and more FIFO tail drops. This situation is known as "global TCP synchronization", and results in an interface bandwidth utilization graph resembling a "saw tooth", with periods of high utilization followed by periods of low utilization, and an average utilization rate significantly lower than interface available bandwidth.

To avoid such behavior the Random Early Detection (RED) feature can be used to selectively drop traffic before tail dropping is necessary. The idea of RED is to recognize potential interface congestion before the interface's FIFO queue is full, and start signaling individual TCP flows to slow down the sending rate. RED is known as a *congestion avoidance* technique, because through selective dropping it tries to prevent the output queue from filling up, which reduces the need for trail dropping. The key points about the RED technique are as follows.

First and foremost, the interface queueing method is still FIFO, but packets may be dropped before the queue is absolutely full. Next, instead of tracking the current size, or depth, of the output queue, which may vary quickly depending on packet bursts, an average depth is calculated to smooth out the queue depth every time a new packet arrives. This AverageQueueSize is calculated as:

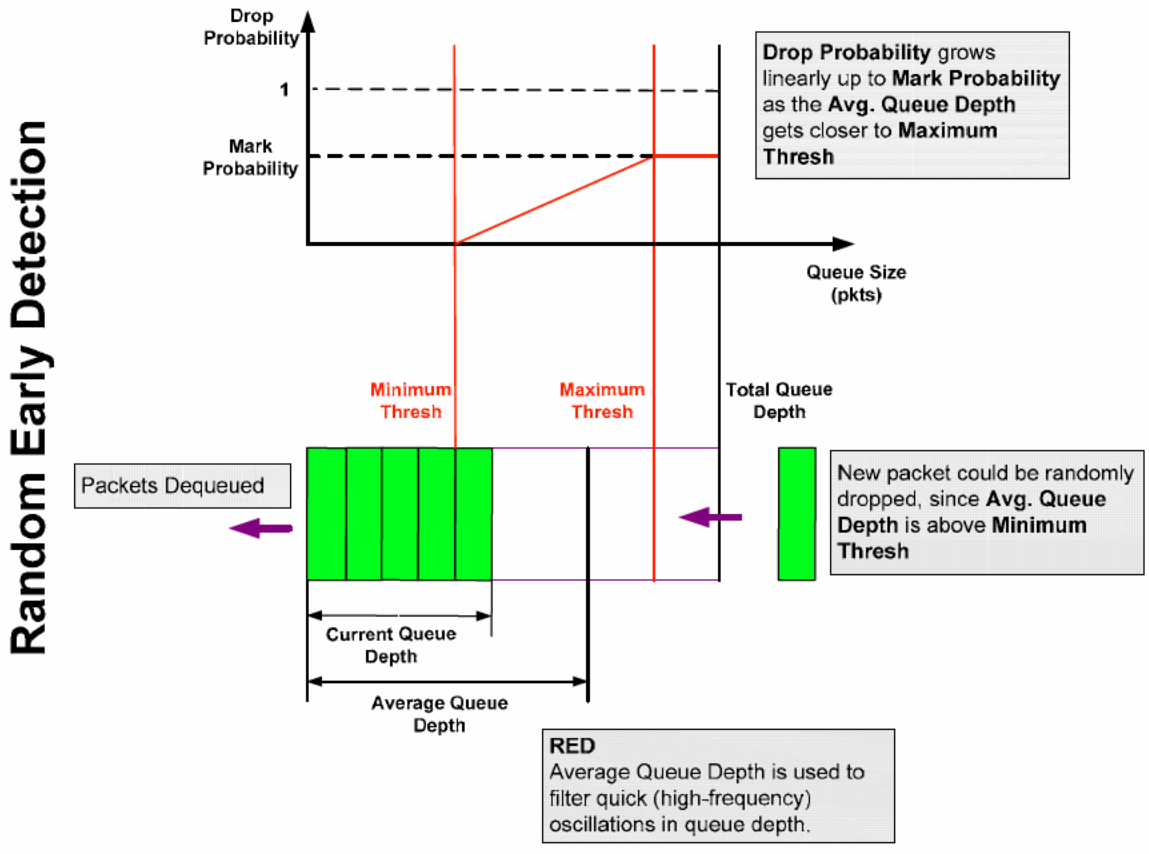
$$AvgQueueSize = OldQueueSize*(1-1/2^N) + NewQueueSize*1/(2^N)$$

Copyright © 2010 Internetwork Expert

This value slowly tracks the current queue size, lagging behind by some time interval, and dampens quick oscillations in the tracked value. The algorithm computes this average value every time a packet arrives or leaves the queue. Here *N* is the so-called *Exponential Weighting Constant*. The higher the value of *N*, the less sensitive the average queue size will be to the interface queue size changes.

To determine if and when a packet should be dropped from the queue, two thresholds are defined for the average queue depth, the *Minimum* and *Maximum* thresholds. Using these thresholds, the queue drop strategy is defined as follows.

If a new packet arrives and the *AvgQueueSize* computed is less than the *Minimum* value, the packet is accepted into the queue. If a new packet arrives and the *AvgQueueSize* is between *Minimum* and *Maximum*, then the packet is randomly dropped with the probability of *(AvgQueueSize-Minimum)/(Maximum-Minimum)*MarkProbability,* where *MarkProbability* is the last constant defined for RED (e.g. 1/10, 1/5, etc). If a new packet arrives and the *AvgQueueSize* is above *Maximum* but less than the total queue size, the packet is dropped with the probability of *MarkProbability*. Finally if a new packet arrives and the queue is 100% full, tail drop occurs.



Overall the effect is that the queue drops packets randomly before the queue size grows up to its limit. This allows for avoiding massive packet drops in multiple flows, resolving the TCP synchronization issue.

Cisco's implementation of the feature is known as Weighted Random Early Detection (WRED), or Weighted RED, which allows for defining different *Maximum* and *Minimum* thresholds as well as *MarkProbability* per IP Precedence of DSCP value. The result of WRED with the default values is that packets with a higher weighting (e.g. VoIP or routing updates) are less likely to be dropped as compared to lower weighted packets (e.g. FTP flows). All packets still share the same FIFO queue, however WRED starts dropping some packets earlier than others. As for the optimal values for WRED, they are subject to empirical verification. Cisco IOS automatically calculates the values based on the queue depth and WRED mode (IP Precedence or DSCP). The *Maximum* threshold value is automatically taken from the queue-depth of the link, meaning that if this threshold is breached packets are tail dropped. For most implementations the default values are sufficient.

RED configuration can be verified as follows. Note that you can see noticeable effects from RED only when you run multiple TCP flows on a relatively high-speed interface. For lower-speed WAN interfaces, its usually better to use WFQ.

```
Rack1R4#show interfaces serial 0/1/0
Serial0/1/0 is up, line protocol is up
  Hardware is QUICC Serial
  Internet address is 155.1.45.4/24
  MTU 1500 bytes, BW 128 Kbit, DLY 20000 usec,
     reliability 255/255, txload 247/255, rxload 35/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  Last input 00:00:00, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops:
58
  Queueing strategy: random early detection(RED)
<snip>
```

```
Rack1R4#show queueing interface serial 0/1/0
Interface Serial0/1/0 queueing strategy: random early detection (WRED)
    Random-detect not active on the dialer
    Exp-weight-constant: 4 (1/16)
    Mean queue depth: 5

  class          Random drop      Tail drop    Minimum Maximum  Mark
                  pkts/bytes       pkts/bytes    thresh  thresh  prob
      0             41/22058          0/0            5      10  1/10
      1               0/0             0/0            5      10  1/10
      2               0/0             0/0            6      10  1/10
      3               0/0             0/0            6      10  1/10
      4               0/0             0/0            7      10  1/10
      5               0/0             0/0            7      10  1/10
      6               0/0             0/0           11      12  1/10
      7               0/0             0/0            8      10  1/10
   rsvp               0/0             0/0            9      10  1/10
```

For further verification configure R1 and R6 as HTTP servers and transfer their IOS images to SW2 and SW4. In addition to this, shutdown R5's Frame-Relay interface to ensure the traffic takes the path across the serial link between R4 and R5. You may also need to increase the MTU value on R4's serial interface to make the IOS TCP stack work properly on slow connections.

```
R1 & R6:
ip http server
ip http path flash:

R4:
interface Serial 0/1
 no ip mtu
 load-interval 30

R5:
interface Serial 0/0
 shutdown
```

```
Rack1SW2#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-
mz.124-24.T.bin null:

Rack1SW4#copy http://admin:cisco@155.1.146.1/c2600-adventerprisek9-
mz.124-10.bin null:
```

Copyright © 2010 Internetwork Expert

Check the statistics and queue usage on R4:

```
Rack1R4#show interfaces serial 0/1/0
Serial0/1/0 is up, line protocol is up
  Hardware is QUICC Serial
  Internet address is 155.1.45.4/24
  MTU 1500 bytes, BW 128 Kbit, DLY 20000 usec,
     reliability 255/255, txload 247/255, rxload 37/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  Last input 00:00:05, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops:
96
  Queueing strategy: random early detection(RED)
  30 second input rate 19000 bits/sec, 53 packets/sec
  30 second output rate 124000 bits/sec, 27 packets/sec
  <snip>

Rack1R4#show queueing interface serial 0/1/0
Interface Serial0/1/0 queueing strategy: random early detection (WRED)
    Random-detect not active on the dialer
    Exp-weight-constant: 4 (1/16)
    Mean queue depth: 5

  class           Random drop      Tail drop    Minimum Maximum  Mark
                   pkts/bytes       pkts/bytes    thresh  thresh  prob
      0              97/53749          0/0            5      10  1/10
      1                0/0             0/0            5      10  1/10
      2                0/0             0/0            6      10  1/10
      3                0/0             0/0            6      10  1/10
      4                0/0             0/0            7      10  1/10
      5                0/0             0/0            7      10  1/10
      6                0/0             0/0           11      12  1/10
      7                0/0             0/0            8      10  1/10
   rsvp                0/0             0/0            9      10  1/10
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show queue serial 0/1/0
Output queue for Serial0/1/0 is 5/0

Packet 1, linktype: ip, length: 580, flags: 0x88
  source: 155.1.146.1, destination: 155.1.108.10, id: 0xAA2C, ttl: 254,
  TOS: 0 prot: 6, source port 80, destination port 11000
    data: 0x0050 0x2AF8 0x32F8 0xD825 0xE64C 0xB783 0x5010
          0x0F62 0x70AA 0x0000 0xFCC7 0x5856 0x6DD5 0xCBAF

Packet 2, linktype: ip, length: 580, flags: 0x88
  source: 155.1.146.1, destination: 155.1.108.10, id: 0xAA2D, ttl: 254,
  TOS: 0 prot: 6, source port 80, destination port 11000
    data: 0x0050 0x2AF8 0x32F8 0xDA3D 0xE64C 0xB783 0x5010
          0x0F62 0xB6AF 0x0000 0xE39F 0xBE57 0xFB4E 0x7FF1

Packet 3, linktype: ip, length: 580, flags: 0x88
  source: 155.1.146.1, destination: 155.1.108.10, id: 0xAA2E, ttl: 254,
  TOS: 0 prot: 6, source port 80, destination port 11000
    data: 0x0050 0x2AF8 0x32F8 0xDC55 0xE64C 0xB783 0x5010
          0x0F62 0xA4AB 0x0000 0xBD11 0x63D9 0x3FE1 0x7CF3

Packet 4, linktype: ip, length: 580, flags: 0x88
  source: 155.1.146.1, destination: 155.1.108.10, id: 0xAA2F, ttl: 254,
  TOS: 0 prot: 6, source port 80, destination port 11000
    data: 0x0050 0x2AF8 0x32F8 0xDE6D 0xE64C 0xB783 0x5010
          0x0F62 0x2B2F 0x0000 0x3A70 0x0EE4 0x7B30 0x63F9

Packet 5, linktype: ip, length: 580, flags: 0x88
  source: 155.1.146.1, destination: 155.1.108.10, id: 0xAA30, ttl: 254,
  TOS: 0 prot: 6, source port 80, destination port 11000
    data: 0x0050 0x2AF8 0x32F8 0xE085 0xE64C 0xB783 0x5010
          0x0F62 0xA50E 0x0000 0xF18D 0x8535 0xFCF6 0x7784
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.9 Legacy Flow-Based Random Early Detection

- Enable flow-based RED on the point-to-point Serial interface of R4.
- Set the FIFO queue depth to 10 packets.
- Set the average flow depth scale factor to 2.
- Set the maximum number of flows to 16.

Configuration

```
R4:
interface Serial0/1/0
 random-detect
 random-detect flow
 random-detect flow count 16
 random-detect flow average-depth-factor 2
 hold-queue 10 out
```

Verification

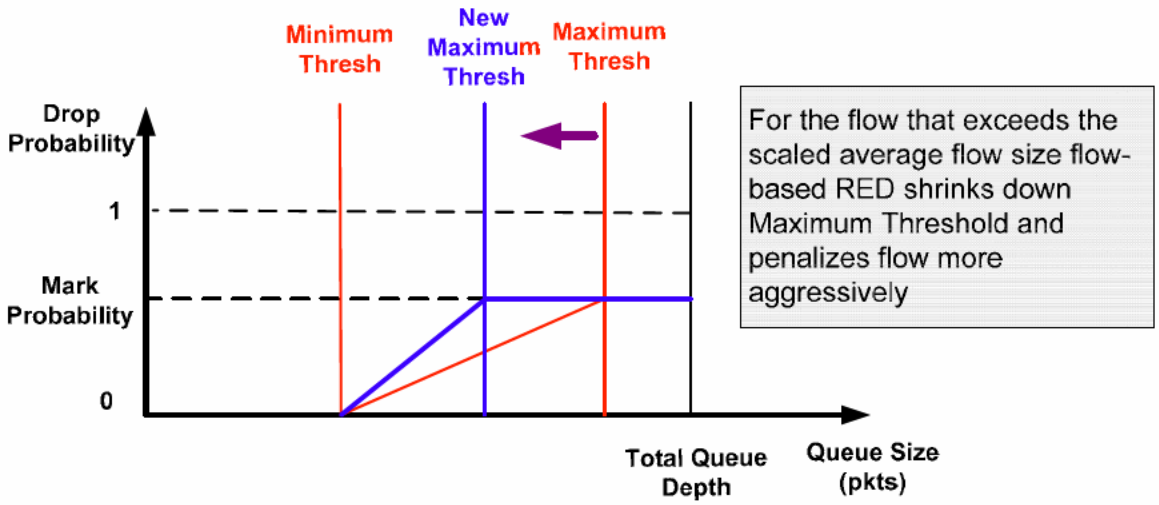
Note

Classic RED uses a FIFO queue, and does not distinguish between traffic flows (e.g. TCP/UDP sessions). This may lead to some undesirable effects, such as unevenly dropping packets in one flow while ignoring the others. This situation is more likely to happen when multiple flows with different behaviors occupy the same interface queue. RED packet drops could penalize an interactive section such as Remote Desktop Protocol (RDP) more when the interface queue is congested by an aggressive UDP flow, resulting in poor RDP session performance.

To resolve this issue, Cisco extended RED to support the concept of traffic flows. Using a special hash-function, flow-based RED classifies data streams into traffic flows, and tries to apply packet drops evenly across the flows. To accomplish this the interface's FIFO queue size *Q* is divided by the number of currently active flows *N* (flows that have queued packets in the interface queue). This yields the average queue size per flow, *Avg=Q/N*. For every flow, the flow depth *Depth* is compared with the scaled average queue size per flow as *Depth <= Avg*Scale*, where *Scale* is the scaling factor configured. If this assertion holds true, packets from the flow are not dropped. On the other hand if the flow size is greater than the scaled average queue size per flow, RED starts dropping packets from the flow more aggressively.

Copyright © 2010 Internetwork Expert

Specifically if the RED scheduler detects that a flows size is over the scaled average size, it shrinks down the maximum threshold for this flow using the formula *NewMaxThreshold = MinThreshold + (MaxThreshold-MinThreshold)/2*.



This procedure ensures that the aggressive flow is penalized more aggressively, and does not consume the queue space from other flows. For the flow-based RED it is possible to tune the maximum number of flows per interface queue and the scaling factor. This configuration can be verified as follows.

```
Rack1R4#show queueing random-detect
Current random-detect configuration:
  Serial0/1/0
    Queueing strategy: random early detection (WRED)
    Random-detect not active on the dialer
    Exp-weight-constant: 4 (1/16)
    Mean queue depth: 0
    Max flow count: 16        Average depth factor: 2
    Flows (active/max active/max): 0/0/16

  class           Random drop      Tail drop    Minimum Maximum  Mark
                   pkts/bytes       pkts/bytes    thresh  thresh  prob
      0                0/0              0/0            5      10  1/10
      1                0/0              0/0            5      10  1/10
      2                0/0              0/0            6      10  1/10
      3                0/0              0/0            6      10  1/10
      4                0/0              0/0            7      10  1/10
      5                0/0              0/0            7      10  1/10
      6                0/0              0/0           11      12  1/10
      7                0/0              0/0            8      10  1/10
   rsvp                0/0              0/0            9      10  1/10
```

Copyright © 2010 Internetwork Expert

10.10 Selective Packet Discard

- Enable Selective Packet Discard on R1 in aggressive mode.
- Increase R1's input queue size on its link to VLAN 146 to twice the default.
- Increase the amount of the memory headroom for IGP packets to 150 buffers.
- The headroom for BGP packets should be set to 120 packets.
- Start dropping low-priority packets randomly when the input queue is 50% full.

Configuration

```
R1:
spd extended-headroom 150
spd headroom 120
ip spd mode aggressive
ip spd queue max-threshold 150
ip spd queue min-threshold 75
!
interface FastEthernet 0/0
 hold-queue 150 in
```

Verification

Note

Selective Packet Discard is the queue management technique for interface input queueing. The SPD commands are hidden in the IOS parser, but you can see them in the running configuration once you enter them. By default SPD is enabled in Normal mode. The following is the list of SPD commands:

```
spd enable
spd headroom <N>
spd extended-headroom <N>
ip spd mode aggressive
ip spd queue max-threshold <N>
ip spd queue min-threshold <N>
```

Copyright © 2010 Internetwork Expert

Every physical interface has an input FIFO queue. The router uses this queue to buffer packets going to the route processor. Usually these packets include control plane packets, such as Layer 2 Keepalives (e.g. HDLC/PPP keepalives), IGP packets (OSPF, ISIS, etc.) and BGP packets. The routing protocol packets are classified based on their default IP precedence of 6 or higher. In addition to control plane packets, the input queue holds other packets destined to the route processor, such as packets with an expired TTL, wrong header length, wrong checksum, or non-existent local UDP port numbers. The latter packets are malformed, in the sense that they require the router to generate an ICMP error message in response. Lastly the input queue holds packets that are to be process-switched, which is uncommon on modern CEF-based systems.

SPD input queueing is desirable for a number of reasons. The first is for control plane security. It's possible to block the router's input queue with a high rate of malformed packets, which effectively blocks legitimate routing traffic. The result is a control plane DoS against the router. The next reason is for layer 2 keepalive, IGP, and BGP traffic separation.

Large BGP tables generate considerably large updates. These updates could potentially block the input queue for some time, preventing the router from receiving keepalive packets or IGP updates. This may result in IGP adjacency flapping or layer 2 link loss reports. The third reason is due to issues with process switching.

If for some reason CEF is disabled, the IP INPUT process can result in regular IP traffic blocking the input queues, causing a loss of the control plane. SPD prevents this through input drops.

So how does SPD work? First, the input queue consists of two parts. One part is the regular hold queue, which is visible through **`show interface`** command, and the other part is the priority queue, which stores routing updates and keepalives. The processor serves the priority queue first until it is empty, and then switches to the regular hold-queue. Additionally, the priority queue consists of two parts, the SPD Headroom and the SPD Extended Headroom. The Extended Headroom queue is emptied before the SPD Headroom in a priority manner. Specifically input packets are queued as follows.

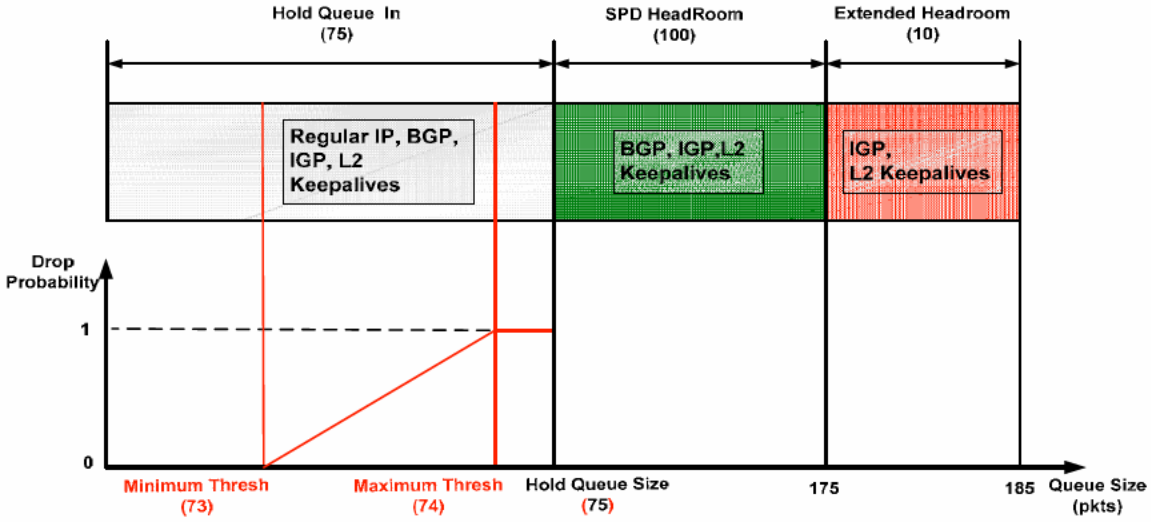
Copyright © 2010 Internetwork Expert

Layer 2 keepalives and IGP packets go to the SPD Extended Headroom. If there is no space available in the SPD Extended Headroom, packets go to the SPD Headroom. As a last resort, if both the Extended Headroom and Headroom are full, these packets go to the regular Hold Queue. BGP updates go directly to SPD Headroom. If the SPD Headroom is full, BGP packets hit the Hold Queue. All other IP packets (malformed or process-switched) go to the Hold Queue. The result is that L2 Keepalive/IGP packets are serviced first, BGP next, and other packets last.

Although the Hold Queue is FIFO, it uses the RED drop procedure. Two thresholds (*Minimum* and *Maximum*) set for hold queue define the random drop behavior. If the current hold queue length is less than the *Minimum Threshold*, packets are never dropped. If the queue length grows beyond *Minimum*, but is less than *Maximum*, every new packet is randomly dropped with the probability proportional to queue depth:

Prob = (*QueueDepth – MinimumThresh*)/(*MaximumThresh-MinimumThresh*)

If the queue depth is above *Maximum Threshold*, SPD drops every new incoming packet. See the figure below for an illustration using the SPD default values:



Note the important fact that SPD thresholds are global for all queues. SPD computes *Min* and *Max* thresholds based on the lowest hold-queue size in the system. Therefore if you set the hold queue size lower on some interfaces, you will affect all other interface drop thresholds.

Copyright © 2010 Internetwork Expert

Finally, SPD has two modes of operation, normal and aggressive. They differ in their treatment of malformed packets (packets that require the router to generate ICMP responses). When SPD is set for normal mode (the default) it treats malformed packets as it would all regular IP packets. It places them in the hold queue, subject to random drop. However, in aggressive mode, the malformed packets are dropped as soon as the hold queue grows above the minimum threshold. Effectively, SPD Aggressive mode replaces the random drop for malformed packets with an unconditional drop. SPD configuration can be verified as follows.

```
Rack1R1#show interface fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  Hardware is AmdFE, address is 000d.659b.9140 (bia 000d.659b.9140)
  Internet address is 155.1.146.1/24
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex, 100Mb/s, 100BaseTX/FX
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:00, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 3/150/0/0 (size/max/drops/flushes); Total output drops:
0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
<snip>

Rack1R1#show ip spd
Current mode: normal.
Queue min/max thresholds: 75/150, Headroom: 120, Extended Headroom: 150
IP normal queue: 2, priority queue: 0.
SPD special drop mode: aggressively drop bad packets
```

Copyright © 2010 Internetwork Expert

10.11 Payload Compression on Serial Links

- Configure PPP on the link between R1 and R3 using Predictor compression.
- Enable Stacker compression on the HDLC link between R2 and R3.
- Enable FRF.9 compression on the Frame Relay PVC between R2 and R5.

Configuration

```
R1:
interface Serial0/1
 encapsulation ppp
 compress predictor

R2:
interface Serial0/1
 encapsulation ppp
 compress stac
!
interface Serial0/0
 frame-relay map ip 155.1.0.5 205 broadcast IETF payload-compression
FRF9 stac one-way-negotiation

R3:
interface Serial1/2
 encapsulation ppp
 compress predictor
!
interface Serial 1/3
 encapsulation ppp
 compress stac

R5:
interface Serial0/0/0
 frame-relay map ip 155.1.0.2 502 broadcast IETF payload-compression
FRF9 stac one-way-negotiation
```

Copyright © 2010 Internetwork Expert

Verification

Note

Payload compression algorithms try to increase the available bandwidth on a link by removing redundant information present in data packets. Payload compression does not modify packet headers, unlike TCP or RTP header compression, but instead only applies to the packet's contents. Cisco IOS supports a variety of compression algorithms on serial links.

Some of these algorithms are proprietary (e.g. Frame-Relay packet-by-packet compression) and some are standard (e.g. FRF.9). Most of the compression procedures use the Lempel-Ziv algorithm and its descendants (e.g. LZS – Lempel-Ziv-Stac). LZ "Stacker" compression removes redundancy in the payload by replacing repetitive occurrences of the same strings with an index linking back to a dictionary of the most common strings. This procedure is memory efficient, yet consumes many CPU cycles. This is why compression is only allowed on slow serial links (HDLC, PPP, Frame-Relay, up to 2Mbps) and often routers use hardware accelerators to offload compression from their central processors. The "Predictor" compression algorithm uses a minimum of CPU cycles on the router, but needs more memory compared to the others. Instead of looking for repetitive substrings, it tries to predict every upcoming character in the flow, using the prediction tables common to both sending and receiving parties. The algorithm is very fast, but less efficient compared to the LZ family.

Some encapsulations support only a single compression procedure, such as Cisco's HDLC sole support for the Cisco proprietary "Stacker" implementation of LZS. The only encapsulation supporting the "Predictor" algorithm is PPP, and it is less effective than the LZ group of algorithms. Frame Relay supports per-VC compression through the **`frame-relay map`** command, either with the Cisco proprietary packet compression based on the LZS algorithm, or the standard FRF.9 compression. Note that using FRF.9 automatically changes the Frame Relay encapsulation to IETF from the default of "Cisco".

A key question that now arises is how compression interacts with other QoS features. Do traffic shapers and policers take compressed or uncompressed packet sizes into account? With Legacy QoS features (e.g. CQ, GTS, CAR) the byte counters and packet rates are based on uncompressed packet sizes. However, after the compression stage, the system informs the QoS feature about the actual packet size being sent on the line, and credits the respective queue.

Copyright © 2010 Internetwork Expert

With MQC QoS features such as LLQ or CBWFQ, packets are compressed before computing the current burst or queue weight. Thus you can specify parameters based on compressed packet sizes, taking into account the compression effects.

Finally keep in mind that compression is only effective on redundant data streams, such as clear text. Sending already compressed payloads, such as PDFs, MP3s, GZIPs, or encrypted information will not yield any performance benefits.

To verify this configuration generate ICMP packets with highly redundant data payloads and view the compression statistics.

```
Rack1R1#show interfaces serial 0/1
Serial0/1 is up, line protocol is up
  Hardware is PowerQUICC Serial
  Internet address is 155.1.13.1/24
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation PPP, LCP Open
  Open: IPCP, CCP, CDPCP, loopback not set
  Keepalive set (10 sec)
  Last input 00:00:07, output 00:00:00, output hang never
  Last clearing of "show interface" counters 00:00:50
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: weighted fair
<snip>
```

Note that PPP negotiated the CCP phase (compression control protocol).

```
Rack1R1#show compress
 Serial0/1
         Software compression enabled
         uncompressed bytes xmt/rcv 0/0
         compressed bytes   xmt/rcv 0/0
         Compressed bytes sent:         0 bytes   0 Kbits/sec
         Compressed bytes recv:         0 bytes   0 Kbits/sec
         1  min avg ratio xmt/rcv 0.000/0.000
         5  min avg ratio xmt/rcv 0.000/0.000
         10 min avg ratio xmt/rcv 0.000/0.000
         no bufs xmt 0 no bufs rcv 0
         resyncs 0

Rack1R1#ping 155.1.23.3 size 1000 repeat 100 data AAAA

Type escape sequence to abort.
Sending 100, 1000-byte ICMP Echos to 155.1.23.3, timeout is 2 seconds:
Packet has data pattern 0xAAAA
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 52/55/112 ms
```

```
Rack1R1#show compress
 Serial0/1
         Software compression enabled
         uncompressed bytes xmt/rcv 100526/100526
         compressed bytes   xmt/rcv 14461/14472
         Compressed bytes sent:     14461 bytes   1 Kbits/sec  ratio: 6.951
         Compressed bytes recv:     14472 bytes   1 Kbits/sec  ratio: 6.946
         1  min avg ratio xmt/rcv 5.466/5.442
         5  min avg ratio xmt/rcv 5.466/5.442
         10 min avg ratio xmt/rcv 5.466/5.442
         no bufs xmt 0 no bufs rcv 0
         resyncs 0
```

Note the average RTT of 55ms. The link speed is 128Kbps, so a 1000 byte uncompressed packet would take approximately 125ms to serialize. Now verify the "Stacker" compression in sending packets from R2 to R3 and R5.

```
Rack1R2#ping 155.1.0.5 size 1000 repeat 100 data ABAB

Type escape sequence to abort.
Sending 100, 1000-byte ICMP Echos to 155.1.0.5, timeout is 2 seconds:
Packet has data pattern 0xABAB
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 44/49/176 ms

Rack1R2#ping 155.1.23.3 size 1000 repeat 100 data ABAB

Type escape sequence to abort.
Sending 100, 1000-byte ICMP Echos to 155.1.23.3, timeout is 2 seconds:
Packet has data pattern 0xABAB
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 24/27/36 ms

Rack1R2#show compress
 Serial0/0 - DLCI: 205
         Software compression enabled
         uncompressed bytes xmt/rcv 101048/101744
         compressed bytes   xmt/rcv 6303/6591
         Compressed bytes sent:      6303 bytes   1 Kbits/sec  ratio: 16.031
         Compressed bytes recv:      6591 bytes   2 Kbits/sec  ratio: 15.436
         1  min avg ratio xmt/rcv 1.044/0.753
         5  min avg ratio xmt/rcv 1.044/0.753
         10 min avg ratio xmt/rcv 1.044/0.753
         no bufs xmt 0 no bufs rcv 0
         resyncs 0
         Additional Stac Stats:
         Transmit bytes:  Uncompressed =        0 Compressed =       5582
         Received bytes:  Compressed =       5863 Uncompressed =        0
```

Copyright © 2010 Internetwork Expert

```
Serial0/1
         Software compression enabled
         uncompressed bytes xmt/rcv 100996/102076
         compressed bytes   xmt/rcv 7553/7994
         Compressed bytes sent:      7553 bytes   2 Kbits/sec  ratio: 13.371
         Compressed bytes recv:      7994 bytes   2 Kbits/sec  ratio: 12.769
         1  min avg ratio xmt/rcv 12.566/12.017
         5  min avg ratio xmt/rcv 12.566/12.017
         10 min avg ratio xmt/rcv 12.566/12.017
         no bufs xmt 0 no bufs rcv 0
         resyncs 0
         Additional Stac Stats:
         Transmit bytes:  Uncompressed =       60 Compressed =       7553
         Received bytes:  Compressed =       8422 Uncompressed =        0
```

```
Rack1R2#show frame-relay map
Serial0/0 (up): ip 155.1.0.1 dlci 205(0xCD,0x30D0), static,
              CISCO, status defined, active
Serial0/0 (up): ip 155.1.0.3 dlci 205(0xCD,0x30D0), static,
              CISCO, status defined, active
Serial0/0 (up): ip 155.1.0.4 dlci 205(0xCD,0x30D0), static,
              CISCO, status defined, active
Serial0/0 (up): ip 155.1.0.5 dlci 205(0xCD,0x30D0), static,
              broadcast,
              IETF, status defined, active
              Payload Compression FRF9
```

Note that just one PVC on the Frame-Relay interface is affected by compression.

Copyright © 2010 Internetwork Expert

10.12 Generic TCP/UDP Header Compression

- Optimize the bandwidth usage between R4 and R5 by reducing TCP and RTP header sizes.
- Allow for a maximum of 16 concurrent RTP and TCP sessions.
- R5 should optimize traffic only if it detects already optimized traffic from R4.

Configuration

```
R4:
interface Serial0/1/0
 ip tcp header-compression
 ip tcp compression-connections 32
 ip rtp header-compression
 ip rtp compression-connections 32

R5:
interface Serial0/1/0
 ip tcp header-compression passive
 ip tcp compression-connections 32
 ip rtp header-compression passive
 ip rtp compression-connections 32
```

Verification

Note

The idea behind header compression is that interactive applications and voice packets usually have a very small payload size compared to the IP/TCP/UDP/RTP header size. For example a telnet packet has 1 or 2 bytes of payload per key press, and VoIP has a 20 byte payload for a G.729 packet. On slow links (up to 768Kbps) the serialization delay for these packets can be significantly reduced if the header information is somehow reduced, and only the differences between sequential headers are sent across the link.

Header compression works between two directly connected routers, on a hop-by-hop basis. For each TCP or RTP data stream special contexts are created on each side containing the header information. The actual TCP or RTP packet is sent with the original header stripped, and carries just a small header of around 4 to 5 bytes depending on the protocol and its options (e.g. preservation of checksums). The receiving side selects the local context based on the small header, and then recreates the original header using the stored information plus the difference vector received. Periodically the router sends the full headers to refresh the contexts on both sides.

Copyright © 2010 Internetwork Expert

Cisco IOS allows for using different formats of header compression, as well as limiting the number of local compression contexts (flows). Additionally it is possible to configure the router for passive compression, which means that it only starts packet compression when it receives compressed information. Note that since each VoIP/TCP connection is usually bi-directional, it requires 2 contexts on every side (sending and receiving).

TCP header compression can be verified using telnet streams as follows.

```
Rack1R4#show ip rtp header-compression
RTP/UDP/IP header compression statistics:
  Interface Serial0/1/0 (compression on, Cisco, RTP)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 32 rx slots, 32 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 32 free contexts

Rack1R4#show ip tcp header-compression
TCP/IP header compression statistics:
  Interface Serial0/1/0 (compression on, VJ)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 32 rx slots, 32 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 32 free contexts
```

Open TCP connections from SW2 and SW4 through R4's Serial connection to R5.

```
Rack1SW2#telnet 155.1.45.4
Trying 155.1.45.4 ... Open


User Access Verification

Password: cisco
Rack1R4>en
Password: cisco

Rack1SW4#telnet 155.1.146.6
Trying 155.1.146.6 ... Open


User Access Verification

Password: cisco
Rack1R6>en
Password: cisco
```

Copyright © 2010 Internetwork Expert

Check the connection contexts.

```
Rack1R4#show ip tcp header-compression serial 0/1/0 detail
TCP/IP header compression statistics:
Configured:
 Max Header 168 Bytes, Max Time 5 Secs, Max Period 256 Packets, Feedback On
Negotiated:
 Max Header 168 Bytes, Max Time 5 Secs, Max Period 256 Packets, Feedback On
TX contexts:

RX contexts:

  Connection 0:
   IP source: 155.1.108.10, IP destination: 155.1.146.6
   TCP source: 11003, TCP destination: 23
   Last packet   received is VJcompressed-tcp and last sequence received is 0

  Connection 1:
   IP source: 155.1.58.8, IP destination: 155.1.45.4
   TCP source: 11008, TCP destination: 23
   Last packet   received is VJcompressed-tcp and last sequence received is 0

Rack1R5#show ip tcp header-compression serial 0/1/0 detail
TCP/IP header compression statistics:
Configured:
 Max Header 168 Bytes, Max Time 5 Secs, Max Period 256 Packets, Feedback On
Negotiated:
 Max Header 168 Bytes, Max Time 5 Secs, Max Period 256 Packets, Feedback On
TX contexts:

RX contexts:

  Connection 0:
   IP source: 155.1.146.6, IP destination: 155.1.108.10
   TCP source: 23, TCP destination: 11003
   Last packet   received is VJcompressed-tcp and last sequence received is 0

  Connection 1:
   IP source: 155.1.45.4, IP destination: 155.1.58.8
   TCP source: 23, TCP destination: 11008
   Last packet   received is VJcompressed-tcp and last sequence received is 0
Rack1R5#
```

Copyright © 2010 Internetwork Expert

Check the compression statistics.

```
Rack1R4#show ip tcp header-compression
    TCP/IP header compression statistics:
      Interface Serial0/1/0 (compression on, VJ)
        Rcvd:    253 total, 238 compressed, 0 errors, 0 status msgs
                 0 dropped, 0 buffer copies, 0 buffer failures
        Sent:    182 total, 168 compressed, 0 status msgs, 0 not predicted
                 5690 bytes saved, 3910 bytes sent
                 2.45 efficiency improvement factor
        Connect: 32 rx slots, 32 tx slots,
                 14 misses, 0 collisions, 0 negative cache hits, 32 free
    contexts
                 92% hit ratio, five minute miss rate 0 misses/sec, 0 max

Rack1R5#show ip tcp header-compression
TCP/IP header compression statistics:
  Interface Serial0/1/0 (compression on, VJ, passive)
    Rcvd:    182 total, 168 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    253 total, 238 compressed, 0 status msgs, 0 not predicted
             8312 bytes saved, 2117 bytes sent
             4.92 efficiency improvement factor
    Connect: 32 rx slots, 32 tx slots,
             15 misses, 2 collisions, 0 negative cache hits, 32 free contexts
             94% hit ratio, five minute miss rate 0 misses/sec, 0 max
```

Copyright © 2010 Internetwork Expert

10.13 MLP Link Fragmentation and Interleaving

- Enable PPP encapsulation on the serial link between R4 and R5.
- Configure PPP Multilink with interleaving on this connection.
- Ensure the maximum serialization delay is 10ms.
- Assume the bandwidth of the link is 128Kbps.

Configuration

```
R4:
interface Virtual-Template1
 bandwidth 128
 ip address 155.1.45.4 255.255.255.0
 fair-queue
 ppp multilink fragment delay 10
 ppp multilink interleave
!
multilink virtual-template 1
!
interface Serial0/1/0
 bandwidth 128
 no ip address
 encapsulation ppp
 load-interval 30
 ppp multilink

R5:
interface Virtual-Template1
 bandwidth 128
 ip address 155.1.45.5 255.255.255.0
 fair-queue
 ppp multilink fragment delay 10
 ppp multilink interleave
!
multilink virtual-template 1
!
interface Serial0/1/0
 bandwidth 128
 no ip address
 encapsulation ppp
 load-interval 30
 ppp multilink
 clock rate 128000
```

Copyright © 2010 Internetwork Expert

Verification

Note

The PPP multilink feature allows the binding of multiple physical interfaces into a single logical interface. Multilink PPP is usually used to load balance at layer 2, as opposed to layer 3. For example if two T1 links are provisioned between two routers, layer 3 load balancing occurs by inserting parallel routes in the routing table, or though some method such as EIGRP variance or BGP multipath. With PPP Multilink these T1's can be treated as one logical layer 3 link, removing the need for additional layer 3 control plane overhead. The multilink procedure shrinks outgoing packets into pieces, assigns them sequence numbers, and forwards them down the parallel physical links. Based on the physical interface bandwidth some links forward more fragments than others. Therefore this behavior can be used as a QoS solution in addition to a load balancing solution.

Consider the following problem: a company uses a slow link (less than T1 rate) to transfer data and VoIP traffic. VoIP packets are small (around 60 bytes) and the general goal of the network is to provide as little delay as possible when sending these packets across the link (e.g. 10ms). This is certainly possible, as it does not take that much time to serialize 60 bytes across the WAN link. The problem arises, however, when large data packets (e.g. 1500 bytes) and voice packets share the same link, as the large data packets may block the transmit ring of the line for a considerable length of time delaying the transmission of voice packets. This may result is delays much higher than 10ms for small voice packets. The solution to this problem consists of two steps.

First, split the large packets into smaller pieces so that each piece does not take more than 10ms to serialize across the slow link. Secondly, interleave the fragments of the large packets with small voice packets, so a large "train" of data fragments will not block the line before voice packets.

Different implementations of this idea exist, such as solving the problem at layer 3 versus layer 2. A layer 3 solution would be to reduce the IP MTU on the serial link, as seen in previous examples, which causes the router to fragment large packets at the IP layer. The interface software queue (e.g. WFQ) then interleaves the large fragments with the small voice packets. This problem with this solution is that excessive fragmentation of IP packets occurs, which is highly undesirable due to performance and security reasons. An alternate solution is to solve the problem with layer 2 fragmentation.

Copyright © 2010 Internetwork Expert

With layer 2 fragmentation, fragments only exist on the point-to-point layer 2 link between two routers in the transit path, such a PPP link or Frame Relay PVC, as opposed to layer 3 fragmentation which occurs end-to-end. Multilink PPP with Link Fragmentation and Interleaving (MLP LFI) is one of these layer 2 schemes, and is accomplished as follows.

First, run PPP multilink on a **single** physical link (no real bundling) to activate fragmentation. Secondly, do not fragment IP packets smaller than a defined size (e.g. 60 bytes) but send them between fragments of large packets. The problem with this solution is that only works across one physical link. The sending of non--fragmented packets interleaved with PPP fragments across multiple links results in the receiver getting packets out of order, and losing sequence number synchronization. These out of order fragments are simply dropped, since there is no reliable re-sending procedure in MLP. The following requirements are necessary for MLP LFI to work effectively.

First, a software interface queue is needed that prioritizes voice (small) packets. This could be either simple WFQ, or some more advanced solution, like LLQ or IP RTP Priority. Next, an interleaving queue is needed that sits between the software queue (e.g. WFQ) and physical link queue (FIFO). This queue has two parts – high priority and normal priority. Small, non-fragmented packets use the high priority part, while large packet fragments go to the normal part. Using this queue the scheduler ensures that a group of data fragments never blocks any voice packet.

The interleaving queue is not visible when using MLP LFI, but it does exist. The fragment size for MLP is calculated based on two factors, the logical MLP interface bandwidth represented by the **`bandwidth`** command, and the fragment delay in milliseconds represented by the **`ppp multilink fragment delay`** command. Effectively the *fragment_size* = *bandwidth*delay*.

It is important to remember that fragment size should be based upon the physical rate of the link (e.g. clock rate) not the logical bandwidth (e.g. PVC CIR). In the case of MLP Interleaving it is irrelevant, since PPP does not support logical circuits, but it's important when fragmenting packets over Frame Relay with PPP.

To solve the problem of multilink fragmentation and interleaving over multiple physical links at the same time, a standards based solution called Multiclass Multilink PPP is available. It allows MLP to keep track of traffic "classes" running across the MLP bundle. Each class has its own sequence numbering, independent of other classes. This permits encapsulating voice packets using fragmentation headers, while the receiving side can still recover the proper sequence numbers with interleaved packets.

Copyright © 2010 Internetwork Expert

For this particular task there are multiple ways to configure MLP LFI. In this case we use a virtual-template interface to group the physical link, however we could also use either a Multilink interface or a Dialer Profile. Note that this solution also uses WFQ via the **fair-queue** command to provide more expedited treatment to small voice packets, and enable interleaving with the secondary queue.

To verify the MLP LFI configuration, first shut down the Frame-Relay interface of R5 to ensure all packets from VLAN 146 take the path across the Serial link between R4 and R5. Next, configure R6 to send SLA Jitter probes to R5 using G.729 sized packets, and configure R5 to respond to those probes. Lastly, configure R1 as an HTTP server and start transferring the IOS image from R1 to SW2 across the Serial link between R4 and R5.

```
R1:
ip http server
ip http path flash:

R5:
ip sla responder
!
interface Serial 0/0/0
 shutdown

R6:
ip sla monitor 1
 type jitter dest-ipaddr 155.1.45.5 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
ip sla monitor schedule 1 life forever start-time now


Rack1SW2#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-
mz.124-24.T.bin null:
Loading http://admin:cisco@155.1.146.1/c2600-adventerprisek9-mz.124-
10.bin !!!!!!
```

Copyright © 2010 Internetwork Expert

Verify the status of the multilink bundle.

```
Rack1R4#show ppp multilink

Virtual-Access2, bundle name is Rack1R5
  Endpoint discriminator is Rack1R5
  Bundle up for 00:13:44, total bandwidth 128, load 241/255
  Receive buffer limit 12192 bytes, frag timeout 1000 ms
  Interleaving enabled
    0/0 fragments/bytes in reassembly list
    0 lost fragments, 0 reordered
    0/0 discarded fragments/bytes, 0 lost received
    0x10F51 received sequence, 0x18912 sent sequence
  Member links: 1 (max not set, min not set)
    Se0/1, since 00:13:44, 160 weight, 152 frag size
No inactive multilink interfaces

Rack1R4#sh interface virtual-access 2
Virtual-Access2 is up, line protocol is up
  Hardware is Virtual Access interface
  Internet address is 155.1.45.4/24
  MTU 1500 bytes, BW 128 Kbit, DLY 100000 usec,
     reliability 255/255, txload 155/255, rxload 77/255
  Encapsulation PPP, LCP Open, multilink Open
  Open: IPCP
  MLP Bundle vaccess, cloned from Virtual-Template1
  Vaccess status 0x40, loopback not set
  Keepalive set (10 sec)
  DTR is pulsed for 5 seconds on reset
  Last input 00:00:06, output never, output hang never
  Last clearing of "show interface" counters 00:05:57
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: weighted fair
  Output queue: 6/1000/64/0/8445 (size/max
total/threshold/drops/interleaves)
     Conversations  2/3/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 96 kilobits/sec
  5 minute input rate 39000 bits/sec, 61 packets/sec
  5 minute output rate 78000 bits/sec, 51 packets/sec
     20645 packets input, 1154494 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     16389 packets output, 3549759 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
```

Note the size of the WFQ queue and the number of interleaves. Two conversations currently occupy the WFQ.

Copyright © 2010 Internetwork Expert

Now note the member (physical link) queue status. The queue is FIFO, and the original queueing method is not effective.

```
Rack1R4#show interface serial 0/1/0
Serial0/1/0 is up, line protocol is up
  Hardware is PowerQUICC Serial
  MTU 1500 bytes, BW 128 Kbit, DLY 20000 usec,
     reliability 255/255, txload 241/255, rxload 83/255
  Encapsulation PPP, LCP Open, multilink Open
  Link is a member of Multilink bundle Virtual-Access2, loopback not
set
  Keepalive set (10 sec)
  Last input 00:00:00, output 00:00:00, output hang never
  Last clearing of "show interface" counters 00:29:58
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 3
  Queueing strategy: weighted fair  [suspended, using FIFO]
  FIFO output queue 0/10, 3 drops
  30 second input rate 42000 bits/sec, 87 packets/sec
  30 second output rate 121000 bits/sec, 126 packets/sec
     99239 packets input, 6112709 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     9 input errors, 0 CRC, 9 frame, 0 overrun, 0 ignored, 0 abort
     143778 packets output, 17191470 bytes, 0 underruns
     0 output errors, 0 collisions, 16 interface resets
     0 output buffer failures, 0 output buffers swapped out
     22 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

Look at the WFQ contents on R4. Note the packet length for the voice conversation is 62 bytes (60 bytes plus 2 for PPP encapsulation) and the HTTP conversation is 578 bytes (576 bytes plus 2 for PPP). Also note the number of "Interleaves" next to the HTTP conversation.

```
Rack1R4#show queueing interface virtual-access 2
Interface Virtual-Access2 queueing strategy: fair
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: weighted fair
  Output queue: 7/1000/64/0/9079 (size/max total/threshold/drops/interleaves)
     Conversations  2/3/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 96 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 1/32384/0/0/0
  Conversation 79, linktype: ip, length: 62
  source: 155.1.146.1, destination: 155.1.45.5, id: 0x01A1, ttl: 254,
  TOS: 0 prot: 17, source port 54312, destination port 16384

  (depth/weight/total drops/no-buffer drops/interleaves) 6/32384/0/0/9010
  Conversation 137, linktype: ip, length: 578
  source: 155.1.146.6, destination: 155.1.58.8, id: 0x7397, ttl: 254,
  TOS: 0 prot: 6, source port 80, destination port 11004
```

Copyright © 2010 Internetwork Expert

Now verify the IP SLA statistics at R6. Note that a large packet would take approximately 576*8/128000=36ms to serialize (plus the delay to reach R4 from R6 and additional timing overhead) without MLP LFI. At the same time, the average measured RTT is just 23ms, thanks to fragmentation and interleaving on R4.

```
Rack1R1#show ip sla monitor statistics 1
Round trip time (RTT)   Index 1
        Latest RTT: 23 ms
Latest operation start time: 05:59:38.896 UTC Thu Aug 14 2008
Latest operation return code: OK
RTT Values
        Number Of RTT: 1000
        RTT Min/Avg/Max: 14/23/50 ms
Latency one-way time milliseconds
        Number of one-way Samples: 0
        Source to Destination one way Min/Avg/Max: 0/0/0 ms
        Destination to Source one way Min/Avg/Max: 0/0/0 ms
Jitter time milliseconds
        Number of Jitter Samples: 999
        Source to Destination Jitter Min/Avg/Max: 1/7/35 ms
        Destination to Source Jitter Min/Avg/Max: 1/1/6 ms
Packet Loss Values
        Loss Source to Destination: 0           Loss Destination to Source: 0
        Out Of Sequence: 0      Tail Drop: 0    Packet Late Arrival: 0
Voice Score Values
        Calculated Planning Impairment Factor (ICPIF): 11
MOS score: 4.06
Number of successes: 16
Number of failures: 1
Operation time to live: Forever
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.14 Legacy Generic Traffic Shaping

- Configure shaping on R6 to limit the rate of packets going towards R4's Loopback0 via the connection to VLAN 146 to 128Kbps.
- Use a shaping interval of 10ms and disable the extended burst functionality.
- Limit the shaper's queue size to 1024 packets.

Configuration

```
R6:
access-list 104 permit ip any 150.1.4.0 0.0.0.255
!
interface FastEthernet 0/0.146
 traffic-shape group 104 128000 1280 0 1024
```

Verification

Note

Legacy Generic Traffic Shaping (GTS) was the first feature to limit the outbound traffic rate of an interface introduced into IOS. With GTS all traffic leaving an interface/subinterface can be shaped, or a subset of traffic can be classified using an access-list. The syntax is as follows.

traffic-shape rate *<CIR> <Bc> <Be> <QueueLimit>*
traffic-shape group *<ACL> <CIR> <Bc> <Be> <QueueLimit>*

<*QueueLimit*> sets the maximum size of the WFQ buffer space. WFQ is the built-in traffic-shaping queue scheduler. Additionally the system can inspect multiple traffic-shape statements on the interface in the order configured to classify the outgoing traffic in a top-down fashion. The additional arguments of the traffic-shape statement are explored in detail below.

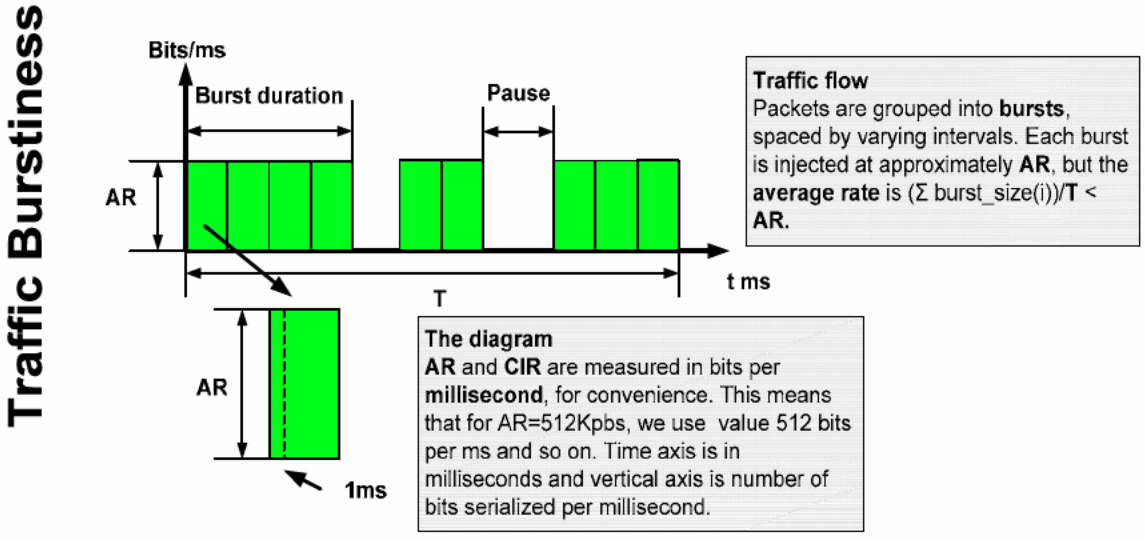
The purpose of traffic shaping is to "format" an outbound packet flow so that it conforms to a traffic contract. The formatting process slows down the average bitrate and the packet flow structure, resulting in a traffic flow consisting of uniformly spaced traffic bursts. Service provider traffic contracts are usually verified using ingress policers, such as Frame Relay Traffic Policing. The burst values used in outbound customer edge shaping typically should match the inbound metering function used on the service provider edge. This input metering function is explored further in the sections relating to policing with CAR.

Copyright © 2010 Internetwork Expert

An example when traffic shaping is needed is the case that a customer's interface's physical rate, the Access Rate (AR), is higher than the contracted rate guaranteed by the service provider. For example a customer buys a Frame Relay circuit provisioned at 10Mbps, but the physical link to the provider is DS3 (45Mbps). Since the customer's interface always serializes packets outbound at 45Mbps, and the service provider performs traffic policing/admission control inbound, shaping is needed on the customer side to slow the average output rate down to 10Mbps.

Another case is when a remote site's connection speed is lower than the local site's. This means that the local side may overwhelm the remote side's connection by sending packets faster than the other side can accept. An example of this is in a WAN hub-and-spoke design, where the total sending rates of the spokes combined exceed the rate at which the hub can receive. By shaping the spoke sites down, the hub's site is not oversubscribed.

In both situations mentioned above, the physical AR is greater than the desired output rate. To slow the rate down, the first task of the shaper is to meter the traffic coming into the output queue, and decide whether it exceeds the target average rate. The concept of metering is based on the fact that traffic leaves an interface in a serial manner (bit by bit, packet by packet), and that packets are usually grouped in bursts, separated by periods of interface silence, as illustrated in the figure below.



While the router sends each burst at AR speed, the spacing between bursts makes the average rate less than the AR. The goal of metering is to mark those bursts that exceed (do not conform to) the desired average rate, called the Committed Information Rate (CIR).

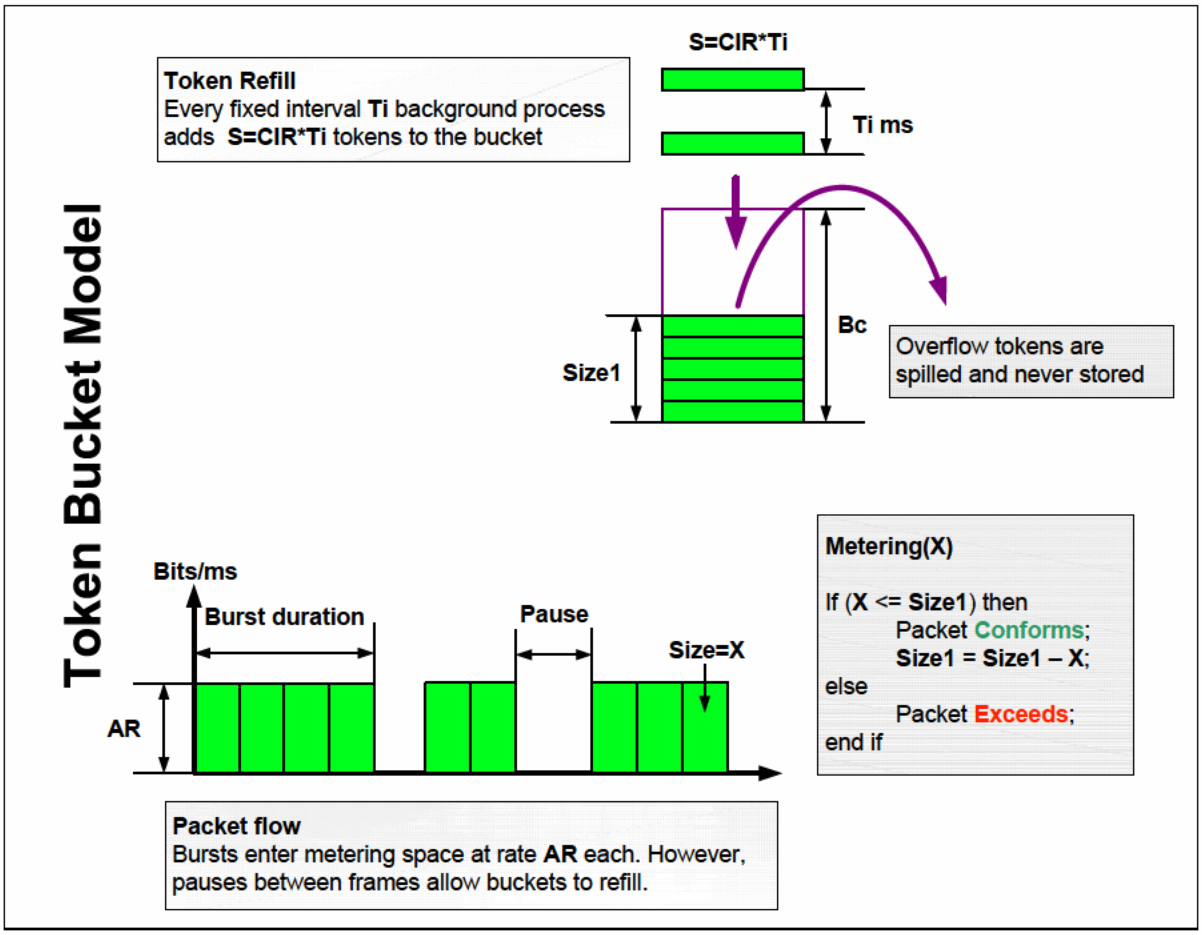
Copyright © 2010 Internetwork Expert

Note that for the purpose of metering, the shaper's "CIR" value does not relate to the service provider's "CIR" value, where the latter actually means the "guaranteed" rate of the circuit based on the SLA. Instead, "CIR" in the context of shaping simply means the target average rate.

The metering function of traffic shaping uses what is known as a *token bucket* model to determine if traffic conforms to, or exceeds, the average rate. Every time a packet tries to be dequeued to the transmit ring, the metering function compares the size of the packet trying to leave to the amount of tokens, or credit, in the token bucket. If the size of the packet is less than or equal to the amount of credit, the packet conforms and is sent. If the size of packet is greater than the amount of credit in the token bucket, the packet exceeds, and is delayed.

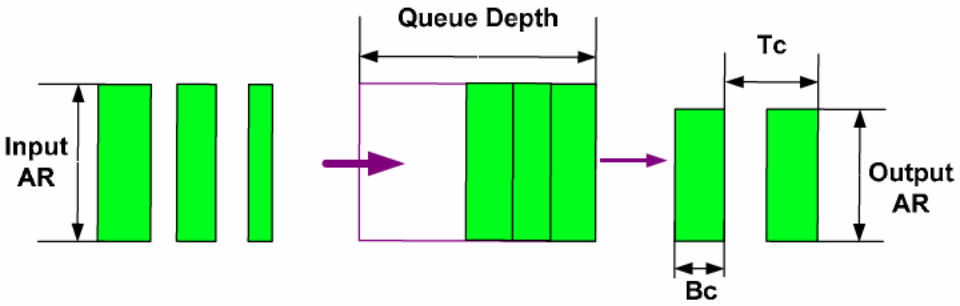
The size of the token bucket is calculated by taking the desired average rate (CIR) in bits per second, and breaking it down into a smaller value of burst in bits per interval in milliseconds. These values are expressed as B_c (Burst Committed) bits, and T_c (Time Committed) milliseconds. The size of the token bucket is B_c bits, and the system refills the token bucket at a rate of B_c bits per T_c milliseconds. The key point here is that B_c bits per T_c interval is the same value as CIR bits per second, but is simply expressed in smaller units.

Copyright © 2010 Internetwork Expert


Token Bucket Model
Token Refill
Every fixed interval Ti background process adds S=CIR*Ti tokens to the bucket
S=CIR*Ti
Ti ms
Bc
Size1
Overflow tokens are spilled and never stored
Bits/ms
Burst duration
Pause
Size=X
AR
Metering(X)
If (X <= Size1) then
Packet Conforms;
Size1 = Size1 – X;
else
Packet Exceeds;
end if
Packet flow
Bursts enter metering space at rate AR each. However, pauses between frames allow buckets to refill.

When a packet conforms to the average rate per interval it is dequeued to the transmit ring, and the number of tokens in the bucket equal to the size of the packet sent in bits is deducted. If packet exceeds because there are not enough tokens in the bucket, the shaping process delays the packets and holds it in the internal shaping queue. By this logic even though traffic is always sent at AR, the periods of delay incurred by non-conforming traffic in the shaping queue result in the overall average rate (CIR) being lower than the AR. This method is known the *leaky bucket* algorithm.

Leaky Bucket Model



Leaky Bucket Process
The process queues packets coming at rate **input AR**. Every **Tc** interval scheduler runs and dequeues up to **Bc** amount of bytes and sends them at output **AR**.

Since the shaper sends packets in bursts every T_c milliseconds, the outgoing traffic "shape" becomes uniform, meaning that packet bursts of almost the same size are separated by T_c intervals. The size of T_c is configured indirectly by configuring the CIR and the B_c values based on the formula $B_c = CIR * T_c/1000$. Note that most documentation for traffic shaping denotes this as $B_c = CIR*T_c$, which is true as long as T_c is expressed in *seconds*, not *milliseconds*, which it is not by default.

The minimal value for T_c is platform and version dependent, and is generally 10ms. This limits the scheduler "precision", but also puts a limit on the CPU utilization. Most importantly though, the value of T_c sets the minimum delay between packet bursts to 10ms. This generally limits the use of traffic shaping to WAN connections, since using it across a LAN connection may introduce unacceptably high delay. This point is explored in more detail within the context of priority queueing and traffic shaping simultaneously.

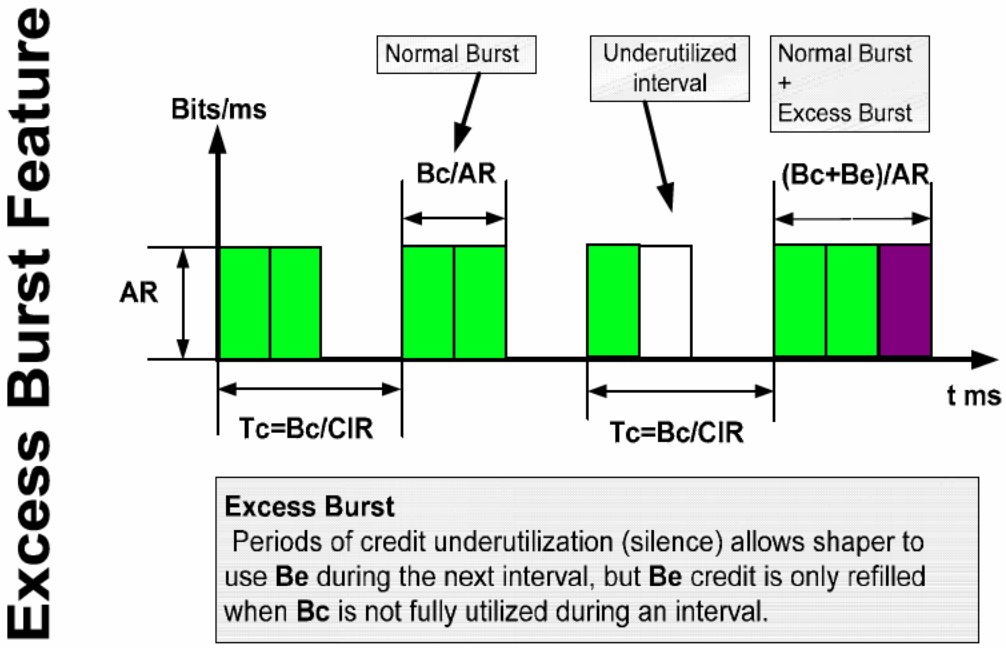
Copyright © 2010 Internetwork Expert

One possible problem with the above calculation for B_c is the case that the packet trying to be dequeued is larger than B_c, which means that there would never be enough credit in the token bucket to send it. For example if a packet's size is 1500 bytes, but the B_c is only 1000 bytes. To deal with this situation the shaper calculates a deficit counter (e.g. 1000-1500=-500) and adds this counter to the accumulated credit in the next round. In effect this reduces the amount of traffic to send the next time around. With this method although the scheduler sends a packet every time the accumulated credit is above zero, the accumulated credit will eventually drop to zero and the scheduler will have to wait 2 T_c intervals to accumulate credit and send the next packet. This may result in an average sending rate per T_c to be greater than or less than CIR, but the average rate over a longer period of time still never exceeds CIR. To avoid this problem altogether ensure that B_c is greater than the average packet size, which will achieve a smoother packet distribution. This is not always possible though, since there are cases when CIR value is too low. In the latter case layer 2 fragmentation can be introduced, which is covered in more detail in later tasks.

The next problem case that the scheduler can run into is when it has no traffic to send during a time interval (e.g. a pause in the packet stream), but it has more than B_c bytes to send in the next time interval. Based on the leaky token bucket algorithm, no more than B_c bytes can be sent per T_c interval, even if in previous intervals it did not send enough traffic. The result of this is that the shaper achieves *less* than the desired average rate. To resolve this problem traffic shaping uses what is known as a *dual leaky token bucket*, with the first token bucket represented as Committed Burst (B_c) and the second token bucket as Excess Burst (B_e).

The Excess Burst bucket is only filled in the case that the full B_c bucket was not emptied in the previous interval. The extra credit left over from the B_c bucket is then moved to the B_e bucket before B_c is refilled. For example if the B_c size is 10 bits, but only 8 bits were sent in the current interval, a credit of 2 bits can be moved to the B_e bucket if space is available. During the next interval the scheduler can now de-queue up to B_c+B_e bits. If B_c capacity is again not used completely, the left over credit is moved to B_e, up to its maximum size.

Copyright © 2010 Internetwork Expert



Like B_c, B_e has some finite size defined which controls how much credit can be stored. The size of the B_e bucket is constrained by the Access Rate of the physical link, since the packets are always serialized at this rate. Therefore the maximum B_e value (*maxB_e*) is equal to (AR-CIR)*T_c/1000 which implies that if the shaper sends B_c+maxB_e per T_c, it is sending at the Access Rate. The B_e value can be set lower than maxB_e, but should never exceed maxB_e. Note that since B_e is only populated due to a lack of B_c being used, the average sending rate over time still never exceeds the CIR. By default, with legacy GTS, if you do not specify a B_e value it is equal to B_c.

GTS configuration can be verified as follows.

```
Rack1R6#show traffic-shape

Interface   Fa0/0.146
       Access Target     Byte   Sustain    Excess     Interval  Increment Adapt
VC     List   Rate       Limit  bits/int   bits/int   (ms)      (bytes)   Active
-      104    128000     160    1280       0          10        160       -
```

In the above output the *Target Rate* is CIR in bps, *Byte Limit* is B_c+B_e in bytes – the maximum amount of data that could be sent during an interval - *Sustain bits/int* is B_c in bits, *Excess Bits/int* is B_e in bits, *Interval* is T_c in ms, and *Increment* is the B_c in bytes – the number of bytes that could be sent every regular interval.

Copyright © 2010 Internetwork Expert

To see the traffic-shaper in action, configure R6 to send IP SLA jitter probes to R4's Loopback0 interface, and configure R4 to respond to the probes. Additionally configure R4 to meter the incoming traffic destined for the Loopback0 interface via the MQC.

```
R6:
ip sla monitor 1
 type jitter dest-ipaddr 150.1.4.4 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
!
ip sla monitor schedule 1 life forever start-time now

R4:
ip sla responder
!
access-list 120 permit ip any 150.1.4.0 0.0.0.255
!
class-map match-all TRAFFIC_TO_LO0
 match access-group 120
!
policy-map METER
 class TRAFFIC_TO_LO0
!
interface FastEthernet 0/1
 service-policy input METER
```

Now source ICMP ping packets from SW1 and SW3 to R4's Loopback0 IP address.

```
Rack1SW1#ping 150.1.4.4 repeat 1000000 size 160 timeout 1

Type escape sequence to abort.
Sending 1000000, 160-byte ICMP Echos to 150.1.4.4, timeout is 1
seconds:
!!!
```

Now view the shaper queue with the **show traffic-shape queue** command. Note the weights assigned to each flow – they are the same since all flows use an IP precedence of 0. Also note the queue size, which is now 1024, the buffer limit we set. The system determines the maximum number of flow queues (conversations) automatically based on the CIR value provided for the shaper. This limits the traffic-shape command's usage for high-speed shapers, since automatic WFQ does not scale well over 2Mbps sending rates.

```
Rack1R6#show traffic-shape queue fastEthernet 0/0.146
Traffic queued in shaping queue on FastEthernet0/0.146
 Traffic shape group: 104
  Queueing strategy: weighted fair
  Queueing Stats: 3/1024/64/0 (size/max total/threshold/drops)
     Conversations  3/3/16 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 128 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 1/32384/0/0/0
  Conversation 9, linktype: ip, length: 78
  source: 155.1.146.6, destination: 150.1.4.4, id: 0x02EC, ttl: 255,
  TOS: 0 prot: 17, source port 53779, destination port 16384

  (depth/weight/total drops/no-buffer drops/interleaves) 1/32384/0/0/0
  Conversation 6, linktype: ip, length: 178
  source: 155.1.67.7, destination: 150.1.4.4, id: 0xEEB4, ttl: 254,
prot: 1

  (depth/weight/total drops/no-buffer drops/interleaves) 1/32384/0/0/0
  Conversation 4, linktype: ip, length: 178
  source: 155.1.79.9, destination: 150.1.4.4, id: 0xFD33, ttl: 253,
prot: 1

Rack1R6#show traffic-shape statistics
                 Acc. Queue Packets   Bytes     Packets   Bytes     Shaping
I/F              List Depth                     Delayed   Delayed   Active
Fa0/0.146          104 2     1530365   216942630 1496672   212005056 yes
```

Note the packet sizes in the above output. Conversation 9 is 78 bytes long, which is our 60-byte "voice" packets plus the 18 byte overhead for an Ethernet header with a VLAN tag. The ICMP packets are 160 bytes plus the 18 bytes for Ethernet. This implies that WFQ uses the L2 packet sizes to calculate scheduling times and implement queue disciplines.

Copyright © 2010 Internetwork Expert

Now verify the input rate at R4.

```
Rack1R4#show policy-map interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy input: METER

    Class-map: TRAFFIC_TO_LO0 (match-all)
      915439 packets, 125028746 bytes
      30 second offered rate 123000 bps
      Match: access-group 120

    Class-map: class-default (match-any)
      1768 packets, 522792 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any.
```

The input rate is slightly less than the configured target rate of 128Kbps due to the layer 2 overhead taken into consideration by WFQ, but not by the MQC, and a slightly uneven packet distribution in the shaped packet stream.

Copyright © 2010 Internetwork Expert

10.15 Legacy CAR for Admission Control

- Configure R4 to mark traffic received from R1's IP address 155.X.146.1 as it enters R4's connection to VLAN 146 using legacy committed access rate.
- Traffic up to 256Kbps should be marked with an IP precedence of 1.
- Exceeding traffic should be marked with an IP precedence of 0.
- Use an average traffic burst size of 4000 bytes.

Configuration

```
R4:
!
access-list 111 permit ip host 155.1.146.1 any
!
interface FastEthernet0/1
 rate-limit input access-group 111 256000 4000 4000 conform-action set-
prec-transmit 1 exceed-action set-prec-transmit 0
```

Verification

Note

The Legacy Committed Access Rate (CAR) feature, or rate-limiting, was designed for two purposes - admission control (e.g. packet remarking), and traffic limiting (e.g. policing) at the network edge. Unlike traffic shaping, rate limiting can be configured as both an input and output feature, and it does not buffer (delay) exceeding traffic bursts. Traffic classification for CAR can be based on all traffic on the interface, a standard or extended IP access-list, a MAC access-list (**`access-list rate-limit [0-99]`**), or an IP Precedence access-list (**`access-list rate-limit [100-199]`**).

Multiple rate-limit statements can be configured sequentially on the same interface, using the **`continue`** keyword, allowing for a hierarchical structure, such as the following:

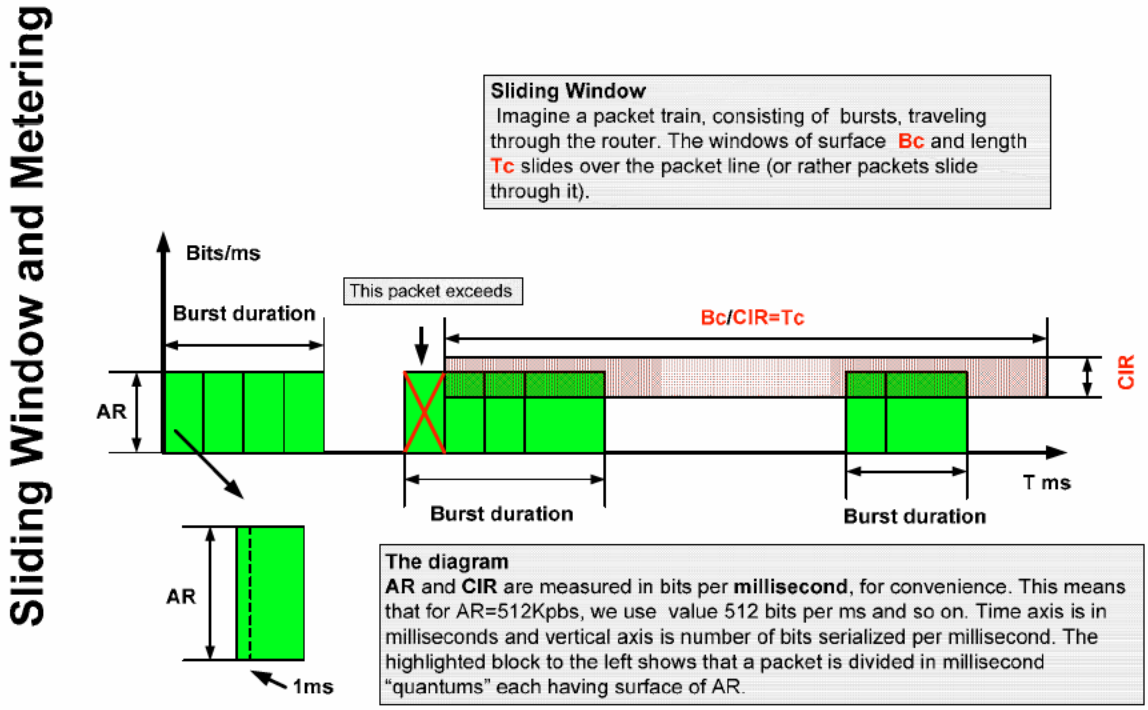
```
rate-limit input access-group 100 128000 8000 8000 conform-action transmit
exceed action continue
rate-limit input 256000 8000 8000 conform-action transmit exceed action drop
```

Configuration statements are evaluated sequentially until the first match is found and the specified action is executed. The **`continue`** keyword instructs the router to follow to the next statement in list, which makes the order significant.

Copyright © 2010 Internetwork Expert

Admission control through CAR is similar to how traffic-shaping's algorithm works, but with some key differences. Consider the situation in which a customer's edge device connects to the service provider via an interface with a physical Access Rate (AR) that is higher than the contracted rate (the provider's CIR). In order to enforce the contract, the provider needs to meter the input traffic's bitrate, and mark (dropping is also a form of marking) the packets based on the measured rate. Like in shaping the idea of metering is based on the fact that network traffic enters the interface in a serial manner (bit by bit, packet by packet), and that packets are usually grouped in bursts, separated by "islands" of networking interface silence. While the router receives each burst at AR, the spacing between bursts makes average input rate less than AR. The goal of metering is marking those bursts that conform or exceed the contracted average rate (CIR). For example, if two shortly separated bursts enter the router at AR speed, the average rate measured over those two bursts would be close to AR and much higher than CIR, and thus the second burst would most likely be exceeding.

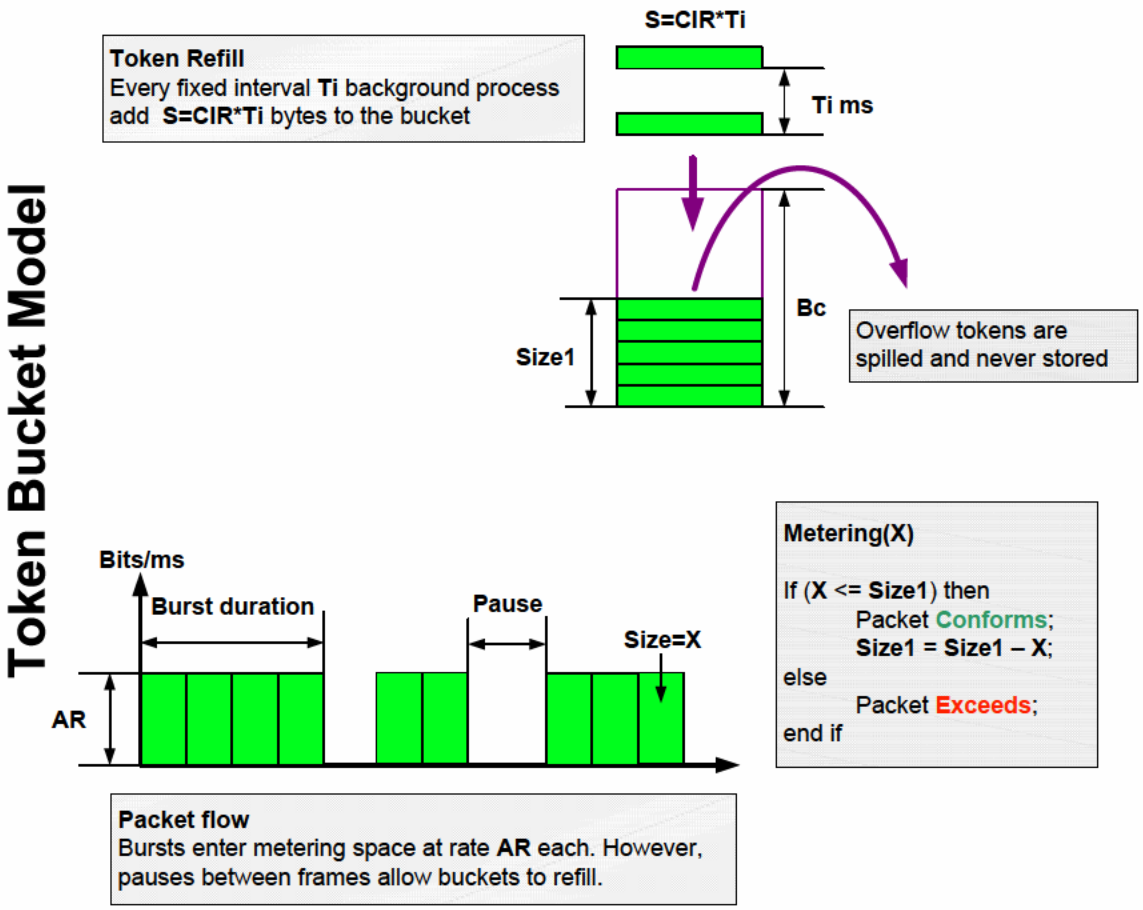
To measure the average speed, metering uses a sliding "averaging time interval" (T_c), not to be confused with traffic shaping's T_c. The process considers a new incoming packet as conforming if the amount of traffic already received during the current T_c, plus the size of the new packet, is less than or equal to Committed Burst (B_c). The interval T_c is sliding in the sense that it moves across the packet line as packets enter the router. The larger the T_c value, the greater amount of averaging that is performed over the input packet rate.



Copyright © 2010 Internetwork Expert

Note that like in traffic shaping, T_c, B_c and CIR values are not independent, since B_c=CIR*T_c. CIR is enforced by the fact that during any interval T_c, the amount of conforming traffic is no more than the B_c, and thus the average CIR the provider enforces is B_c per T_c. The difference between shaping and policing however, is that shaping produces packet bursts uniformly separated by interval T_c, each burst of size B_c, while CAR uses a sliding T_c window to measure the current average traffic rate and mark every new packet accordingly.

The metering implementation uses the same token bucket model as shaping, however the process does not delay exceeding packets, but instead simply remarks or drops them.



The token bucket for CAR works as follows. In the background the rate-limiting process adds CIR*T_i bytes to the token bucket ever T_i ms. For traffic shaping T_i = T_c, but for policing T_i is a constant value of about 1/8000 of a second. This is why you can only change the CAR rate in steps of 8000bps. Like shaping, the depth of the bucket is still defined as B_c. When a new packet of size *X* arrives, CAR consults the token bucket to see if it can borrow *X* credits. If there are enough tokens, the packet conforms, the conforming action is executed, and *X* tokens are subtracted from the bucket. If there are not enough tokens in the bucket, the process considers the packet as exceeding, and the exceed action is performed.

The key factor for correct inbound admission control with CAR depends on the burstiness of the traffic. If the source formats traffic in burst sizes that do not match the configured CAR burst, metering may be incorrect (e.g too much traffic is marked as exceeding or conforming). This is why customers connected to service providers should shape their traffic according to traffic contract parameters (CIR, B_c, etc) that match the policing settings on the service provider side.

Finally, similar to GTS, CAR computes the bandwidth used based on full layer 2 packet sizes, including the layer 2 encapsulation overhead.

To test this configuration set R1 to shape to uniform traffic bursts of 4000 bytes every 125ms, and generate packets using an unconstrained ping command with the packet size of 1000 bytes. Based on the below B_c value the traffic shaper will send 4x1000 byte packets every 125ms. In this case, the shaper burst size matches the CAR burst size:

```
R1:
interface FastEthernet0/0
 traffic-shape rate 256000 32000 0 1000

Rack1R1#ping 150.1.4.4 size 1000 timeout 0 repeat 1000000

Type escape sequence to abort.
Sending 1000000, 1000-byte ICMP Echos to 150.1.4.4, timeout is 0
seconds:
 ..............................................................
 ..............................................................
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show interfaces fastEthernet 0/1 rate-limit
FastEthernet0/1
  Input
    matches: access-group 111
      params:  256000 bps, 4000 limit, 4000 extended limit
      conformed 3841 packets, 3885702 bytes; action: set-prec-transmit 1
      exceeded 24 packets, 24336 bytes; action: set-prec-transmit 0
      last packet: 20ms ago, current burst: 3928 bytes
      last cleared 00:02:21 ago, conformed 219000 bps, exceeded 1000 bps
```

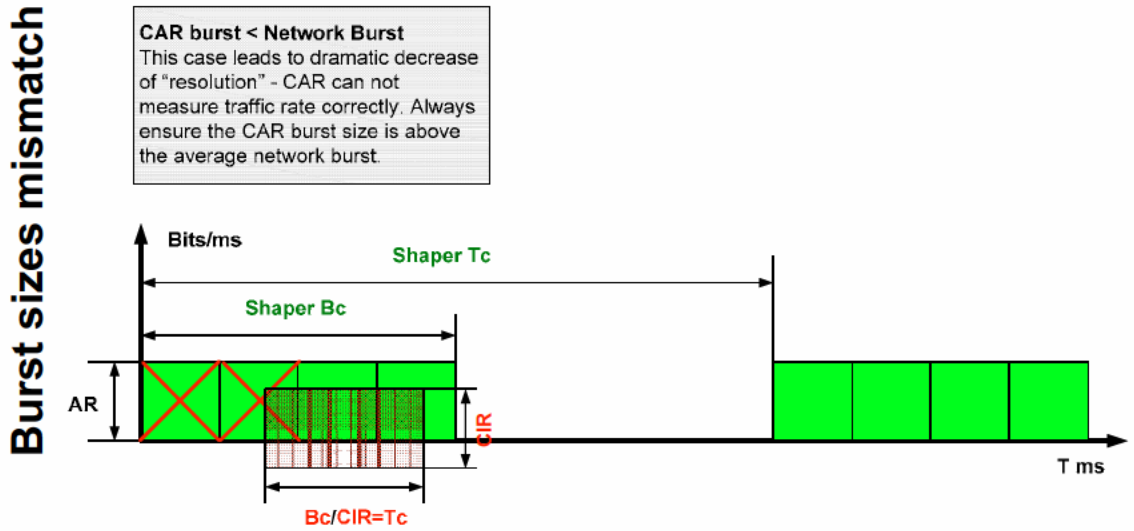
Some packets exceed due to small variations of inter-packet gaps, however since the burst values match on each side CAR isn't very sensate to the small variations.

Now let's see what happens when the CAR burst is set to a value that is *smaller* than the average network burst. To accomplish this change R1's shaper to emit bursts of twice the previous size (8000 bytes=8 packets of 1000 bytes) at the same rate (256Kbps) resulting in a T_c of 250ms. Leave the CAR configuration the same (CIR=256Kbps, B_c=4000bytes, T_c=125ms) on R4 and view the output.

```
R1:
interface FastEthernet0/0
 traffic-shape rate 256000 64000 0 1000

Rack1R1#ping 150.1.4.4 size 1000 timeout 0 repeat 1000000

Type escape sequence to abort.
Sending 1000000, 1000-byte ICMP Echos to 150.1.4.4, timeout is 0
seconds:
 ..............................................................
 ..............................................................
<snip>

Rack1R4#show interfaces fastEthernet 0/1 rate-limit
FastEthernet0/1
  Input
    matches: access-group 111
      params:  256000 bps, 4000 limit, 4000 extended limit
      conformed 1169 packets, 1180182 bytes; action: set-prec-transmit 1
      exceeded 817 packets, 828438 bytes; action: set-prec-transmit 0
      last packet: 232ms ago, current burst: 3928 bytes
      last cleared 00:01:18 ago, conformed 120000 bps, exceeded 84000 bps
```

Copyright © 2010 Internetwork Expert

Although the shaper is sending at the same average rate, its burst size is twice as big as the compared to the CAR burst. This results in incorrect measurements, as CAR can not average across the whole bursts. The following figure illustrates this concept.



The small averaging interval does not allow for a correct rate measurement. It actually makes the CAR algorithm think traffic is being sent faster than the configured average rate. This example demonstrates the importance of picking the correct CAR burst size, which should be greater than the average traffic burst size. Remember though that this only applies to bursty flows. The case of uniform packet density is explored in the tasks relating to rate limiting with CAR.

Copyright © 2010 Internetwork Expert

10.16 Oversubscription with Legacy CAR and WFQ

- Configure WFQ on R4's point-to-point link to R5, and clock the link at 128Kbps.
- Configure a CAR policy on R4 to achieve the following:
  - 64Kbps that R4 receives from R1 and R6 each should be guaranteed on the link out to R5.
  - Traffic from R1 and R6 should be allowed up to 128Kbps each total.
  - Traffic above 128Kbps should be dropped.
  - Use the averaging time interval of 200ms.

Configuration

```
R4:
access-list 111 permit ip host 155.1.146.1 any
access-list 116 permit ip host 155.1.146.6 any
!
interface FastEthernet0/1
 rate-limit input access-group 111 64000 3200 3200 conform-action set-
prec-transmit 1 exceed-action continue
 rate-limit input access-group 111 128000 3200 3200 conform-action set-
prec-transmit 0 exceed-action drop
!
 rate-limit input access-group 116 64000 3200 3200 conform-action set-
prec-transmit 1 exceed-action continue
 rate-limit input access-group 116 128000 3200 3200 conform-action set-
prec-transmit 0 exceed-action drop
!
interface Serial 0/1/0
 fair-queue
 clock rate 128000

R5:
interface Serial 0/1/0
 clock rate 128000
```

Copyright © 2010 Internetwork Expert

Verification

Note

Oversubscription is one of the prominent features of packet switched networks. Statistical multiplexing allows multiple users to share the same resources in dynamic proportions. The idea behind oversubscription is that not all users may need the same resources at the same time, and thus it's possible to grant some users the ability to temporary use more resources than the network can guarantee. This is in clear contrast with time-division multiplexing (TDM) networks, where users cannot exceed the guaranteed share of resources, such as a group of TDM slots.

In packet networks, service providers may sell more bandwidth than they can actually provide. For example a 10Mbps link could be shared between 10 users, with the agreement that each user can pull down up to 10Mbps if nobody else uses the link, but users are only guaranteed 1Mbps of link bandwidth in the case of heavy congestion. The actual implementation of this "downstream" sharing may vary, but most often it utilizes some sort of "fair-queuing" (slow links) or "round-robin" (high-speed links) scheduling, which implements the concept of max-min fairness for resource sharing.

The same idea can apply to traffic sent from users to the network ("upstream" oversubscription). A provider may allow customers to send more traffic than it can actually guarantee to deliver. If the network is not congested, it delivers all oversubscribed traffic. However if there is resource competition, the network delivers only the guaranteed part of the traffic. This brings in two important concepts – Committed Information Rate (CIR) and Peak Information Rate (PIR). (Note that CIR is similar to the concept used in traffic shaping, but it serves a different purpose here). CIR is the guaranteed rate, and PIR is the maximum acceptable rate. Often, the term EIR (Excess Information Rate) is also used, which is defined as EIR=PIR-CIR. To implement this "upstream" oversubscription, the provider uses some sort of traffic metering procedure, for example CAR or policing to mark the incoming traffic. Theoretically, a policer may sit at the ingress edge of the service provider's network, and meter traffic received from users, comparing it with the contracted CIR and PIR values. Note that the PIR value may be optional and the network may just meter against a CIR value. All the traffic marked above CIR is not guaranteed delivery across the network, so it could be dropped somewhere in case of congestion. However, in a carefully planned network, the sum of all incoming CIRs should not exceed the network capacity, and thus guarantee delivery for in-profile (conforming) traffic. As an extreme case, it is even possible for the provider to sell traffic with a CIR of zero, allowing users to send at any possible rate up to AR, but not guaranteeing any delivery.

Copyright © 2010 Internetwork Expert

In this task the oversubscribed resource is the link between R4 and R5. R4 allows its customers (R1 and R6) to send up to a maximum of 128Kbps, which ensures that if one user is not sending, the other can use 100% of the link bandwidth. However, if both users are sending traffic simultaneously, we want them to share the link in equal proportions. To accomplish this, we utilize WFQ on the serial link.

Traffic received from the customers at a rate of 64Kbps or below is marked with an IP precedence of 1. If the traffic rate exceeds 64Kbps, the packets are marked with an IP precedence of 0. Since WFQ shares bandwidth proportional to IP precedence, any customer's "conforming" traffic will preempt the "exceeding" traffic of the other customer. Additionally a second nested rate-limit statement drops traffic from each customer if the rate exceeds 128Kbps.

The burst size is calculated based on a maximum transmit rate of 128Kbps and averaging interval of 200ms, where B_c = 128000/8 *200/1000 = 3200 bytes. Note that CAR takes the burst value in bytes, not bits, so 128000bps is divided by 8 to convert to bytes, and 200ms is divided by 1000 to convert to seconds.

Next consider the case where a customer's traffic causes congestion in the provider's network due to oversubscription. The service provider must have some way to signal the customer to slow down their sending rate. In legacy layer 2 networks, such as Frame Relay or ATM, there exists explicit layer 2 notifications (e.g. FECNs, BECNs) that allow the provider to directly tell customer equipment to slow down the sending rate to CIR. If no explicit congestion mechanisms exist, the provider could rely on upper-level protocol mechanics to respond to implicit congestion notification (e.g. packet delays or drops). An example of such upper layer protocols is TCP, which dynamically responds to packet loss by slowing down the sending rate. However, some upper level protocols do not respond to any implicit congestion notification (e.g. UDP packet floods). Such protocols pose serious problems to oversubscribed networks if no explicit Layer 2 signaling exists.

Based on their congestion behavior, traffic flows are commonly divided into three types: adaptive, aggressive, and sensitive. Adaptive flows, such as TCP, respond to implicit or explicit (e.g. TCP ECN) congestion notification by slowing down the sending rate. Aggressive flows, such as ICMP packet floods, do not respond to packet drops, and may monopolize network resources. Sensitive flows, such as voice traffic, are sensitive to congestion and/or packet drops. The provider should treat them in a special manner, such as using priority queuing.

Copyright © 2010 Internetwork Expert

In general, the provider should not let aggressive flows enter their network at the peak rate. The provider needs to limit such flows at the edge by using an admission procedure. Sensitive flows should also not be oversubscribed. Only the adaptive flows are good candidates for oversubscription, due to their positive congestion response behavior. However, since TCP carries the majority of Internet traffic, oversubscription has become common in modern packet networks.

To verify this particular scenario enable the HTTP server service on R1 and R6, and configure R5 to meter the input traffic from R1 and R6 on its serial link to R4. Set the IP MTU to 576 on the serial interfaces of R4 and R5 to avoid problems with TCP traffic retransmissions and reordering due to fragmentation. Lastly shutdown the Frame Relay interface of R5.

```
R1, R6:
ip http server
ip http path flash:

R4:
interface Serial 0/1/0
 ip mtu 576

R5:
access-list 111 permit ip host 155.1.146.1 any
access-list 116 permit ip host 155.1.146.6 any
!
class-map match-all R1_PREC0
 match access-group 111
 match ip precedence 0
!
class-map match-all R1_PREC1
 match access-group 111
 match ip precedence 1
!
class-map match-all R6_PREC0
 match access-group 116
 match ip precedence 0
!
class-map match-all R6_PREC1
 match access-group 116
 match ip precedence 1
!
no policy-map METER
policy-map METER
 class R1_PREC0
 class R1_PREC1
 class R6_PREC0
 class R6_PREC1
!
interface Serial 0/1/0
 service-policy input METER
 ip mtu 576
!
interface Serial 0/0/0
 shutdown
```

Copyright © 2010 Internetwork Expert

Set SW2 to download an IOS image from R1.

```
Rack1SW2#copy http://admin:cisco@155.1.146.1/c2600-adventerprisek9-
mz.124-10.bin null:
Loading http://admin:cisco@155.1.146.1/c2600-adventerprisek9-mz.124-
10.bin !!!!!!
```

Check the admission statistics on R4. Note that 64Kbps are conforming and almost the same amount is exceeding. R1 effectively uses the total serial link bandwidth.

```
Rack1R4#show interfaces fastEthernet 0/1 rate-limit
FastEthernet0/1
  Input
    matches: access-group 111
      params:  64000 bps, 3200 limit, 3200 extended limit
      conformed 48 packets, 28320 bytes; action: set-prec-transmit 1
      exceeded 46 packets, 27140 bytes; action: continue
      last packet: 24ms ago, current burst: 2874 bytes
      last cleared 00:00:03 ago, conformed 64000 bps, exceeded 61000 bps
    matches: access-group 111
      params:  128000 bps, 3200 limit, 3200 extended limit
      conformed 46 packets, 27140 bytes; action: set-prec-transmit 0
      exceeded 0 packets, 0 bytes; action: drop
      last packet: 28ms ago, current burst: 0 bytes
      last cleared 00:00:03 ago, conformed 61000 bps, exceeded 0 bps
    matches: access-group 116
      params:  64000 bps, 3200 limit, 3200 extended limit
      conformed 1 packets, 226 bytes; action: set-prec-transmit 1
      exceeded 0 packets, 0 bytes; action: continue
      last packet: 2545ms ago, current burst: 0 bytes
      last cleared 00:00:03 ago, conformed 0 bps, exceeded 0 bps
    matches: access-group 116
      params:  128000 bps, 3200 limit, 3200 extended limit
      conformed 0 packets, 0 bytes; action: set-prec-transmit 0
      exceeded 0 packets, 0 bytes; action: drop
      last packet: 235787ms ago, current burst: 0 bytes
      last cleared 00:00:06 ago, conformed 0 bps, exceeded 0 bps
```

Copyright © 2010 Internetwork Expert

Check the metering results on R5. The traffic flow is split almost equally between precedence 0 (exceeding) and precedence 1 (conformed).

```
Rack1R5#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy input: METER

    Class-map: R1_PREC0 (match-all)
      310 packets, 179800 bytes
      30 second offered rate 58000 bps
      Match: access-group 111
      Match: ip precedence 0

    Class-map: R1_PREC1 (match-all)
      316 packets, 183280 bytes
      30 second offered rate 62000 bps
      Match: access-group 111
      Match: ip precedence 1

    Class-map: R6_PREC0 (match-all)
      0 packets, 0 bytes
      30 second offered rate 0 bps
      Match: access-group 116
      Match: ip precedence 0

    Class-map: R6_PREC1 (match-all)
      0 packets, 0 bytes
      30 second offered rate 3000 bps
      Match: access-group 116
      Match: ip precedence 1

    Class-map: class-default (match-any)
      3 packets, 1248 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

Check the WFQ queue contents on R4. There should be one flow with alternating weights, IP Precedence 1 and IP precedence 0 packets hitting the same flow sequentially. Note that weight 16912 corresponds to an IP precedence of 1 and the weight 32384 corresponds to an IP precedence of 0.

```
Rack1R4#show queue serial 0/1/0
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 128
  Queueing strategy: weighted fair
  Output queue: 5/1000/64/128 (size/max total/threshold/drops)
     Conversations  1/3/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 1158 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 5/16192/0/0/0
  Conversation 137, linktype: ip, length: 580
  source: 155.1.146.1, destination: 155.1.58.8, id: 0x5CDA, ttl: 254,
  TOS: 32 prot: 6, source port 80, destination port 11004

Rack1R4#show queue serial 0/1/0
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 128
  Queueing strategy: weighted fair
  Output queue: 5/1000/64/128 (size/max total/threshold/drops)
     Conversations  1/3/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 1158 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 5/32384/0/0/0
  Conversation 137, linktype: ip, length: 580
  source: 155.1.146.1, destination: 155.1.58.8, id: 0x5D77, ttl: 254,
  TOS: 0 prot: 6, source port 80, destination port 11004
```

Now download R6's IOS image from SW4.

```
Rack1SW4#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-
mz.124-24.T.bin null:
Loading http://admin:cisco@155.1.146.6/c2600-adventerprisek9-mz.124-
10.bin !!!!!!
```

Copyright © 2010 Internetwork Expert

Check the admission control on R4. Since TCP is an adaptive protocol, it slowed down both sending rates to 64Kbps because of packet drops in the path. Now both flows are conforming, and no excess traffic is admitted to the network due to TCP's adaptive behavior.

```
Rack1R4#show interfaces fastEthernet 0/1 rate-limit
FastEthernet0/1
  Input
    matches: access-group 111
      params:  64000 bps, 3200 limit, 3200 extended limit
      conformed 403 packets, 236618 bytes; action: set-prec-transmit 1
      exceeded 0 packets, 0 bytes; action: continue
      last packet: 12ms ago, current burst: 834 bytes
      last cleared 00:00:29 ago, conformed 63000 bps, exceeded 0 bps
    matches: access-group 111
      params:  128000 bps, 3200 limit, 3200 extended limit
      conformed 0 packets, 0 bytes; action: set-prec-transmit 0
      exceeded 0 packets, 0 bytes; action: drop
      last packet: 83549ms ago, current burst: 0 bytes
      last cleared 00:00:29 ago, conformed 0 bps, exceeded 0 bps
    matches: access-group 116
      params:  64000 bps, 3200 limit, 3200 extended limit
      conformed 403 packets, 236678 bytes; action: set-prec-transmit 1
      exceeded 0 packets, 0 bytes; action: continue
      last packet: 60ms ago, current burst: 870 bytes
      last cleared 00:00:29 ago, conformed 63000 bps, exceeded 0 bps
    matches: access-group 116
      params:  128000 bps, 3200 limit, 3200 extended limit
      conformed 0 packets, 0 bytes; action: set-prec-transmit 0
      exceeded 0 packets, 0 bytes; action: drop
      last packet: 85100ms ago, current burst: 0 bytes
      last cleared 00:00:32 ago, conformed 0 bps, exceeded 0 bps
```

Copyright © 2010 Internetwork Expert

Check the statistics on R5. We no longer see IP Precedence 0 packets, as sources don't send any traffic above the guaranteed rate.

```
Rack1R5#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy input: METER

    Class-map: R1_PREC0 (match-all)
      67184 packets, 9741533 bytes
      30 second offered rate 0 bps
      Match: access-group 111
      Match: ip precedence 0

    Class-map: R1_PREC1 (match-all)
      102271 packets, 15322824 bytes
      30 second offered rate 62000 bps
      Match: access-group 111
      Match: ip precedence 1

    Class-map: R6_PREC0 (match-all)
      3 packets, 1740 bytes
      30 second offered rate 0 bps
      Match: access-group 116
      Match: ip precedence 0

    Class-map: R6_PREC1 (match-all)
      1875 packets, 1086387 bytes
      30 second offered rate 62000 bps
      Match: access-group 116
      Match: ip precedence 1

    Class-map: class-default (match-any)
      535 packets, 93304 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

Check the WFQ queue on R4. Now we see just two TCP flows with the same weights, corresponding to an IP Precedence value of 1.

```
Rack1R4#show queue serial 0/1/0
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops:
128
  Queueing strategy: weighted fair
  Output queue: 11/1000/64/128 (size/max total/threshold/drops)
     Conversations  2/3/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 1158 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 6/16192/0/0/0
  Conversation 194, linktype: ip, length: 580
  source: 155.1.146.6, destination: 155.1.108.10, id: 0xE2ED, ttl: 254,
  TOS: 32 prot: 6, source port 80, destination port 11004

  (depth/weight/total drops/no-buffer drops/interleaves) 5/16192/0/0/0
  Conversation 137, linktype: ip, length: 580
  source: 155.1.146.1, destination: 155.1.58.8, id: 0x4E8E, ttl: 254,
  TOS: 32 prot: 6, source port 80, destination port 11004
```

Copyright © 2010 Internetwork Expert

10.17 Legacy CAR for Rate Limiting

- Configure R6 to drop traffic received in its link to VLAN 146 in excess of 256Kbps.
- Use a committed burst of 384Kbps and an excess burst of 768Kbps.

Configuration

```
R6:
interface FastEthernet 0/0.146
 rate-limit input 256000 48000 96000 conform-action transmit exceed-
action drop
```

Verification

Note

By using CAR with the exceed action set to "drop" it's possible to limit the rate of both incoming or outgoing traffic. The syntax remains the same, but finding the optimal B_c value becomes a challenge. Also, note that the B_e value with CAR is not used to define the Peak Information Rate (PIR), but rather to implement a pseudo-random "exceed" action. This feature simulates random early detection's drop behavior for TCP flows and prevents flow synchronization.

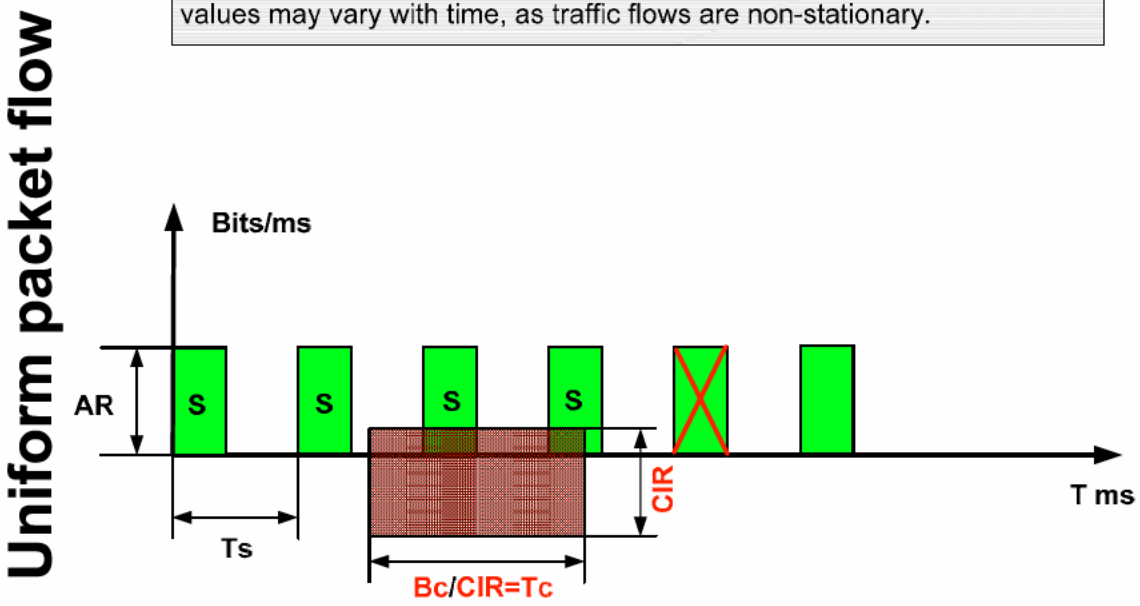
One serious obstacle exists here where policing with CAR though – the metering function assumes burstiness of incoming traffic. Usually a customer will pre-shape outbound traffic to match the inbound metering on the service provider side. As we've seen in the previous tasks, a mismatch in average network burst size and policer burst size can lead to incorrect metering. Based on this observation, how can we accurately estimate the burst size for network traffic if the flow is not already shaped?

Some protocols demonstrate inherent bursty behavior, for example TCP sends traffic in bursts up to the receiver's window size. The receiver's window size for TCP flows tends to be around *Traffic_Rate*RTT*, where *RTT* is the round-trip time for a particular connection. Therefore for a traffic flow made of a single TCP connection, the burst size could be found by multiplying the CIR by the maximum RTT. However, this assumption only holds true for single flow, and only when the RTT is large enough to introduce a significant pause between bursts.

Copyright © 2010 Internetwork Expert

In a practical situation, you cannot expect a flow to be truly bursty. At best it would be reasonable to assume the packet distribution in the flow to be close to uniform. In such situations, when no burst structure exists in a flow, a policer (CAR) would act as a "random" marker. Suppose that a flow sends packets of size *S* every T_s units of time. Then for any policer with parameters CIR and B_c, approximately $S/(CIR*T_s)$ percent of incoming packets will be marked as exceeding, in the case that $S/T_s > CIR$. Note that for uniform flows this percent of exceeding packets does not depend on committed burst size (B_c). However, the marking becomes more accurate with the B_c size growing, due to the large statistical sample. At the same time, a large B_c will inject lengthy packet trains in the network, possibly overfilling queues.

Uniform Traffic flow
The simplest model for traffic flow with average packet size **S** and inter-packet delay of **Ts**. Could be adequate for traffic mix, however S and Ts values may vary with time, as traffic flows are non-stationary.



Another consideration for burst size arises from TCP adaptive behavior. When a packet drop is detected by the TCP stack (via either a timeout or dup-ack) the sending side considers this as a signal of congestion. Different TCP implementations (e.g. Reno, Tahoe, New Reno, Vegas etc.) deal differently with packet loss, but in general, they all reduce the sending rate by some amount. However in real congestion, the sender first expects an increase in the RTT before packet losses, so TCP may adjust more smoothly. Even worse, multiple packet drops in a row may affect *multiple* flows (not just one) and signal them all to slow down. During the recovery phase, all flows will try to increase their sending rate *synchronously*, resulting in a sudden burst and another intense packet drop.

Copyright © 2010 Internetwork Expert

In this manner, multiple flows may "synchronize" and end up with a lower effective bandwidth utilization, due to their mutual influence. Based on those observations, we can see that TCP does not work ideally well with rate-limiting based on CAR, and we can never expect the average sending rate to get close to the configured CIR. However, simply increasing the B_c would make packet drops less "dense", and will allow the TCP sender to recover more quickly.

In practical situations there is no "optimal" value formula, instead the best approach is to find the optimal rate-limit B_c value empirically, starting with some reasonable value. Cisco recommends starting with B_c=CIR*1.5, and testing network performance with a growing window if necessary.

Another CAR enhancement, specifically targeted to rate-limit adaptive TCP flows, is called CAR Excess Burst or B_e. Note that CAR's B_e is totally unrelated to traffic-shaping's B_e, or classic traffic policing's B_e (e.g. as it is used with frame-relay traffic policing). To start, if the B_e value is equal to B_c, then the excess burst is disabled. If $B_e > B_c$ then for every new incoming packet the *probability* of a packet being marked as exceeding is proportional to $(Burst\text{-}B_c)/(B_e\text{-}B_c)$ where *Burst* is the amount of traffic accounted for in the last T_c units of time.

The actual implementation details of CAR's B_e are not important, but the resulting "random" behavior allows packets to be accepted even if the amount of traffic accepted last T_c seconds is greater than B_c. The result is that CAR marks fewer packets as exceeding (dropped) in a row, and distributes drops more evenly across the flow. This feature is specifically useful to avoid TCP flows "global synchronization", and resembles the Random Early Detection congestion avoidance behavior. Cisco recommends setting B_e to $2*B_c$, if the rate-limited flow is based on TCP, but the optimal values can only be found empirically. Note that enabling B_e will increase the amount of packets admitted in your network over B_c during a single T_c, and may result in the measured traffic rate being slightly over CIR.

Copyright © 2010 Internetwork Expert

To verify the effect of using CAR for policing, first disable B_e functionality by letting B_c=B_e. Next enable the HTTP server service on R1 and R4, and start downloading IOS images from those two routers to SW1 and SW3.

```
R1,R4:
ip http server
ip http path flash:

Rack1SW2#copy http://admin:cisco@155.1.146.4/c1841-adventerprisek9-
mz.124-24.T.bin null:
Loading http://admin:cisco@155.1.146.4/ c1841-adventerprisek9-mz.124-
24.T.bin !!!!!!

Rack1SW3#copy http://admin:cisco@155.1.146.1/c2600-adventerprisek9-
mz.124-10.bin null:
Loading http://admin:cisco@155.1.146.1/c2600-adventerprisek9-mz.124-
10.bin !!!!!!
```

Check the rate-limiting statistics on R6.

```
Rack1R6#show interfaces fastEthernet 0/0.146 rate-limit
FastEthernet0/0.146
  Input
    matches: all traffic
      params:  256000 bps, 48000 limit, 48000 extended limit
      conformed 29025 packets, 15419612 bytes; action: transmit
      exceeded 9047 packets, 5333030 bytes; action: drop
      last packet: 0ms ago, current burst: 32213 bytes
      last cleared 00:08:04 ago, conformed 254000 bps, exceeded 87000 bps
```

Note the high number of exceeded packets, almost 23%. That usually means the TCP flows sending rate behaves in a “saw tooth” manner, accelerating over CIR and then rapidly falling down. Now set B_e=2*B_c=96000, download the same files again, and check R6’s statistics.

```
Rack1R6#show interfaces fastEthernet 0/0.146 rate-limit
FastEthernet0/0.146
  Input
    matches: all traffic
      params:  256000 bps, 48000 limit, 96000 extended limit
      conformed 2718 packets, 1501580 bytes; action: transmit
      exceeded 573 packets, 315637 bytes; action: drop
      last packet: 220ms ago, current burst: 66622 bytes
      last cleared 00:00:46 ago, conformed 256000 bps, exceeded 53000 bps
```

Note the amount of exceeding packets is now 17% - over a 6% decrease compared with the case above. In addition, the exceeded packet rate is less than it was before, thanks to the fact that the flows are no longer tightly synchronized.

Copyright © 2010 Internetwork Expert

10.18 Legacy CAR Access-Lists

- Configure R1 to rate limit packets going out its link to VLAN 146 towards the MAC address of R4 to 128Kbps.
- Limit the rate of packets leaving R6 towards SW1 having IP precedence values of one, two, and four to 256Kbps.

Configuration

```
R1:
access-list rate-limit 100 000d.2846.8f21
!
interface FastEthernet 0/0
 rate-limit output access-group rate-limit 100 128000 8000 8000
conform-action transmit exceed-action drop

R6:
access-list rate-limit 0 mask 16
!
interface FastEthernet 0/0.67
 rate-limit output access-group rate-limit 0 128000 8000 8000 conform-
action transmit exceed-action drop
```

Verification

Note

CAR supports two types of special ACLs for rate limiting, MAC-based, and IP precedence based. Each ACL can contain just one line, but IP precedence based ACLs support matching multiple values at the same time by using a special mask format. The idea behind these ACLs is to create high-performance rate-limiting configurations based on just a single-line access-list.

For the MAC based access-list either issue the **`show ip arp`** command on R1 or **`show interface`** command on R4 to find R4's MAC address for the link. For the IP precedence access-list the mask is specified in hex, matching against the precedence values in binary as follows.

[p7][p6][p5][p4][p3][p2][p1][p0]

A binary 1 means to check for the precedence value, where 0 means to ignore (the opposite of a normal access-list wildcard). For example, 0x88 (10001000), mean IP precedence values 7 and 3. For our task precedence values 1, 2, and 4 are represented as 0x16 (00010110).

Copyright © 2010 Internetwork Expert

This configuration can be verified as follows.

```
Rack1R1#ping 155.1.146.4 repeat 1000 size 100 timeout 0

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 155.1.146.4, timeout is 0 seconds:
 ............................................................!.....
....!...!...!. ..................................................
 ................................................................
 ................................................................
<snip>
Success rate is 0 percent (4/1000), round-trip min/avg/max = 1/4/8 ms

Rack1R1#show interfaces fastEthernet 0/0 rate-limit
FastEthernet0/0
  Output
    matches: access-group rate-limit 100
      params:  128000 bps, 8000 limit, 8000 extended limit
      conformed 164 packets, 18696 bytes; action: transmit
      exceeded 836 packets, 95304 bytes; action: drop
      last packet: 3265ms ago, current burst: 7942 bytes
      last cleared 00:09:05 ago, conformed 0 bps, exceeded 1000 bps
```

Note the number of matched packets. Now send the same amount of packets to R6 from R1 and note that the amount of matched packets did not change.

```
Rack1R1#ping 155.1.146.6 repeat 1000 size 100 timeout 0

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 155.1.146.6, timeout is 0 seconds:
 ........................................!.......................
 ............................................................!
<snip>
Success rate is 2 percent (25/1000), round-trip min/avg/max = 1/2/8 ms

Rack1R1#show interfaces fastEthernet 0/0 rate-limit
FastEthernet0/0
  Output
    matches: access-group rate-limit 100
      params:  128000 bps, 8000 limit, 8000 extended limit
      conformed 164 packets, 18696 bytes; action: transmit
      exceeded 836 packets, 95304 bytes; action: drop
      last packet: 118875ms ago, current burst: 7942 bytes
      last cleared 00:11:00 ago, conformed 0 bps, exceeded 1000 bps
```

Copyright © 2010 Internetwork Expert

Source a flood of packets from R6 to SW1 with the default IP precedence of 0.

```
Rack1R6#ping 155.1.67.7 repeat 1000 timeout 0

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 155.1.67.7, timeout is 0 seconds:
!.!. ....!.!.!...!.!.!.!.!....!.!.!.....!.....!.!.....!.!....!.!...
..!!.!...!.!.!...!.......!.!.....!.!.!..!.!.!.!....!.!.....!..!.....
<snip>
Success rate is 25 percent (251/1000), round-trip min/avg/max = 1/2/5 ms

Rack1R6#show interface fastEthernet 0/0.67 rate-limit
FastEthernet0/0.67
  Output
    matches: access-group rate-limit 0
      params:  128000 bps, 8000 limit, 8000 extended limit
      conformed 0 packets, 0 bytes; action: transmit
      exceeded 0 packets, 0 bytes; action: drop
      last packet: 43193693ms ago, current burst: 0 bytes
      last cleared 00:07:15 ago, conformed 0 bps, exceeded 0 bps
```

Note that the show command demonstrates no matches for the rate-limit. Now send packets marked with IP precedence 2 (TOS byte 64).

```
Rack1R6#ping
Protocol [ip]:
Target IP address: 155.1.67.7
Repeat count [5]: 1000
Datagram size [100]:
Timeout in seconds [2]: 0
Extended commands [n]: y
Source address or interface:
Type of service [0]: 64
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 155.1.67.7, timeout is 0 seconds:
.!.!.....!..!...!.!.!.!.!.!.!.....!...!.........!.!.........!.!.!.!
..!.!.!.....!...!.......!.!.!.....................................
<snip>
Success rate is 4 percent (44/1000), round-trip min/avg/max = 1/2/4 ms

Rack1R6#show interfaces fastEthernet 0/0.67 rate-limit
FastEthernet0/0.67
  Output
    matches: access-group rate-limit 0
      params:  128000 bps, 8000 limit, 8000 extended limit
      conformed 181 packets, 21358 bytes; action: transmit
      exceeded 819 packets, 96642 bytes; action: drop
      last packet: 11599ms ago, current burst: 7976 bytes
      last cleared 00:11:05 ago, conformed 0 bps, exceeded 1000 bps
```

Copyright © 2010 Internetwork Expert

Now generate packets with IP precedence 4 (TOS 128).

```
Rack1R6#ping
Protocol [ip]:
Target IP address: 155.1.67.7
Repeat count [5]: 1000
Datagram size [100]:
Timeout in seconds [2]: 0
Extended commands [n]: y
Source address or interface:
Type of service [0]: 128
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 155.1.67.7, timeout is 0 seconds:
..!..!. ........!.!.......!.!.!.!.!.....!.!....!..!.!.....!.!.....
...!.!.!.....!.!.!. ..................................!.............
<snip>
Success rate is 3 percent (38/1000), round-trip min/avg/max = 1/2/4 ms

Rack1R6#show interfaces fastEthernet 0/0.67 rate-limit
FastEthernet0/0.67
  Output
    matches: access-group rate-limit 0
      params:  128000 bps, 8000 limit, 8000 extended limit
      conformed 362 packets, 42716 bytes; action: transmit
      exceeded 1638 packets, 193284 bytes; action: drop
      last packet: 6242ms ago, current burst: 7976 bytes
      last cleared 00:15:39 ago, conformed 0 bps, exceeded 1000 bps
```

Copyright © 2010 Internetwork Expert

10.19 Legacy GTS for Frame Relay

- R3's Frame Relay service provider has agreed to transport up to 128Kbps of incoming traffic, but all traffic that exceeds 96Kbps is marked as Discard Eligible.
- Configure R3 with GTS to slow down to the guaranteed rate if the service provider signals congestion with BECN frames.
- Use 30ms for the traffic-shaping interval.

Configuration

```
R3:
interface Serial 1/0
 traffic-shape rate 128000 3840 0
 traffic-shape adaptive 96000
```

Verification

Note

Frame Relay service providers often practice oversubscription – the selling of more bandwidth than can actually be provided for by the network. This feature is typically allowed by a statistically multiplexed packet network architecture. The general idea is that all users are unlikely to use the maximum bandwidth at the same time, and thus most of the time resources are underutilized. By marking the oversubscribed traffic as low priority at admission, the Frame Relay network can signal QoS mechanics inside the cloud to start dropping oversubscribed traffic first, before the legitimate guaranteed traffic. In this manner the network can offer guarantees and oversubscriptions at the same time.

Frame Relay networks support this design through a combination of markings and explicit congestion notifications. Typically the provider marks frames sent at a rate above the contracted CIR with the Discard Eligible (DE) bit set as they enter the provider network. In case of network congestion the Frame Relay switches drop the DE-marked packets first before other frames. Additionally when congestion occurs, the Frame Relay network can signal the Customer Edge (CE) devices to slow down their sending rates through the Backward Explicit Congestion Notification (BECN), and the Forward Explicit Congestion Notification (FECN) bits in the Frame Relay header. On a congested link, the Frame Relay switch marks all frames going towards the destination with the FECN bit, and frames traveling back towards the sender with the BECN bit. The CE devices can then respond to these notifications through the support of *Adaptive* Frame Relay Traffic Shaping. Specifically this process happens on the provider side as follows.

Copyright © 2010 Internetwork Expert

First, traffic received from the customer edge is subject to admission control through Frame Relay Traffic Policing (FRTP). The Service Provider Edge (PE) equipment meters the input traffic bursts against the configured policer. If the measured traffic rate is within the guaranteed rate (CIR), the PE admits the burst. If the measured rate is higher than CIR, but less than the Peak Information Rate (PIR), which is calculated as the CIR plus the Excessive Information Rate (EIR), the PE admits the burst, but marks the packet with the DE bit. This oversubscribed traffic is allowed in, but has no guarantee of delivery. Finally if the measured burst rate is above PIR, the burst is dropped.

Note that EIR is strictly a service provider side parameter, and it has nothing to do with excessive bursting at the customer side (defined by the customer's B_e shaping value). Excessive bursting is occasional, and only happens when the sending router has accumulated spare credits. Although the provider classifies excessive burst or traffic sent over the contract rate and marks it as DE, the client equipment does not send excessive burst constantly. Sending over the CIR, or sending at maximum allowed PIR=CIR+EIR rate, implies that the customer has their average rate (the shaper's CIR) set above the guaranteed rate.

As previously mentioned when congestion occurs inside the service provider network the switches start discarding DE-marked frames first, and start sending FECN towards the traffic destination and BECN towards the traffic source. The Legacy GTS shaper can respond to the BECN through the implementation of the *minCIR*. minCIR defines the rate which the shaper shrinks down its current B_c towards, while leaving the T_c value the same, when a BECN is received. The effect is that the normal CIR slows down to configured minCIR, or until BECNs stop being received. This feature is configured with GTS via the command **`traffic-shape adaptive <minCIR>`**. FECN adaptation is supported through a feature known as BECN reflection.

FECN adapt is useful in situations when the traffic flow over the link is unidirectional, such as a video broadcast server. Due to the absence of a return traffic flow, the Frame Relay network cannot piggyback BECNs in packets returning to the source, but the destination CE device still received packets marked with FECNs. When the receiver CE enabled FECN adapt, it generates its own dummy test frame with the BECN set back towards the source DLCI, effectively "reflecting" a BECN back. In this manner the receiver informs the unidirectional sender about congestion in the network. This feature is configured with GTS via the command **`traffic-shape fecn-adapt`**.

Copyright © 2010 Internetwork Expert

Although GTS can be applied to a Frame Relay encapsulated interface, as seen in this task, this configuration is technically not *Frame Relay Traffic Shaping* (FRTS). The main reason why is that GTS does not allow the attaching of a shaper to a particular PVC, only applying it to an interface or subinterface. Thus all active PVCs on an interface are shaped the same. True FRTS supports the attaching of different policies to different PVCs through a function known as *Per-VC Queueing*, which is explored in depth in later sections.

Another limitation of GTS applied to a Frame Relay interface is that with adaptive shaping, the reception of a BECN frame on any active PVC results in the shaper slowing down its sending rate for all VCs. Furthermore, the queueing technique for GTS is limited to WFQ, and does not allow using more advanced queueing techniques with Frame-Relay VCs.

In this particular task the service provider instructs the customer that the guaranteed rate (minCIR) is 96Kbps, and the desired average rate (CIR) is 128Kbps. Using the desired T_c value of 30ms we obtain the B_c value of B_c = CIR * T_c/1000 = 128000 * 30/1000 = 3840 bits. We don't set the B_e value since any traffic above 128Kbps is simply dropped by the provider.

Copyright © 2010 Internetwork Expert

To verify the traffic shaping settings send a barrage of ping packets from SW1 to R5 and verify the resulting statistics. Note that the *VC* field is NULL, since FRTS Per-VC Queueing is not supported through GTS.

```
Rack1SW1#ping 155.1.0.5 repeat 1000000 timeout 0

Type escape sequence to abort.
Sending 1000000, 100-byte ICMP Echos to 155.1.0.5, timeout is 0
seconds:
 ..............................................................
 ..............................................................
 ..............................................................
```

```
Rack1R3#show traffic-shape

Interface   Se1/0
       Access Target    Byte   Sustain   Excess    Interval  Increment Adapt
VC     List   Rate      Limit  bits/int  bits/int  (ms)      (bytes)   Active
-             128000    480    3840      0         30        480       BECN
```

```
Rack1R3#show traffic-shape statistics
                 Acc. Queue Packets   Bytes     Packets   Bytes     Shaping
I/F              List Depth                     Delayed   Delayed   Active
Se1/0                 43    2517      906392    1891      684636    yes
```

```
Rack1R3#show traffic-shape queue serial 1/0
Traffic queued in shaping queue on Serial1/0
  Queueing strategy: weighted fair
  Queueing Stats: 21/1000/64/40994 (size/max total/threshold/drops)
     Conversations  1/3/16 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 128 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 21/32384/11009/0/0
  Conversation 10, linktype: ip, length: 104
  source: 155.1.37.7, destination: 155.1.0.5, id: 0x22B4, ttl: 254, prot: 1
```

Note the high number of packet drops due to buffer overflows. Also note that the packet length is 104, meaning that a 100 byte IP packet has a 4 byte Frame-Relay header.

Copyright © 2010 Internetwork Expert

10.20 Legacy Frame Relay Traffic Shaping

- R2, R3, and R5's physical connection rates to the Frame Relay network are 128Kbps, 384Kbps, and 1536Kbps respectively.
- R3's contracted rate is 256Kbps, but the service provider allows occasional packet bursts up to the physical line capacity without discarding them; the provider does not allow sustained oversubscription by R3.
- Configure Legacy Frame Relay Traffic Shaping on R3 and R5 to meet the following requirements:
    - Ensure that R5 does not overwhelm R2's or R3's physical connections.
    - Ensure that R3 adheres to its service contract.
    - Use a Tc value of 10ms to support a VoIP deployment planned in the near future.

Configuration

```
R3:
map-class frame-relay TO_R5
 frame-relay cir 256000
 frame-relay bc 2560
 frame-relay be 1280
!
interface Serial 1/0
 frame-relay traffic-shaping
 frame-relay interface-dlci 305
  class TO_R5

R5:
map-class frame-relay TO_R2
 frame-relay cir 128000
 frame-relay bc 1280
 frame-relay be 0
!
map-class frame-relay TO_R3
 frame-relay cir 256000
 frame-relay bc 2560
 frame-relay be 0
!
interface Serial 0/0/0
 frame-relay traffic-shaping
 frame-relay interface-dlci 503
  class TO_R3
 frame-relay interface-dlci 502
  class TO_R2
```

Copyright © 2010 Internetwork Expert

Verification

Note

Legacy Frame Relay Traffic Shaping (FRTS) was intended as a replacement for legacy GTS through the support of Per-VC Queuing. When the command **`frame-relay traffic-shaping`** is applied on the main Serial interface, a separate queue is created per Virtual Circuit. Note that once enabled, a CIR of 56Kbps and a T_c of 125ms applies to all active PVCs, and should be adjusted accordingly.

Shaping parameters for FRTS are configured using **`map-class frame-relay`** command, which is then attached to a particular PVC with the **`class <name>`** command or to the interface with the **`frame-relay class <name>`** command. When applied to the VC level the map-class applies just to that VC, but when applied to the interface level the map-class applies to all VCs on the interface, or if it is the main interface, all VCs at the main interface and on any subinterfaces.

Legacy FRTS implements three levels of queueing, the per-VC queues, which are the logical queues of the per-VC shapers, the main interface FIFO queue, where packets go after they are dequeued from the per-VC queue, and the physical interface's transmit ring.

More advanced queueing strategies such as custom queueing, priority queuing, or CBWFQ, along with additional QoS features such as fragmentation and interleaving, and various forms of adaptive shaping, can be called from inside the frame-relay map-class. You cannot change the main interface's queuing strategy, but you can tune the queue size using **`hold-queue out`** command as well as adjust the transmit ring's size.

In this particular scenario R2 does not require any additional configuration, as its physical circuit speed already matches the provider's guarantee. On R3 the connection speed is higher than the contracted CIR, so shaping is needed. On R5 two classes are used to set individual CIRs to match the respective remote sites' connection speeds. Note that the values calculated for B_c and B_e use the same logic as GTS, where B_c = CIR * T_c/1000, and maxB_e = (AR – CIR) * T_c/1000. The Tc value is indirectly set to 10ms by configuring B_c to provide the minimum amount of latency possible for VoIP packets.

Copyright © 2010 Internetwork Expert

This configuration can be verified as follows.

```
Rack1R5#show traffic-shape

Interface   Se0/0/0
       Access Target     Byte   Sustain   Excess    Interval  Increment Adapt
VC     List   Rate       Limit  bits/int  bits/int  (ms)      (bytes)   Active
502           128000     160    1280      0         10        160       -
503           256000     320    2560      0         10        320       -
504           56000      875    7000      0         125       875       -
513           56000      875    7000      0         125       875       -
501           56000      875    7000      0         125       875       -

Rack1R3#show traffic-shape

Interface   Se1/0
       Access Target     Byte   Sustain   Excess    Interval  Increment Adapt
VC     List   Rate       Limit  bits/int  bits/int  (ms)      (bytes)   Active
301           56000      875    7000      0         125       875       -
302           56000      875    7000      0         125       875       -
304           56000      875    7000      0         125       875       -
305           256000     480    2560      1280      10        320       -
```

Note that all PVCs, even non-configured PVCs, have shaping applied with the default CIR/T_c values 56Kbps/125ms. It is possible to verify the shaping settings per VC, as well as see the default queuing strategy for the per-VC shapers.

```
Rack1R3#show frame-relay pvc 305

PVC Statistics for interface Serial1/0 (Frame Relay DTE)

DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial1/0

  input pkts 89754          output pkts 1033755       in bytes 10568524
  out bytes 120994552       dropped pkts 0            in pkts dropped 0
  out pkts dropped 0                  out bytes dropped 0
  in FECN pkts 0            in BECN pkts 0            out FECN pkts 0
  out BECN pkts 0           in DE pkts 0              out DE pkts 0
  out bcast pkts 2558       out bcast bytes 908868
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 06:34:36, last time pvc status changed 06:34:04
  cir 256000    bc 2560      be 1280      byte limit 480    interval 10
  mincir 128000    byte increment 320   Adaptive Shaping none
  pkts 496       bytes 176576    pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/40, 0 drop, 0 dequeued
```

If you want to change the per-VC FIFO queue depth configure it under the map-class settings using the **frame-relay holdq** command. In addition you can change the interface's FIFO queue depth using the **hold-queue** command. This can be verified as follows.

```
R3:
map-class frame-relay TO_R5
 frame-relay holdq 100
!
interface Serial 1/0
 hold-queue 50 out
```

```
Rack1R3#show interfaces serial 1/0
Serial1/0 is up, line protocol is up
<snip>
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops:
943611
  Queueing strategy: fifo
  Output queue: 0/50 (size/max)
  5 minute input rate 1000 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     92400 packets input, 10673708 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     92697 packets output, 10350543 bytes, 0 underruns
     0 output errors, 0 collisions, 2 interface resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

```
Rack1R3#show frame-relay pvc 305

PVC Statistics for interface Serial1/0 (Frame Relay DTE)

DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial1/0

<snip>
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 06:49:24, last time pvc status changed 06:48:52
  cir 256000    bc 2560      be 1280      byte limit 480    interval 10
  mincir 128000    byte increment 320   Adaptive Shaping none
  pkts 592       bytes 210752    pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/100, 0 drop, 0 dequeued
```

Copyright © 2010 Internetwork Expert

10.21 Legacy Adaptive FRTS

- The service provider used by R3 has agreed to allow link oversubscription up to 384Kbps, but only guarantees delivery up to 256Kbps.
- Configure R3 to send as fast as possible on this circuit, but to slow down its sending rate to the guaranteed rate when service provider network notifies it of congestion or the physical interface is congested.
- Ensure that R5 can inform R3 about congestion in the case that R5 is receiving a unidirectional traffic feed from R3.

Configuration

```
R3:
map-class frame-relay TO_R5
 frame-relay cir 384000
 frame-relay mincir 256000
 frame-relay bc 3840
 frame-relay be 0
 frame-relay adaptive-shaping becn
 frame-relay adaptive-shaping interface-congestion

R5:
map-class frame-relay TO_R3
 frame-relay fecn-adapt
```

Verification

Note

Legacy FRTS supports four types of adaptive shaping, BECN adapt, FECN reflection, queue depth monitoring, and a proprietary method called ForeSight. Like GTS on Frame Relay interfaces, BECN adapt allows the router to slow its sending rate down to minCIR when BECNs are received from the provider cloud. FECN reflection allows the receiving router to generate a test frame back to the sender with the BECN field set when a FECN is received from the provider network. Like previously discussed this feature is needed for unidirectional traffic flows, such as IPTV. Queue depth monitoring allows the sending router to slow its sending rate if the main interface's logical queue depth exceeds a certain threshold. Since the interface level queue with FRTS is FIFO, and is shared by all PVCs, once the interface queue reaches a configured threshold, all PVCs set to adapt to interface congestion slow their sending rate down to the configured minCIR. Queue depth monitoring is configured with the command **`frame-relay adaptive-shaping interface-congestion <depth>`** under the frame-relay map-class, as seen in the above example.

Copyright © 2010 Internetwork Expert

In this particular scenario, the service provider allows R3 to send traffic above the guaranteed CIR. Note that in this case *CIR* refers to the guaranteed rate of the service provider contract, and not the average rate the router is shaping to with the **`frame-relay cir`** command. The traffic rate above the provider's CIR is called the Excess Information Rate (EIR), and the sum CIR+EIR is called the Peak Information Rate (PIR). In oversubscription scenarios it is common to call the provider's guaranteed rate "minCIR" and configure the **`frame-relay cir`** parameter to the value of PIR. This is where much confusion arises in Frame Relay Traffic Shaping, as the **`frame-relay cir`** command would be more appropriately named the *frame-relay average-rate*, as the router's CIR does not relate in any way to the provider's guaranteed CIR.

The router's parameters for this task are therefore as follows:

CIR = 384Kbps
minCIR = 256Kbps
T_c=10ms (for future voice traffic deployments per the previous task)
B_c = CIR * T_c/1000 = 384000 * 10/1000 = 3840 bits
B_e=0 (no Excess Bursting since we are already sending at the maximum allowed interface speed).

This configuration can be verified as follows.

```
Rack1R3#show frame-relay pvc 305

PVC Statistics for interface Serial1/0 (Frame Relay DTE)

DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial1/0

<snip>
  Shaping adapts to BECN
  pvc create time 08:23:18, last time pvc status changed 08:22:46
  cir 384000    bc 3840      be 0         byte limit 480    interval 10
  mincir 256000    byte increment 480   Adaptive Shaping BECN and IF_CONG
  pkts 1202      bytes 427912    pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/100, 0 drop, 0 dequeued
```

Copyright © 2010 Internetwork Expert

Send a barrage of ICMP packets from SW1 towards R5 and see if R3's output queue fills up and adaptive shaping kicks in.

```
Rack1SW1#ping 155.1.0.5 repeat 1000000 timeout 0

Type escape sequence to abort.
Sending 1000000, 100-byte ICMP Echos to 155.1.0.5, timeout is 0
seconds:
 ............................................................................
 ............................................................................

Rack1R3#show interfaces serial 1/0
Serial1/0 is up, line protocol is up
<snip>
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops:
1055360
  Queueing strategy: fifo
  Output queue: 50/50 (size/max)
  5 minute input rate 30000 bits/sec, 35 packets/sec
  5 minute output rate 29000 bits/sec, 35 packets/sec
<snip>

Rack1R3#show traffic-shape queue
Traffic queued in shaping queue on Serial1/0 dlci 305
  Queueing strategy: fcfs
  Queueing Stats: 96/100/71477 (size/max total/drops)
Packet 1, linktype: ip, length: 104, flags: 0x10000088
  source: 155.1.37.7, destination: 155.1.0.5, id: 0xA44E, ttl: 254, prot: 1
    data: 0x0800 0xDE91 0x001D 0x6E55 0x0000 0x0000 0x0223
          0x2F23 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD

Packet 2, linktype: ip, length: 104, flags: 0x10000088
  source: 155.1.37.7, destination: 155.1.0.5, id: 0xA44F, ttl: 254, prot: 1
    data: 0x0800 0xDE90 0x001D 0x6E56 0x0000 0x0000 0x0223
          0x2F23 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD

Packet 3, linktype: ip, length: 104, flags: 0x10000088
  source: 155.1.37.7, destination: 155.1.0.5, id: 0xA450, ttl: 254, prot: 1
    data: 0x0800 0xDE8F 0x001D 0x6E57 0x0000 0x0000 0x0223
          0x2F23 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD

Packet 4, linktype: ip, length: 104, flags: 0x10000088
  source: 155.1.37.7, destination: 155.1.0.5, id: 0xA451, ttl: 254, prot: 1
    data: 0x0800 0xDE8E 0x001D 0x6E58 0x0000 0x0000 0x0223
          0x2F23 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD 0xABCD
<snip>
```

The above output *Queueing stragety: fcfs* means First Come First Served, or in other words, FIFO.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.22 Legacy FRTS with Per-VC WFQ

- Configure WFQ in R3's map-class for the PVC towards R5.
- Set the number of flow queues to 32, the queue size to 512, and start discarding packets when any flow reaches a length of 16 packets.

Configuration

```
R3:
map-class frame-relay TO_R5
 frame-relay fair-queue 16 32 0 512
```

Verification

> **Note**
>
> Legacy FRTS supports two levels of queuing, Per-VC, and per-interface. While the interface queue is fixed to a simple FIFO queue, it is possible to tune the Per-VC queues. Per-VC queuing allows moving congestion management to the VC level, thus preventing congestion at one PVC from affecting other VCs. Note that the VC queue is implemented using the shaper, thus the minimum time quantum for the queue scheduler is based on the minimum T_c for the platform.
>
> The settings for Per-VC WFQ are the same as for legacy interface level WFQ, but the size of the queue is set with the **`frame-relay fair-queue`** command, not the **`frame-relay holdq`**. With Per-VC WFQ it's also possible to allocate RSVP flows, and activate other WFQ features such as IP RTP Priority. The syntax is:
>
> **`frame-relay fair-queue <CDT> <NumFlows> <RSVP Flows> <Max Buffers>`**

Copyright © 2010 Internetwork Expert

This configuration can be verified as follows.

```
Rack1R3#show traffic-shape queue
Traffic queued in shaping queue on Serial1/0 dlci 305
  Queueing strategy: weighted fair
  Queueing Stats: 0/512/16/0 (size/max total/threshold/drops)
     Conversations  0/0/32 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 256 kilobits/sec
```

WFQ computes the available bandwidth based on the minCIR setting in the map-class. Note that if this value is not defined it defaults to half of the **`frame-relay cir`**. In any case the queue only activates when the PVC exceeds it's configured traffic rate, e.g. when the shaper starts delaying traffic.

```
Rack1R3#show frame-relay pvc 305

PVC Statistics for interface Serial1/0 (Frame Relay DTE)

DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial1/0

<snip>
  cir 384000    bc 3840      be 0         byte limit 480    interval 10
  mincir 256000    byte increment 480   Adaptive Shaping BECN and IF_CONG
  pkts 148       bytes 52688     pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: weighted fair
  Current fair queue configuration:
   Discard     Dynamic      Reserved
   threshold   queue count  queue count
    16          32           0
  Output queue size 0/max total 512/drops 0
```

10.23 Legacy FRTS with Per-VC PQ

- Configure R5 to prioritize HTTP traffic sent to R3 over the Frame Relay link, but only if there are no RIP routing updates to send.
- ICMP traffic should only be sent if there is no other traffic queued on the link.

Configuration

```
R5:
access-list 100 permit icmp any any
!
priority-list 1 protocol ip high tcp www
priority-list 1 protocol ip low list 100
!
map-class frame-relay TO_R3
 frame-relay priority-group 1
```

Verification

Note

Legacy FTRS allows the attaching of different priority-lists on a per-VC basis, thereby allowing different per-VC priority queues. Create the priority-list as normal, then attach it to the PVC by issuing the **`frame-relay priority-group <ListNumber>`** command under the frame-relay map-class.

Note that even though this task states that RIP packets should be dequeued before HTTP, no additional configuration is needed to do this since RIP updates will *not* be queued at the per-VC queue level. The reason is that by default Frame-Relay *broadcasts* (packets that require replication over multiple VC) are queued to special interface-level priority queue, called the frame-relay broadcast queue. This queue is separate from the interface-level FIFO queue shared by all VCs, and is discussed in more detail later.

The operation of the priority queue can be verified as follows.

```
Rack1R5#show traffic-shape queue
Traffic queued in shaping queue on Serial0/0/0 dlci 501
  Queueing strategy: fcfs
Traffic queued in shaping queue on Serial0/0/0 dlci 513
  Queueing strategy: fcfs
Traffic queued in shaping queue on Serial0/0/0 dlci 504
  Queueing strategy: fcfs
Traffic queued in shaping queue on Serial0/0/0 dlci 503
  Queueing strategy: priority-group 1
Traffic queued in shaping queue on Serial0/0/0 dlci 502
  Queueing strategy: fcfs
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show frame-relay pvc 503

PVC Statistics for interface Serial0/0/0 (Frame Relay DTE)

DLCI = 503, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0

  input pkts 122699        output pkts 130586       in bytes 18493120
  out bytes 24033430       dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 36099     out bcast bytes 13978192
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 6000 bits/sec, 2 packets/sec
  pvc create time 3d03h, last time pvc status changed 1d10h
  cir 256000    bc 2560      be 0         byte limit 320    interval 10
  mincir 128000    byte increment 320   Adaptive Shaping none
  pkts 9035      bytes 1399132   pkts delayed 5275      bytes delayed 626532
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: priority-list 1

List   Queue  Args
1      low    protocol ip          list 100

List   Queue  Args
1      high   protocol ip          tcp port www
  Output queue: high 0/20/0, medium 0/40/0, normal 0/60/0, low 0/80/0
```

To verify the priority queue temporarily set the shaping rate to 64Kbps for the map-class for the DLCI switched to R3. Next, enable the HTTP server service on R5 and start transferring IOS images from R5 to R3, and ping R3 from SW2.

```
R5:
ip http server
ip http path flash:
!
map-class frame-relay TO_R3
 frame-relay cir 64000
```

```
Rack1R3#copy http://admin:cisco@155.1.0.5/c2600- c1841-adventerprisek9-
mz.124-24.T.bin null:
Loading http://***********@155.1.0.5/ c1841-adventerprisek9-mz.124-24.T.bin
!!!!

Rack1SW2#ping 155.1.0.3 repeat 1000 size 100

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 155.1.0.3, timeout is 2 seconds:
!!..!
<snip>
```

Copyright © 2010 Internetwork Expert

Verify the queuing statistics on the PVC connecting R5 to R3. Note the queue usage counters for each queue.

```
Rack1R5#show frame-relay pvc 503

PVC Statistics for interface Serial0/0 (Frame Relay DTE)

DLCI = 503, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0

  input pkts 1991          output pkts 1952         in bytes 93680
  out bytes 167452         dropped pkts 23          in pkts dropped 0
  out pkts dropped 23               out bytes dropped 1932
  late-dropped out pkts 23          late-dropped out bytes 1932
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 85        out bcast bytes 11060
  5 minute input rate 3000 bits/sec, 5 packets/sec
  5 minute output rate 3000 bits/sec, 6 packets/sec
  pvc create time 00:02:35, last time pvc status changed 00:02:22
  cir 64000     bc 2560      be 0         byte limit 320    interval 40
  mincir 32000     byte increment 320   Adaptive Shaping none
  pkts 1931      bytes 163288    pkts delayed 1669      bytes delayed 139588
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: priority-list 1

List   Queue  Args

List   Queue  Args
1      high   protocol ip          tcp port www
1      low    protocol ip          list 100
  Output queue: high 14/20/23, medium 0/40/0, normal 0/60/0, low 2/80/0
```

Check the contents of the traffic-shaping queue.

```
Rack1R5#show traffic-shape queue serial 0/0/0 dlci 503
Traffic queued in shaping queue on Serial0/0 dlci 503
  Queueing strategy: priority-group 1
  Queueing Stats: high 9/20/116 (queue/size/max total/drops)
Packet 1, linktype: ip, length: 84, flags: 0x10000008
  source: 155.1.0.5, destination: 155.1.0.3, id: 0xCBC1, ttl: 255,
  TOS: 0 prot: 6, source port 80, destination port 24427
    data: 0x0050 0x5F6B 0x0CC9 0xF509 0x3D35 0x677E 0x5010
          0x1020 0x1C35 0x0000 0x6EFC 0x74A2 0xBA31 0x45A4

Packet 2, linktype: ip, length: 84, flags: 0x10000008
  source: 155.1.0.5, destination: 155.1.0.3, id: 0xCBC2, ttl: 255,
  TOS: 0 prot: 6, source port 80, destination port 24427
    data: 0x0050 0x5F6B 0x0CC9 0xF531 0x3D35 0x677E 0x5010
          0x1020 0x3B78 0x0000 0x409E 0xC9D5 0x9324 0x34CA
<snip>
```

Copyright © 2010 Internetwork Expert

10.24 Legacy FRTS with Per-VC CQ

- Configure R5 to share the bandwidth for the VC connecting R5 to R2 in the following proportions:
  - 60% of bandwidth for HTTP traffic
  - 20% of bandwidth for telnet traffic
  - 10% of bandwidth for ICMP traffic
  - RIP should use the system queue
- Limit each queue size to 15 packets.

Configuration

```
R5:
access-list 100 permit icmp any any
!
queue-list 1 queue 0 limit 15
queue-list 1 protocol ip 1 tcp www
queue-list 1 protocol ip 2 tcp telnet
queue-list 1 protocol ip 3 list 100
!
queue-list 1 queue 0 limit 15
queue-list 1 queue 1 byte-count 600 limit 15
queue-list 1 queue 2 byte-count 200 limit 15
queue-list 1 queue 3 byte-count 100 limit 15
!
map-class frame-relay TO_R2
 frame-relay custom-queue-list 1
!
interface Serial 0/0/0
 ip mtu 80
```

Copyright © 2010 Internetwork Expert

Verification

Note

Similar to legacy priority queueing, legacy custom queueing can be applied with FRTS on a per-VC basis. To apply this create the custom queue as normal, then attach it to the map-class with the **frame-relay queue-list <ListNumber>** command.

As discussed in the previous task, RIP packets will not use the per-VC queue if they are broadcasts or multicasts, and instead will go to the broadcast queue. To force RIP packets to go into the per-VC queue the multicast updates can be converted to unicast updates by using the **neighbor** command under the RIP process.

To verify this configuration, temporarily configure RIP unicasts across the Frame Relay interface, enable the HTTP server service on R5, and start a file transfer from R5 down to R2. Temporality shape R2's PVC on R5 down to 64Kbps to match the actual physical line rate.

```
R5:
router rip
 passive-interface Serial 0/0
 neighbor 155.1.0.2
 neighbor 155.1.0.3
 neighbor 155.1.0.4
 neighbor 155.1.0.1
!
map-class TO_R2
 frame-relay cir 64000
!
ip http server
ip http path flash:
```

```
Rack1R2#copy http://admin:cisco@155.1.0.5/c1841-adventerprisek9-mz.124-
24.T.bin null:
Loading http://***********@155.1.0.5/c2600-adventerprisek9-mz.124-
10.bin !!!!
```

```
Rack1SW2#ping 155.1.0.2 repeat 10000 size 80

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 155.1.0.2, timeout is 2 seconds:
!!
<snip>
```

Verify the custom queue statistics on VC 502 as follows.

```
Rack1R5#show frame-relay pvc 502

PVC Statistics for interface Serial0/0 (Frame Relay DTE)

DLCI = 502, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0

<snip>
  pvc create time 06:51:37, last time pvc status changed 06:51:25
  cir 128000     bc 1280      be 0          byte limit 160     interval 10
  mincir 64000      byte increment 160   Adaptive Shaping none
  pkts 197       bytes 15432     pkts delayed 145       bytes delayed 11132
  shaping active
  traffic shaping drops 0
  Queueing strategy: custom-list 1

List   Queue  Args

List   Queue  Args
1      0      protocol ip          udp port rip
1      1      protocol ip          tcp port www
1      2      protocol ip          tcp port telnet
1      3      protocol ip          list 100
1      0      limit 15
1      1      byte-count 600 limit 15
1      2      byte-count 200 limit 15
1      3      byte-count 100 limit 15
  Output queues: (queue #: size/max/drops/dequeued)
     0: 0/15/0/12 1: 2/15/47/148 2: 0/15/0/0 3: 0/15/0/6 4: 0/20/0/0
     5: 0/20/0/0 6: 0/20/0/0 7: 0/20/0/0 8: 0/20/0/0 9: 0/20/0/0
     10: 0/20/0/0 11: 0/20/0/0 12: 0/20/0/0 13: 0/20/0/0 14: 0/20/0/0
     15: 0/20/0/0 16: 0/20/0/0
```

Look at the queue usage counters and note the system queue counter used to queue unicast RIP updates is going up, since RIP no longer uses the interface's broadcast queue. Below are the actual contents of the VC's custom-queue. It shows the HTTP flow between R5 and R2.

```
Rack1R5#show traffic-shape queue serial 0/0/0 dlci 502
Traffic queued in shaping queue on Serial0/0 dlci 502
  Queueing strategy: custom-queue list 1
  Queueing Stats: 1/9/15/9 queue #/size/max total/drops)
Packet 1, linktype: ip, length: 84, flags: 0x10000008
  source: 155.1.0.5, destination: 155.1.0.2, id: 0x64CB, ttl: 255,
  TOS: 0 prot: 6, source port 80, destination port 23603
    data: 0x0050 0x5C33 0x1FEC 0xBCE0 0xA7F4 0x5B03 0x5010
          0x1020 0xA5B5 0x0000 0x417B 0x2E1D 0x5158 0xCFE2

Packet 2, linktype: ip, length: 84, flags: 0x10000008
  source: 155.1.0.5, destination: 155.1.0.2, id: 0x64CC, ttl: 255,
  TOS: 0 prot: 6, source port 80, destination port 23603
    data: 0x0050 0x5C33 0x1FEC 0xBD08 0xA7F4 0x5B03 0x5010
          0x1020 0x9CE6 0x0000 0xD8DD 0x5019 0x5B01 0xD548
<snip>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.25 Legacy FRTS with Per-VC Fragmentation

- Remove any artificial IP MTU settings from the Frame Relay interfaces of R3 and R5 along with any per-VC queueing strategies.
- The connection between R3 and R5 is used to send data and voice traffic.
- Configure Frame Relay fragmentation so that voice packets are never delayed for more than 10ms when serialized.
- Assume the speed of R3's connection is 384Kbps
- Ensure R2 is not affected by this configuration.

Configuration

```
R3:
map-class frame-relay TO_R5
 frame-relay fragment 480
!
interface Serial 1/0
 no ip mtu

R5:
map-class frame-relay TO_R3
 frame-relay fragment 480
!
interface Serial 0/0/0
 no ip mtu
```

Verification

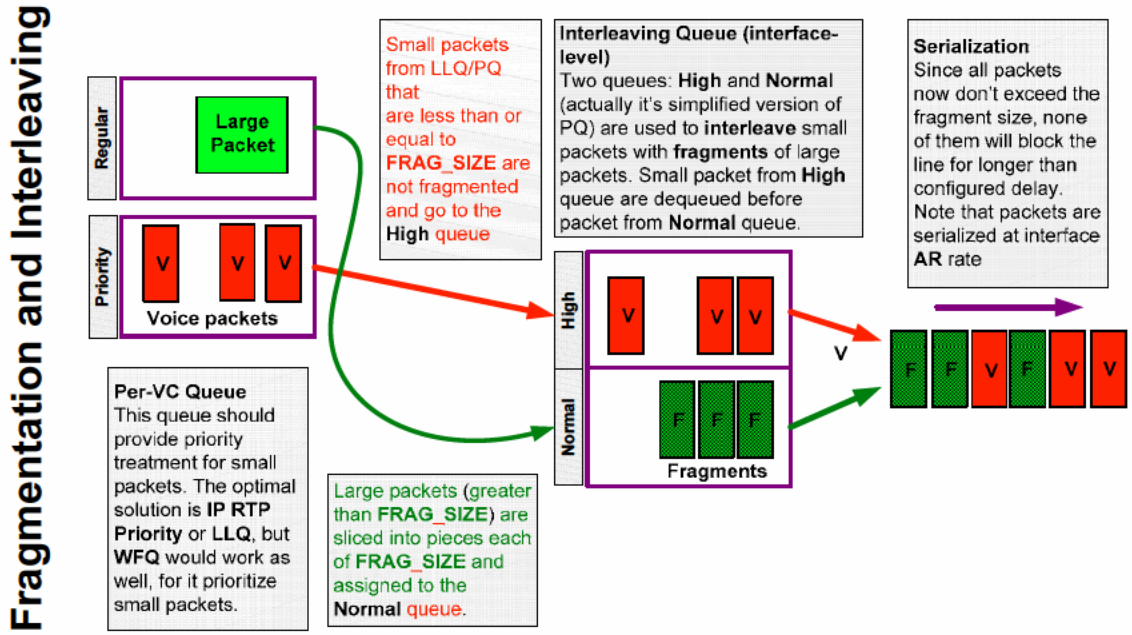
Note

Fragmentation is a technique used to avoid a "line blocking" effect due to the transmit ring of slow links. "Slow" links are defined as any media that takes more than 10ms to serialize an average data packet of 1500 bytes. On these links small voice packets, such as packets with a size around 60 bytes, may have to wait a considerable amount of time if the system serializes them after a large data packet. This "line blocking" effect can occur even if prioritization is configured in the software queue to send VoIP as fast as possible, and can effectively cancel out any QoS calculations.

Copyright © 2010 Internetwork Expert

For example assume that a priority VoIP packet of 60 bytes is trying to be sent out a 512Kbps Serial link. As the VoIP packet arrives in the output queue, it is moved from the tail of the queue to the head of the queue due to priority QoS configuration. At this exact instant, a 1500 byte data packet is admitted to the transmit ring, which has a queue depth of exactly 1 packet. Based on the data packet size and the 512Kbps access rate of the link, the VoIP packet now has to wait almost 24ms (1500 bytes * 8 bits/byte * 1000 ms/sec * 1 sec/512000 bits) to be admitted to the TxR, even though it is prioritized. If the TxR depth is more than one packet, the delay in the output queue can get even worse for the VoIP packet. Fragmentation techniques are designed to avoid this effect by breaking these larger packets into smaller pieces that require no more than 10ms to serialize.

Fragmentation on its own however, solves only part of the problem. We still need to ensure that fragmentation occurs for large packets, that VoIP is prioritized, and that the VoIP packets can be inserted (interleaved) between the now fragmented data packets. Frame Relay End-to-End Fragmentation (FRF.12) coupled with priority queueing can be used to solve this problem, as illustrated below.



FRF.12 utilizes two levels of queueing, the per-VC queue and the interface level queue. When you turn on fragmentation for a map-class, WFQ becomes the VC queuing strategy automatically. As we will see below, the WFQ strategy alone is *not* enough to enable interleaving. Next, the system converts the interface level FIFO queue into a special dual-FIFO queue. This dual-FIFO is actually the legacy PQ with two active queues – High and Normal. The system uses this special priority queue for interleaving as follows.

First, the shaper delays packets when congestion above CIR occurs. If there is a priority queue enabled on the VC (IP RTP Priority or LLQ), packets from this queue are serviced first. If the packet size is less than or equal to the configured fragment size, the system places the packet into the *high* portion of the dual-FIFO. In other words packets only get into the dual-FIFO interleaving priority queue if they were already prioritized and they don't need to be fragmented.

Next, large data packets with a size greater than the fragment size are sliced into smaller pieces, each of size *FRAG_SIZE* or smaller. These fragments are then queued to the *normal* interface queue of the dual-FIFO. Fragments are never queued to the *high* queue.

Once packets are at the interface level queue, the high queue is emptied completely before the normal queue is serviced. The result of this is that a large group of fragments can never block a small voice packet on the line. Note that the system always fragments large packets greater than *FRAG_SIZE*, even control plane packets such as routing updates, irrespective of whether there is congestion on the PVC or not. For this reason control plane problems can occur if the fragment size on both ends of the link is not the same. Also from a design prospective it does not make sense to enable fragmentation just on a single PVC, as other PVCs containing large packets may block the line. If you enable fragmentation, ensure it's enabled for all PVCs on the interface.

The computation of the fragment size must be based on the interface's physical access rate, not the PVC's CIR value, and should result in the fragment taking 10ms at most to be serialized. If the B_c value in your FRTS calculations already results in a T_c of 10ms (e.g. B_c = CIR * 10/1000), then the *FRAG_SIZE* will simply be the B_c value in bytes, as seen in the *Byte Increment* field in the **`show traffic-shape`** output. Fragmentation is usually unneeded on links with a speed of T1 and higher, as they already serialize an average 1500 byte packet in less than 10ms.

Copyright © 2010 Internetwork Expert

FRF.12 can be verified as follows.

```
Rack1R3#clear counters
Rack1R3#clear frame-relay counter

Rack1R3#show frame-relay pvc 305

PVC Statistics for interface Serial1/0 (Frame Relay DTE)

DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial1/0

<snip>
  fragment type end-to-end fragment size 480
<snip>
  frags 56        bytes 19028     frags delayed 56        bytes delayed 19028
  shaping active
  traffic shaping drops 0
  Queueing strategy: weighted fair
  Current fair queue configuration:
   Discard     Dynamic      Reserved
   threshold   queue count  queue count
    64          16           0
  Output queue size 3/max total 600/drops 0

Rack1R3#show frame-relay fragment
interface              dlci frag-type  size in-frag    out-frag   dropped-frag
Se1/0                  305  end-to-end 480  12         82         0
```

Check the interface queue of the Frame Relay interface of R3 and note that it is now dual-FIFO.

```
Rack1R3#show interfaces serial 1/0
Serial1/0 is up, line protocol is up
  Hardware is CD2430 in sync mode
  Internet address is 155.1.0.3/24
  MTU 1500 bytes, BW 128 Kbit, DLY 20000 usec,
     reliability 255/255, txload 43/255, rxload 19/255
  Encapsulation FRAME-RELAY, loopback not set
  Keepalive set (10 sec)
  LMI enq sent  185, LMI stat recvd 185, LMI upd recvd 0, DTE LMI up
  LMI enq recvd 0, LMI stat sent  0, LMI upd sent  0
  LMI DLCI 1023  LMI type is CISCO  frame relay DTE
  FR SVC disabled, LAPF state down
  Broadcast queue 0/64, broadcasts sent/dropped 176/0, interface
broadcasts 176
  Last input 00:00:00, output 00:00:00, output hang never
  Last clearing of "show interface" counters 00:30:42
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: dual fifo
  Output queue: high size/max/dropped 0/256/0
  Output queue: 0/128 (size/max)
<snip>
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#show queueing interface serial 1/0
Interface Serial1/0 queueing strategy: priority

Output queue utilization (queue/count)
       high/192 medium/0 normal/32238 low/0
```

Check the fragmentation statistics on R5 and confirm that packets between R2 and R5 are not fragmented.

```
Rack1R5#clear counters
Rack1R5#clear frame-relay counter

Rack1R5#show frame-relay fragment
interface              dlci frag-type  size in-frag    out-frag   dropped-frag
Se0/0/0                503  end-to-end 480  0          320        0

Rack1R5#show frame-relay pvc 503

PVC Statistics for interface Serial0/0/0 (Frame Relay DTE)

DLCI = 503, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0

<snip>
  frags 83        bytes 10612     frags delayed 24        bytes delayed 4476
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: weighted fair
  Current fair queue configuration:
   Discard     Dynamic      Reserved
   threshold   queue count  queue count
    64          16           0
  Output queue size 0/max total 600/drops 0

Rack1R5#show frame-relay pvc 502

PVC Statistics for interface Serial0/0/0 (Frame Relay DTE)

DLCI = 502, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0

<snip>
  pvc create time 09:33:23, last time pvc status changed 09:33:10
  cir 128000    bc 1280      be 0         byte limit 160    interval 10
  mincir 64000     byte increment 160   Adaptive Shaping none
  pkts 27        bytes 8752      pkts delayed 13        bytes delayed 1248
  shaping active
  traffic shaping drops 0
  Queueing strategy: custom-list 1

List   Queue  Args
1      0      protocol ip          udp port rip

List   Queue  Args
1      1      protocol ip          tcp port www
1      2      protocol ip          tcp port telnet
1      3      protocol ip          list 100
1      0      limit 15
1      1      byte-count 600 limit 15
1      2      byte-count 200 limit 15
1      3      byte-count 100 limit 15
```

Copyright © 2010 Internetwork Expert

```
  Output queues: (queue #: size/max/drops/dequeued)
     0: 0/15/0/22 1: 0/15/0/0 2: 0/15/0/0 3: 0/15/0/0 4: 0/20/0/0
     5: 0/20/0/0 6: 0/20/0/0 7: 0/20/0/0 8: 0/20/0/0 9: 0/20/0/0
     10: 0/20/0/0 11: 0/20/0/0 12: 0/20/0/0 13: 0/20/0/0 14: 0/20/0/0
     15: 0/20/0/0 16: 0/20/0/0
```

Now let's try to see if the interface-level dual-FIFO works. Configure R5 to download an IOS image from R3, and set R3's PVC speed to 64Kbps to match the interface rate. Additionally, source "small" ping packets from SW1 across R3 to R5.

```
R3:
map-class TO_R5
 frame-relay cir 64000
!
ip http server
ip http path flash:

Rack1R5#copy http://admin:cisco@155.1.0.3/c1841-adventerprisek9-mz.124-
24.T.bin null:
Loading http://************@155.1.0.3/c1841-adventerprisek9-mz.124-
24.T.bin !!!!

Rack1SW1#ping 155.1.0.5 repeat 10000 size 80

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 155.1.0.2, timeout is 2 seconds:
!!
```

Check the interleaving and shaping queue statistics. Note that two flows exist, corresponding to ping packets (80 bytes) and file transfers (large packets, sourced from port 80). Note the "High" queue is not used, and "Interleaves" counters are zero. That means **interleaving is not working**, since we didn't enable prioritization yet, and have just basic WFQ enabled per-queue.

```
Rack1R3#show queueing interface serial 1/0
Interface Serial1/0 queueing strategy: priority

Output queue utilization (queue/count)
      high/0 medium/0 normal/138 low/0
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#show traffic-shape queue serial 1/0 dlci 305
Traffic queued in shaping queue on Serial1/0 dlci 305
  Queueing strategy: weighted fair
  Queueing Stats: 3/600/64/0 (size/max total/threshold/drops)
     Conversations  2/5/16 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 32 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 1/32384/0/0/0
  Conversation 10, linktype: ip, length: 84
  source: 155.1.37.7, destination: 155.1.0.5, id: 0x5499, ttl: 254, prot: 1

  (depth/weight/total drops/no-buffer drops/interleaves) 2/32384/0/0/0
  Conversation 6, linktype: ip, length: 738
  source: 155.1.0.3, destination: 155.1.0.5, id: 0x264F, ttl: 255,
  TOS: 0 prot: 6, source port 80, destination port 25437
```

To confirm that interleaving is not working, enable debugging for the interface level priority queue (dual-FIFO). Note the packets of size 84 – those are the ping packets of size 80 plus the Frame-Relay 4 byte header. The other packets – 486 bytes and 86 bytes are fragments of the 1500 byte data packets. Each 1500-byte packet is split into 3x480 and 1x60 byte packets at L3, and then a 6 byte header is added. The additional 2 bytes are the FRF.12 header information for fragmented packets. Note that non-fragmented packets do not carry this additional 2 bytes of FRF.12 overhead.

```
Rack1R3#debug priority
Priority output queueing debugging is on
PQ: Serial1/0 output (Pk size/Q 84/2)
PQ: Serial1/0 output (Pk size/Q 486/2)
PQ: Serial1/0 output (Pk size/Q 486/2)
PQ: Serial1/0 output (Pk size/Q 486/2)
PQ: Serial1/0 output (Pk size/Q 486/2)
PQ: Serial1/0 output (Pk size/Q 486/2)
PQ: Serial1/0 output (Pk size/Q 486/2)
PQ: Serial1/0 output (Pk size/Q 486/2)
PQ: Serial1/0 output (Pk size/Q 486/2)
PQ: Serial1/0 output (Pk size/Q 68/2)
PQ: Serial1/0 output (Pk size/Q 84/2)
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.26 Legacy FRTS with Per-VC IP RTP Priority

- Modify R3 and R5's FRTS configuration to prioritize RTP VoIP traffic on the PVC between them.
- Allocate 128Kbps of bandwidth to voice traffic in the port range of 16384 – 32767.

Configuration

```
R3:
map-class frame-relay TO_R5
 frame-relay ip rtp priority 16384 16383 128

R5:
map-class frame-relay TO_R3
 frame-relay ip rtp priority 16384 16383 128
```

Verification

Note

Configuration of Frame Relay IP RTP Priority is the same as classic IP RTP Priority, with the exception that it applies per-VC, activating a priority flow inside the shaper's queue. The <*Bandwidth*> value in the syntax defines the policer rate applied to priority packets with the burst size of 1 second. This ensures that non-priority packets are not starved of bandwidth like they are with the legacy priority-list.

Once the per-VC Priority Queue is enabled, interleaving should now be active due to the previous FRF.12 fragmentation configuration. Note that Frame-Relay IP RTP priority only activates if FRF.12 fragmentation is active on the same PVC. This configuration can be verified as follows.

```
Rack1R3#show frame-relay pvc 305

PVC Statistics for interface Serial1/0 (Frame Relay DTE)

DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial1/0

<snip>
  fragment type end-to-end fragment size 480
  cir 384000    bc   3840      be 0         limit 480    interval 10
  mincir 256000    byte increment 480   BECN response yes IF_CONG yes
  frags 103       bytes 36668     frags delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  ip rtp priority parameters 16384 32767 128000
  Queueing strategy: weighted fair
  Current fair queue configuration:
<snip>
```

Copyright © 2010 Internetwork Expert

Now disable FRF.12 and see how IP RTP Priority goes away at the same time. This is due to the fact that the interface queue does not provide priority service without FRF.12 interleaving.

```
Rack1R3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#map-class frame-relay TO_R5
Rack1R3(config-map-class)#no frame-relay fragment 480
Rack1R3(config-map-class)#^Z

Rack1R3#show frame-relay pvc 305

PVC Statistics for interface Serial1/0 (Frame Relay DTE)

DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial1/0

  input pkts 243           output pkts 126          in bytes 75848
  out bytes 42656          dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 124       out bcast bytes 42264
  5 minute input rate 1000 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  Shaping adapts to BECN
  pvc create time 00:19:07, last time pvc status changed 00:18:35
  cir 384000    bc 3840      be 0         byte limit 480    interval 10
  mincir 256000    byte increment 480   Adaptive Shaping BECN and IF_CONG
  pkts 112       bytes 39872     pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: weighted fair
  Current fair queue configuration:
   Discard     Dynamic      Reserved
   threshold   queue count  queue count
    64          16           0
  Output queue size 0/max total 600/drops 0
```

Re-enable FRF.12, but now set the fragment size to 80 bytes for testing. At the same time set the CIR to 64Kbps, and the RTP Priority bandwidth to 32Kbps, and enable the HTTP server for verification purposes.

```
R3:
map-class frame-relay TO_R5
 frame-relay mincir 64000
 frame-relay cir 64000
 frame-relay ip rtp priority 16384 16383 32
 frame-relay fragment 80
!
ip http server
ip http path flash:
```

Copyright © 2010 Internetwork Expert

Configure R1 to start an SLA jitter probe towards R5 using G.729 sized packets and a destination port of 16384. Shutdown the VLAN 146 interface on R1 and the frame-relay interface to ensure that packets are taking the path to R5 via R3.

```
R5:
ip sla responder

R1:
no ip domain lookup
ip sla monitor 1
 type jitter dest-ipaddr 155.1.0.5 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
ip sla monitor schedule 1 life forever start-time now
!
interface FastEthernet 0/0
 shutdown
!
Interface Serial 0/0
 shutdown
```

Start transferring a large file (IOS image) from R3 to R5, and ping R5 from SW1.

```
Rack1R5#copy http://admin:cisco@155.1.0.3/c2600-adventerprisek9-mz.124-
10.bin null:
Loading http://***********@155.1.0.3/c2600-adventerprisek9-mz.124-
10.bin !!!!

Rack1SW1#ping 155.1.0.5 repeat 10000 size 80

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 155.1.0.2, timeout is 2 seconds:
!!
```

Check the interleaving and shaping queue statistics. Note that the HTTP flow is now being interleaved, and the UDP flow has a WFQ weight of 0 for priority.

```
Rack1R3#show traffic-shape queue serial 1/0 dlci 305
Traffic queued in shaping queue on Serial1/0 dlci 305
  Queueing strategy: weighted fair
  Queueing Stats: 3/600/64/0 (size/max total/threshold/drops)
     Conversations  2/3/16 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 32 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 2/0/0/0/0
  Conversation 24, linktype: ip, length: 64
  source: 155.1.67.6, destination: 155.1.0.5, id: 0x0003, ttl: 253,
  TOS: 0 prot: 17, source port 56329, destination port 16384

  (depth/weight/total drops/no-buffer drops/interleaves) 1/32384/0/0/1197
  Conversation 5, linktype: ip, length: 1504
  source: 155.1.0.3, destination: 155.1.0.5, id: 0x3720, ttl: 255,
```

```
TOS: 0 prot: 6, source port 80, destination port 44243
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#show frame-relay pvc 305

PVC Statistics for interface Serial1/0 (Frame Relay DTE)

DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial1/0

<snip>
  5 minute input rate 8000 bits/sec, 18 packets/sec
  5 minute output rate 58000 bits/sec, 15 packets/sec
  Shaping adapts to BECN
  pvc create time 00:52:45, last time pvc status changed 00:52:12
  fragment type end-to-end fragment size 80
  cir 64000     bc    3840      be 0         limit 480    interval 60
  mincir 64000     byte increment 480   BECN response yes IF_CONG yes
  frags 61037     bytes 5362484   frags delayed 60736     bytes delayed 5264932
  shaping active
  traffic shaping drops 0
  ip rtp priority parameters 16384 32767 32000
  Queueing strategy: weighted fair
  Current fair queue configuration:
   Discard     Dynamic      Reserved
   threshold   queue count  queue count
    64          16           0
  Output queue size 0/max total 600/drops 0

Rack1R3#show queueing interface serial 1/0
Interface Serial1/0 queueing strategy: priority

Output queue utilization (queue/count)
       high/1981 medium/0 normal/61972 low/0
```

Now that priority queueing is enabled on the VC, the high queue of the dual-FIFO is being used.

Copyright © 2010 Internetwork Expert

Check the SLA statistics on R1. Note the high RTT, which is inevitable with a link speed of 64Kbps, but that it is lower than the ICMP flow.

```
Rack1R1#show ip sla monitor statistics 1
Round trip time (RTT)     Index 1
      Latest RTT: 435 ms
Latest operation start time: 22:46:39.181 UTC Fri Aug 8 2008
Latest operation return code: OK
RTT Values
      Number Of RTT: 1000
      RTT Min/Avg/Max: 284/435/738 ms
Latency one-way time milliseconds
      Number of one-way Samples: 23
      Source to Destination one way Min/Avg/Max: 602/655/711 ms
      Destination to Source one way Min/Avg/Max: 4/25/52 ms
Jitter time milliseconds
      Number of Jitter Samples: 999
      Source to Destination Jitter Min/Avg/Max: 1/14/89 ms
      Destination to Source Jitter Min/Avg/Max: 1/3/253 ms
Packet Loss Values
      Loss Source to Destination: 0         Loss Destination to Source:
0
      Out Of Sequence: 0 Tail Drop: 0 Packet Late Arrival: 0
Voice Score Values
      Calculated Planning Impairment Factor (ICPIF): 21
MOS score: 3.68
Number of successes: 25
Number of failures: 0
Operation time to live: Forever
```

Note that ICMP packets of the same size have a larger RTT, even though the system schedules packets by using per-VC WFQ.

```
Rack1R1#ping 155.1.0.5 repeat 50 size 60

Type escape sequence to abort.
Sending 50, 60-byte ICMP Echos to 155.1.0.5, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (50/50), round-trip min/avg/max = 377/711/1017 ms
```

Copyright © 2010 Internetwork Expert

10.27 Frame-Relay RTP/TCP Header Compression

- Configure R3 and R5 to compress RTP and TCP headers on the PVC between them.
- Compression should not be enabled on any other PVCs.

Configuration

```
R3:
interface Serial 1/0
 frame-relay map ip 155.1.0.5 305 broadcast compress

R5:
interface Serial 0/0/0
 frame-relay ip rtp header-compression
 frame-relay ip tcp header-compression
 frame-relay map ip 155.1.0.1 501 broadcast nocompress
 frame-relay map ip 155.1.0.2 502 broadcast nocompress
 frame-relay map ip 155.1.0.4 504 broadcast nocompress
 frame-relay map ip 155.1.0.3 503 broadcast
```

Verification

Note

Frame-Relay IP RTP/TCP header compression is the same feature found on regular serial interfaces, however it allows applying compression on a per-VC basis. In the above example the feature is enabled two ways, on a per-VC basis with the **`frame-relay map`** statement, and on a per interface basis with the **`frame-relay ip rtp header-compression`**. In the latter case the other PVC (501, 502, and 504) have compression *disabled*, by adding the **`nocompress`** keyword to the end of their **`frame-relay map`**.

Copyright © 2010 Internetwork Expert

This feature can be verified as follows.

```
Rack1R3#show frame-relay pvc 305
PVC Statistics for interface Serial1/0 (Frame Relay DTE)

DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial1/0

  input pkts 26140         output pkts 26104        in bytes 1820351
  out bytes 2710933        dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 166       out bcast bytes 50356
  5 minute input rate 29000 bits/sec, 53 packets/sec
  5 minute output rate 32000 bits/sec, 55 packets/sec
  Shaping adapts to BECN
  pvc create time 00:20:00, last time pvc status changed 00:00:30
  fragment type end-to-end fragment size 80
  cir 64000     bc   3840      be 0         limit 480    interval 60
  mincir 64000     byte increment 480   BECN response yes IF_CONG yes
  frags 40183     bytes 2713014   frags delayed 27390     bytes delayed 1888823
  shaping active
  traffic shaping drops 0
  ip rtp priority parameters 16384 32767 32000
  Queueing strategy: weighted fair
  Current fair queue configuration:
   Discard     Dynamic      Reserved
   threshold   queue count  queue count
    16          32           0
  Output queue size 0/max total 512/drops 0

Rack1R5#show  ip tcp header-compression
TCP/IP header compression statistics:
 DLCI 503      Link/Destination info: ip 155.1.0.3
  Interface Serial0/0/0 DLCI 503 (compression on, VJ)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 256 rx slots, 256 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 256 free contexts

Rack1R5#show ip rtp header-compression
RTP/UDP/IP header compression statistics:
 DLCI 503      Link/Destination info: ip 155.1.0.3
  Interface Serial0/0/0 DLCI 503 (compression on, Cisco, RTP)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 256 rx slots, 256 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 256 free contexts

Rack1R5#show frame-relay ip tcp header-compression
TCP/IP header compression statistics:
 DLCI 305      Link/Destination info: ip 155.1.0.5
  Interface Serial0/0/0 DLCI 305 (compression on, VJ)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
```

Copyright © 2010 Internetwork Expert

```
    Connect: 256 rx slots, 256 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 256 free contexts

Rack1R3#show frame-relay ip rtp header-compression
 DLCI 305      Link/Destination info: ip 155.1.0.5
  Interface Serial1/0 DLCI 305 (compression on, Cisco, RTP)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 256 rx slots, 256 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 256 free context
```

Copyright © 2010 Internetwork Expert

10.28 Frame-Relay Broadcast Queue

- Modify R5's Frame Relay broadcast queue per the following requirements:
  - Limit the broadcast queue rate a maximum of 128Kbps
  - Limit the queue depth to 4 packets per spoke
  - Limit the total number of broadcast packets to 10 per second

Configuration

```
R5:
interface Serial 0/0/0
 frame-relay broadcast-queue 16 16000 10
```

Verification

Note

The Frame Relay broadcast queue is a special interface-level priority queue used to store and send pseudo-broadcast packets. Within the context of Frame Relay interfaces, a pseudo-broadcast is a layer 3 broadcast or multicast packet that must be replicated to many PVCs at the same time at layer 2. Since the broadcast queue include multicast packets, routing updates are queued here by default, which is why it is given preferential treatment automatically.

The **`frame-relay broadcast-queue`** command can be used to rate limit the depth, bit per second, or packet per second rate of broadcasts sent out the link, and can be verified as follows.

```
Rack1R5#show interfaces serial 0/0/0
Serial0/0/0 is up, line protocol is up
  Hardware is PowerQUICC Serial
  Internet address is 155.1.0.5/24
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation FRAME-RELAY, loopback not set
  Keepalive set (10 sec)
  LMI enq sent  2848, LMI stat recvd 2849, LMI upd recvd 0, DTE LMI up
  LMI enq recvd 0, LMI stat sent  0, LMI upd sent  0
  LMI DLCI 1023  LMI type is CISCO  frame relay DTE
  FR SVC disabled, LAPF state down
  Broadcast queue 2/16, broadcasts sent/dropped 20438/0, interface broadcasts
6102
<snip>
```

Copyright © 2010 Internetwork Expert

10.29 Frame-Relay DE Marking

- Configure R3 to mark all IP packets larger than 60 bytes as Discard Eligible as they are sent out to the Frame Relay cloud.

Configuration

```
R3:
frame-relay de-list 1 protocol ip gt 64
!
interface Serial 1/0
 frame-relay de-group 1 305
```

Verification

Note

The Discard Eligible (DE) bit is used in Frame Relay networks to determine which packets should be dropped first in the case of congestion. Like any marking, the DE bit has no meaning on its own without a policy configured in the service provider's network to actually implement the dropping. Usually the policy involves configuring a queue depth threshold where DE-marked packets are discarded, while regular packets are preserved.

Normally the DE bit is set on the PE device during input metering, but as seen in the above example it is possible to set the bit on the CE device outbound. This legacy configuration is rarely used however, as it only supports process switched packets, and does not support other switching paths such as fast switching or CEF switching, like the MQC-based configuration does. Remember that with legacy QoS configuration packet sizes include the full layer 2 packet length, so 64 bytes in the above configuration means any IP packet with a size of 60 bytes, plus 4 bytes of "Cisco" Frame Relay encapsulation, or greater.

Copyright © 2010 Internetwork Expert

To verify this configuration check the DE counter as follows. In this particular case the counter is increasing as local routing packets are larger than 60 bytes and are process switched.

```
Rack1R3#show frame-relay pvc 305 | inc DLCI|PVC|DE
PVC Statistics for interface Serial1/0 (Frame Relay DTE)
DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial1/0
  out BECN pkts 0          in DE pkts 0             out DE pkts 90

Rack1R5#show frame-relay pvc 503 | inc DLCI|PVC|DE
PVC Statistics for interface Serial0/0/0 (Frame Relay DTE)
DLCI = 503, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0
  out BECN pkts 0          in DE pkts 95            out DE pkts 0
```

CEF switched transit packets are not be marked with a DE bit.

```
Rack1SW1#ping 155.1.0.5 size 1000 repeat 10000 timeout 0

Type escape sequence to abort.
Sending 10000, 1000-byte ICMP Echos to 155.1.0.5, timeout is 0 seconds:
 ..............................................................
<snip>

Rack1R3#show frame-relay pvc 305 | inc DLCI|PVC|DE
PVC Statistics for interface Serial1/0 (Frame Relay DTE)
DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial1/0
  out BECN pkts 0          in DE pkts 0             out DE pkts 93
```

With CEF disabled the ICMP flow is marked as DE.

```
Rack1R3(config)#interface Serial1/0
Rack1R3(config-if)#no ip route-cache
Rack1R3(config-if)#end

Rack1R3#show frame-relay pvc 305 | inc DLCI|PVC|DE
PVC Statistics for interface Serial1/0 (Frame Relay DTE)
DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial1/0
  out BECN pkts 0          in DE pkts 0             out DE pkts 861

Rack1R3#show frame-relay pvc 305 | inc DLCI|PVC|DE
PVC Statistics for interface Serial1/0 (Frame Relay DTE)
DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial1/0
  out BECN pkts 0          in DE pkts 0             out DE pkts 1411
```

Copyright © 2010 Internetwork Expert

10.30 Legacy FRTS PVC Interface Priority Queue

- Configure PIPQ on R5 so that frames going out to R3 are always sent first, followed by frames going to R2.
- All other PVCs assigned to R5's interface should be serviced after R3 and R2.

Configuration

```
R5:
map-class frame-relay TO_R2
 frame-relay interface-queue priority normal
!
map-class frame-relay TO_R3
 frame-relay interface-queue priority medium
!
map-class frame-relay DEFAULT
 frame-relay interface-queue priority low
!
interface Serial 0/0/0
 frame-relay interface-queue priority
 frame-relay class DEFAULT
```

Verification

Note

PVC Interface Priority Queueing (PIPQ) enables the legacy Priority Queue at the VC level itself. The interface queue defined in the map-class maps the PVC to the respective high, medium, normal, or low queue. Note that broadcast packets (e.g. routing updates) still go to the interface-level broadcast queue, while Layer 2 keepalives (i.e. LMI keepalives) use the high priority queue.

Enabling PIPQ with per-VC fragmentation and IP RTP Priority breaks interleaving, as the system replaces the dual-FIFO interface queue with a fully functional Priority Queue. The system no longer splits the PVC traffic between interleaving queues, but rather all packets (small voice and large packet fragments) use the same interface-level queue. Therefore if you plan to use PIPQ for VoIP deployments ensure that voice traffic is the only traffic using the priority PVC, or that the connection speed is high enough to eliminate the problem of serialization delay.

Copyright © 2010 Internetwork Expert

PIPQ can be verified as follows.

```
Rack1R5#show interfaces serial 0/0/0
Serial0/0/0 is up, line protocol is up
  Hardware is PowerQUICC Serial
  Internet address is 155.1.0.5/24
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
     reliability 255/255, txload 4/255, rxload 4/255
  Encapsulation FRAME-RELAY, loopback not set
  Keepalive set (10 sec)
  LMI enq sent  211, LMI stat recvd 206, LMI upd recvd 0, DTE LMI up
  LMI enq recvd 0, LMI stat sent  0, LMI upd sent  0
  LMI DLCI 1023  LMI type is CISCO  frame relay DTE
  FR SVC disabled, LAPF state down
  Broadcast queue 0/16, broadcasts sent/dropped 1776/0, interface broadcasts
456
  Last input 00:00:00, output 00:00:00, output hang never
  Last clearing of "show interface" counters 00:35:04
  Input queue: 4/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: DLCI priority
  Output queue (queue priority: size/max/drops):
     high: 0/20/0, medium: 0/40/0, normal: 0/60/0, low: 0/80/0
  5 minute input rate 26000 bits/sec, 50 packets/sec
  5 minute output rate 27000 bits/sec, 52 packets/sec
     93649 packets input, 6201446 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     1 input errors, 0 CRC, 1 frame, 0 overrun, 0 ignored, 0 abort
     94712 packets output, 6481205 bytes, 0 underruns
     0 output errors, 0 collisions, 1 interface resets
     0 output buffer failures, 0 output buffers swapped out
     2 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up

Rack1R5#show frame-relay pvc 503 | include priority
  priority medium

Rack1R5#show frame-relay pvc 501 | include priority
  priority low

Rack1R5#show queueing interface serial 0/0/0
Interface Serial0/0 queueing strategy: priority

Output queue utilization (queue/count)
      high/228 medium/102656 normal/486 low/974
```

Note the PVC priority levels. By default the priority level is normal. Also note the high queue counter getting incremented, as this queue is used for LMI keepalive messages.

Copyright © 2010 Internetwork Expert

10.31 Frame-Relay Priority to DLCI Mapping

- Configure back-to-back Frame Relay on the Serial link between R4 and R5 using DLCIs 100 and 200.
- Configure a static mapping to R4 and R5's IP addresses using DLCI 100 on both sides.
- Create a legacy priority-list on both R4 and R5 that maps RIP updates and packets less than 65 bytes to the high queue.
- Map packets from the high queue to the DLCI 100 on both sides.
- Map all other queues to DLCI 200 on both sides.

Configuration

```
R4:
priority-list 1 protocol ip high lt 64
priority-list 1 protocol ip high udp 520
!
interface Serial0/1/0
 ip address 155.1.45.4 255.255.255.0
 encapsulation frame-relay
 no keepalive
 priority-group 1
 frame-relay priority-dlci-group 1 100 200 200 200
 frame-relay map ip 155.1.45.5 100 broadcast
 frame-relay interface-dlci 100
 frame-relay interface-dlci 200

R5:
priority-list 1 protocol ip high lt 64
priority-list 1 protocol ip high udp 520
!
interface Serial0/1/0
 ip address 155.1.45.5 255.255.255.0
 encapsulation frame-relay
 no keepalive
 priority-group 1
 frame-relay priority-dlci-group 1 100 200 200 200
 frame-relay map ip 155.1.45.4 100 broadcast
 frame-relay interface-dlci 100
 frame-relay interface-dlci 200
```

Copyright © 2010 Internetwork Expert

Verification

Note

Although not widely used, the DLCI-based Frame Relay priority feature is used to forward traffic out different VCs based on the traffic's priority level. The idea of priority to DLCI mapping is to use different "parallel" DLCIs to carry different types of traffic. This strategy however, requires that the service provider cloud assigns different SLAs to the different PVCs.

With this configuration only one **`frame-relay map`** statement is needed to a PVC in the priority group. The priority-list entries and queue to DLCI mappings actually determine which PVC is used when encapsulation occurs.

The format of the command syntax is **`frame-relay priority-dlci-group <N> <High> <Med> <Norm> <Low>`**, where *N* is the list number, and *High, Med, Norm,* and *Low* are the queue levels that the circuits are mapped to. In this particular case the result is that traffic matched by list number 1 in the high queue goes to VC 100, while all other traffic goes to VC 200.

To verify this clear the counters and generate a traffic flow.

```
Rack1R5#clear counters serial 0/1/0
Rack1R5#clear frame-relay counters

Rack1R5#show queueing interface serial 0/1/0
Interface Serial0/1/0 queueing strategy: priority

Output queue utilization (queue/count)
       high/94 medium/0 normal/12 low/0

Rack1R5#show frame-relay pvc 200

PVC Statistics for interface Serial0/1/0 (Frame Relay DTE)

DLCI = 200, DLCI USAGE = LOCAL, PVC STATUS = STATIC, INTERFACE = Serial0/1

  input pkts 10             output pkts 12           in bytes 2040
  out bytes 2692            dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0            in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0           in DE pkts 0             out DE pkts 0
  out bcast pkts 2          out bcast bytes 652
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 01:06:26, last time pvc status changed 01:06:26
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#ping 155.1.45.4 size 200 repeat 10

Type escape sequence to abort.
Sending 10, 200-byte ICMP Echos to 155.1.45.4, timeout is 2 seconds:
!!!!!!!!!!
Success rate is 100 percent (10/10), round-trip min/avg/max = 52/55/57 ms

Rack1R5#show queueing interface serial 0/1
Interface Serial0/1/0 queueing strategy: priority

Output queue utilization (queue/count)
      high/98 medium/0 normal/22 low/0

Rack1R5#show frame-relay pvc 200

PVC Statistics for interface Serial0/1/0 (Frame Relay DTE)

DLCI = 200, DLCI USAGE = LOCAL, PVC STATUS = STATIC, INTERFACE =
Serial0/1

  input pkts 20            output pkts 22           in bytes 4080
<snip>
```

Note that packets assigned to the normal queue use DLCI 200 for encapsulation, even though we map the IP address of the other router to DLCI 100. Now generate traffic for the high priority queue.

```
Rack1R5#clear counters serial 0/1/0
Clear "show interface" counters on this interface [confirm]

Rack1R5#show queueing interface serial 0/1/0
Interface Serial0/1 queueing strategy: priority

Output queue utilization (queue/count)
      high/0 medium/0 normal/0 low/0

Rack1R5#ping 155.1.45.4 repeat 100 size 40

Type escape sequence to abort.
Sending 100, 40-byte ICMP Echos to 155.1.45.4, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
<snip>

Rack1R5#show queueing interface serial 0/1/0
Interface Serial0/1/0 queueing strategy: priority

Output queue utilization (queue/count)
      high/102 medium/0 normal/0 low/0
```

Note the number of packets classified to the high queue. The additional matches are due to the RIP routing updates using the same queue.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.32 Frame-Relay Traffic Policing & Congestion Mgmt

- Enable Frame Relay switching on R3.
- Configure R3's links to R1 and R2 as Frame Relay DCE interfaces, and clock the links at 128Kbps.
- Configure R3 to terminate DLCI 133 to R1 using the IP address 155.X.13.3/24.
- Configure R3 to terminate DLCI 233 to R2 using the IP address 155.X.23.3/24.
- Configure R3 to switch DLCI 132 from R1 to R2 as DLCI 231.
- Configure two point-to-point subinterfaces on R1's link to R3.
- Configure R1 to use the IP address 155.X.13.1/24 and DLCI 133 for the first subinterface, and IP address 155.X.12.1/24 and DLCI 132 for the second subinterface.
- Configure two point-to-point subinterfaces on R2's link to R3.
- Configure R2 to use the IP address 155.X.23.2/24 and DLCI 233 for the first subinterface, and IP address 155.X.12.2/24 and DLCI 231 for the second subinterface.
- Ensure that R3 has IP connectivity to R1 and R2, and R1 and R2 have IP connectivity to each other over these circuits.
- Create two map-classes on R3 for the switched DLCIs with the following policies:
    - The class for R1 should be configured for an input CIR of 64Kbps, an input B_c and B_e of 8000 bits, and an input T_c of 125ms
    - The class for R2 should be configured for an input CIR zero, an input B_c of zero, an input B_e of 16000 bits, and an input T_c of 125ms
- Enable Frame Relay traffic-policing on R3's links to R1 and R2 and apply the above classes to the respective DLCIs.
- Configure generic traffic-shaping on R1 and R2 to average 128Kbps with a B_c of 16000 bits.
- R1 should reduce its sending rate to the guaranteed rate in the case that BECNs are received.
- Enable congestion management on R3's switched PVCs and set the thresholds for DE dropping to 5% and FECN/BECN marking to 3%.

Copyright © 2010 Internetwork Expert

Configuration

```
R1:
interface Serial0/1
 encapsulation frame-relay
!
interface Serial0/1.1 point-to-point
 ip address 155.1.12.1 255.255.255.0
 frame-relay interface-dlci 132
 traffic-shape rate 128000 16000 0
 traffic-shape adaptive 64000
!
interface Serial0/1.2 point-to-point
 ip address 155.1.13.1 255.255.255.0
 frame-relay interface-dlci 133

R2:
interface Serial0/1
 encapsulation frame-relay
!
interface Serial0/1.1 point-to-point
 ip address 155.1.12.2 255.255.255.0
 frame-relay interface-dlci 231
 traffic-shape rate 128000 16000 0
!
interface Serial0/1.2 point-to-point
 ip address 155.1.23.2 255.255.255.0
 frame-relay interface-dlci 233

R3:
frame-relay switching
!
! Note the "in" keyword only works for policing
! It's possible to set "out" keyword to use different values
! for traffic shaping on the same VC
!
map-class frame-relay FROM_R1
 frame-relay cir in 64000
 frame-relay bc in 8000
 frame-relay be in 8000
!
! Zero CIR contract. Thus "cir in" is set to 0, as well
! as "bc in". Note that now the "Tc" variable comes into play,
! as it is used to define the measurement interval.
!
map-class frame-relay FROM_R2
 frame-relay cir in 0
 frame-relay bc in 0
 frame-relay be in 16000
 frame-relay tc 125
```

Copyright © 2010 Internetwork Expert

```
!
! Apply map-classes to switched VCs
! and enable traffic policing
!
interface Serial1/2
 ip address 155.1.13.3 255.255.255.0
 encapsulation frame-relay
 clock rate 128000
 frame-relay interface-dlci 132 switched
  class FROM_R1
 frame-relay interface-dlci 133
 frame-relay intf-type dce
 frame-relay policing
!
! Congestion management thresholds
!
 frame-relay congestion-management
  threshold ecn 3
  threshold de 5
!
! Enable policing on the interface for switched DLCIs
! Apply map-classes to apply contract values
!
interface Serial1/3
 ip address 155.1.23.3 255.255.255.0
 encapsulation frame-relay
 clock rate 128000
 frame-relay interface-dlci 231 switched
   class FROM_R2
 frame-relay interface-dlci 233
 frame-relay intf-type dce
 frame-relay policing
!
! Congestion management thresholds
!
 frame-relay congestion-management
  threshold ecn 3
  threshold de 5
!
! Configure PVC switching
!
connect R1_TO_R2 Serial1/2 132 Serial1/3 231
```

Copyright © 2010 Internetwork Expert

Verification

Note

This task summarizes many of the legacy Frame Relay QoS concepts covered so far, including output shaping, input metering, oversubscription, DE marking, and adaptive shaping with BECN/FECN. In this case R3 is the service provider, and sells traffic contracts in terms of the PIR (Peak Information Rate) and CIR (Committed Information Rate), and defines the metering interval T_c. R3 guarantees CIR to the customer at any time, but the customer may send up to the PIR without any guarantee. The measurement interval is an essential part of the traffic contract, as it defines the "averaging" window to meter the traffic rate at the service providers side. In order to conform to the traffic contract, the customers (R1 and R2) must perform output *shaping* of their packet stream in order to conform to the provider's input *policing* used to meter the input burst rates.

Frame Relay Traffic Policing (FRTP) uses a *two-rate* policing process based on the token bucket model. The following parameters are used by the policer:

Bc=CIR/Tc
Be=(PIR-CIR)/Tc=EIR/Tc

Note that the definition of the B_e in this context is different than with shaping or with CAR. In FRTP B_e defines the EIR (Excess Information Rate) rate – the amount of exceeding burst size that could be accepted from the customer equipment. The B_e value in traffic shaping, on the other hand, defines the amount of excess data that could be sent if the previous shaping interval was underutilized. CAR uses B_e to simulate a random marking/dropping behavior. FRTP uses the classic policing procedure adapted to utilize Frame-Relay DE marking as follows.

If the sum of the incoming packet size plus the size of packets received during the last T_c are less than or equal to B_c, the packet *conforms* to CIR. If the incoming packet size plus the size of packets received during the last T_c is greater than B_c but less than B_c+B_e, the packet *exceeds*, but still under PIR. Packets above CIR but under PIR are marked with the DE bit automatically. If the incoming packet is already marked as DE, it automatically exceeds. If the incoming packet size plus the size of packets received during the last T_c is above B_c+B_e, the packet *violates* the contract, and the policer drops it.

Copyright © 2010 Internetwork Expert

As discussed previously a correct policer behavior depends on the customer shaping traffic properly. The customer accomplishes this by using the CIR, PIR and T_c values matching the provider settings. Note that legacy FRTS as well as GTS have no notion of PIR value, and therefore the customer side is commonly configured with Customer CIR=Provider PIR, and Customer minCIR = Provider CIR. This ensures that in case of congestion signaled by the service provider cloud, the customer throttles down its Frame Relay sending rate to the guaranteed rate.

Note that FRTP is configured with the same syntax as FRTS, but the options “in” and “out” are used to define if policing or shaping is occurring (policing is in, shaping is out). This allows the traffic on the same switched PVC to be policed (admitted) ingress and shaped egress. This feature is useful for network-to-network (NNI) connections, where the PVC is switched between different provider clouds, and it requires ingress contract verification by using policing as well as egress contract enforcement by using shaping.

Cisco IOS implements congestion management on DCE interfaces for traffic crossing switched PVCs only. The system measures the interface queue depth, and if the queue depth crosses the DE threshold (expressed in percent), all DE packets leaving all switched PVCs on the interface are discarded, while conforming packets are kept until the queue is 100% full. If the queue depth crosses a configurable ECN threshold, all Frame Relay packets leaving the interface across switched PVCs are marked with FECNs, and all incoming packets are marked with BECNs.

Verification for FRTP is as follows.

Copyright © 2010 Internetwork Expert

```
Rack1R3#show frame-relay pvc 132

PVC Statistics for interface Serial1/2 (Frame Relay DCE)

DLCI = 132, DLCI USAGE = SWITCHED, PVC STATUS = ACTIVE, INTERFACE = Serial1/2

  input pkts 3030          output pkts 2938         in bytes 1836582
  out bytes 1688472        dropped pkts 9           in pkts dropped 9
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 170        in DE pkts 0             out DE pkts 2257
  out bcast pkts 0         out bcast bytes 0
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  switched pkts 3030
  Detailed packet drop counters:
  no out intf 0            out intf down 0          no out PVC 0
  in PVC down 0            out PVC down 0           pkt too big 0
  shaping Q full 0         pkt above DE 0           policing drop 9
  pvc create time 05:28:47, last time pvc status changed 05:25:23
  policing enabled, 538 pkts marked DE
  policing Bc  8000        policing Be  8000        policing Tc  125 (msec)
  in Bc pkts   2086        in Be pkts   538         in xs pkts   9
  in Bc bytes  1058508     in Be bytes  590152      in xs bytes  5086
```

Note that the policing rate is not specified explicitly, just B_c, B_e, and T_c values are provided. The packets counted as "xs" (excess) have violated the policy and the system dropped them.

```
Rack1R3#show frame-relay pvc 231

PVC Statistics for interface Serial1/3 (Frame Relay DCE)

DLCI = 231, DLCI USAGE = SWITCHED, PVC STATUS = ACTIVE, INTERFACE = Serial1/3

  input pkts 2939          output pkts 2979         in bytes 1688968
  out bytes 1784910        dropped pkts 59          in pkts dropped 7
  out pkts dropped 52                out bytes dropped 52208
  late-dropped out pkts 52           late-dropped out bytes 52208
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 172
  out BECN pkts 0          in DE pkts 0             out DE pkts 538
  out bcast pkts 0         out bcast bytes 0
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  switched pkts 2939
  Detailed packet drop counters:
  no out intf 0            out intf down 0          no out PVC 0
  in PVC down 0            out PVC down 0           pkt too big 0
  shaping Q full 0         pkt above DE 52          policing drop 7
  pvc create time 05:28:06, last time pvc status changed 05:25:47
  policing enabled, 2258 pkts marked DE
  policing Bc  0           policing Be  16000       policing Tc  125 (msec)
  in Bc pkts   0           in Be pkts   2258        in xs pkts   7
  in Bc bytes  0           in Be bytes  1377514     in xs bytes  3346
```

Copyright © 2010 Internetwork Expert

This PVC has zero CIR, so all input packets are marked as DE.

Now let's see how oversubscription works with Frame-Relay. Configure both R1 and R3 as HTTP servers, and start downloading IOS images from both routers to R2. Both flows will oversubscribe the connection to R2 on R3, and we should see backpressure of BECN packets on R1.

```
R1:
ip http server
ip http path flash:
!
! Adjust MTU to minimize serialization delay to 10ms on 128K links
! Note that IP MTU of 156 corresponds to 160 bytes of L2 frame
!
interface Serial 0/1.1
 ip mtu 156

R3:
ip http server
ip http path flash:
!
! Adjust MTU to minimize serialization delay to 10ms on 128K links
! Note that IP MTU of 156 corresponds to 160 bytes of L2 frame
!
interface Serial 1/3
 ip mtu 156
```

```
Rack1R2#copy http://admin:cisco@155.1.23.3/c2600-adventerprisek9-
mz.124-10.bin null:

Rack1R2#copy http://admin:cisco@155.1.12.2/c2600-adventerprisek9-
mz.124-10.bin null:
```

Copyright © 2010 Internetwork Expert

Clear the Frame Relay counters using the command **clear frame-relay counters** and check the PVC statistic counters for the DLCI between R1 and R3. Note the large amount of BECN packets sent back to R1 and that 3800 input packets have been marked as DE.

```
Rack1R3#show frame-relay pvc 132

PVC Statistics for interface Serial1/2 (Frame Relay DCE)

DLCI = 132, DLCI USAGE = SWITCHED, PVC STATUS = ACTIVE, INTERFACE = Serial1/2

  input pkts 45041         output pkts 69551        in bytes 7083975
  out bytes 4490548        dropped pkts 118         in pkts dropped 0
  out pkts dropped 118              out bytes dropped 6321
  late-dropped out pkts 118         late-dropped out bytes 6321
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 346
  out BECN pkts 69661      in DE pkts 0             out DE pkts 69669
  out bcast pkts 0         out bcast bytes 0
  5 minute input rate 50000 bits/sec, 40 packets/sec
  5 minute output rate 24000 bits/sec, 66 packets/sec
  switched pkts 45041
  Detailed packet drop counters:
  no out intf 0            out intf down 0          no out PVC 0
  in PVC down 0            out PVC down 0           pkt too big 0
  shaping Q full 0         pkt above DE 118         policing drop 0
  pvc create time 06:13:33, last time pvc status changed 06:10:07
  policing enabled, 3800 pkts marked DE
  policing Bc  8000        policing Be  8000        policing Tc  125 (msec)
  in Bc pkts   46733       in Be pkts   3800        in xs pkts   9
  in Bc bytes  8251771     in Be bytes  1086818     in xs bytes  5086
```

The DLCI between R2 and R3 demonstrates a large number of dropped packets, which is equal to the number of outgoing DE packets (Frame-Relay counters have been reset during the transfer). Note the large amount of FECN marked frames leaving R3 towards R2.

```
Rack1R3#show frame-relay pvc 231

PVC Statistics for interface Serial1/3 (Frame Relay DCE)

DLCI = 231, DLCI USAGE = SWITCHED, PVC STATUS = ACTIVE, INTERFACE = Serial1/3

  input pkts 70245         output pkts 42081        in bytes 4522679
  out bytes 6635281        dropped pkts 3275        in pkts dropped 0
  out pkts dropped 3275              out bytes dropped 498050
  late-dropped out pkts 3275       late-dropped out bytes 498050
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 41442
  out BECN pkts 108        in DE pkts 0             out DE pkts 3275
  out bcast pkts 0         out bcast bytes 0
  5 minute input rate 24000 bits/sec, 67 packets/sec
  5 minute output rate 45000 bits/sec, 35 packets/sec
  switched pkts 70245
  Detailed packet drop counters:
  no out intf 0            out intf down 0          no out PVC 0
  in PVC down 0            out PVC down 0           pkt too big 0
  shaping Q full 0         pkt above DE 3275        policing drop 0
  pvc create time 06:12:49, last time pvc status changed 06:10:30
  policing enabled, 75455 pkts marked DE
  policing Bc  0           policing Be  16000       policing Tc  125 (msec)
  in Bc pkts   308         in Be pkts   75455       in xs pkts   7
  in Bc bytes  144682      in Be bytes  6502239     in xs bytes  3346
```

At the same time the DLCI for the direct connection between R2 and R3 demonstrates a higher transmission rate and no FECN/DE marked frames. The latter is because the DLCI is not switched, and thus is not subject to congestion management. The higher sending rate is because of two facts, the lower RTT, and the shaper at R1.

```
Rack1R3#show frame-relay pvc 233

PVC Statistics for interface Serial1/3 (Frame Relay DCE)

DLCI = 233, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial1/3

  input pkts 229750        output pkts 121198       in bytes 10182318
  out bytes 19352278       dropped pkts 0           in pkts dropped 0
  out pkts dropped 0               out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 917       out bcast bytes 132560
  5 minute input rate 43000 bits/sec, 122 packets/sec
  5 minute output rate 78000 bits/sec, 61 packets/sec
  pvc create time 06:08:19, last time pvc status changed 06:08:19
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

> **Note**
>
> Load the *QoS* initial configurations prior to starting.

10.33 MQC Classification and Marking

- Configure an outbound MQC policy on R4's Serial link to R5 per the following requirements:
  - WWW traffic from servers on VLAN 146 should be marked with an IP Precedence of 2
  - VoIP packets with UDP ports in the destination range of 16384 - 32767 and a Layer 3 packet size of 60 bytes should be marked with DSCP EF
  - ICMP packets larger than 1000 bytes should be dropped
  - All other packets that came from R4's connection to VLAN 146 with an IP precedence of 0 should be remarked with an IP precedence of 1
- Do not use an access-list to classify ICMP packets.

Configuration

```
R4:
ip access-list extended HTTP
 permit tcp 155.1.146.0 0.0.0.255 eq www any
!
ip access-list extended VOICE
 permit udp any any range 16384 32767
!
class-map HTTP
 match access-group name HTTP
!
class-map match-all LARGE_ICMP
 match protocol icmp
 match packet length min 1001
!
class-map match-all VOICE
 match access-group name VOICE
 match packet length min 60 max 60
!
class-map match-all SCAVENGER
 match input-interface FastEthernet0/1
 match ip precedence 0
!
policy-map SERIAL_LINK
 class VOICE
  set ip dscp ef
 class HTTP
  set ip precedence 2
 class LARGE_ICMP
  drop
```

Copyright © 2010 Internetwork Expert

```
 class SCAVENGER
  set ip precedence 1
!
interface Serial 0/1/0
 service-policy output SERIAL_LINK
```

Copyright © 2010 Internetwork Expert

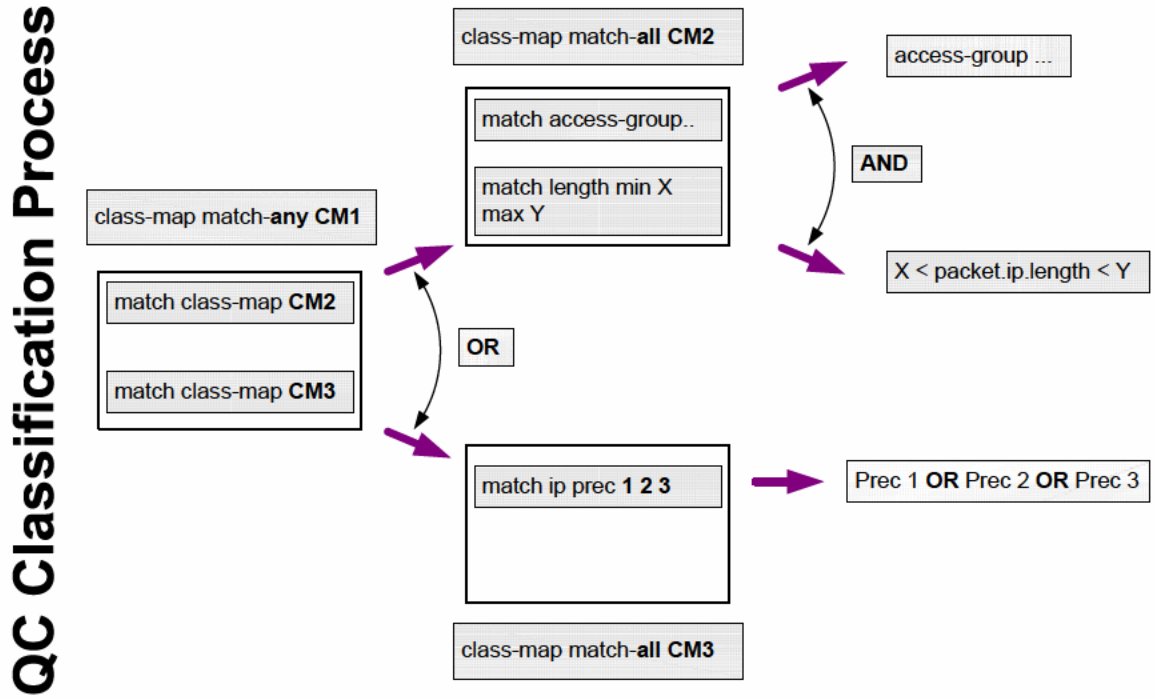
Verification

Note

The Modular Quality of Service Command Line Interface (MQC), also known as Class-Based Weighted Fair Queueing (CBWFQ), unifies all IOS QoS features under a single interface. MQC allows the implementation of a full suite of QoS tools, including classification, congestion management, traffic metering, marking, traffic shaping, and link efficiency. The main advantage of using the MQC over the legacy methods is that multiple QoS features can be applied to the same interface in the same direction. For example, with legacy QoS, you can't apply custom queueing and priority queueing at the same time, but with MQC you can.

Classification in MQC uses case sensitive *class-maps* (not to be confused with a frame-relay map-class) to group criteria. Each class-map performs a logical AND (match-all) or a logical OR (match-any) on its criteria. In other words in a match-all class-map, all matches must be TRUE in order for the class to be TRUE. Class-maps can be nested inside other maps to build complicated classification "AND-OR" logic gates. If multiple match criteria appear on the same line (e.g. match ip dscp, or match ip precedence), they are treated as a logical OR match.

Copyright © 2010 Internetwork Expert



Different IOS versions and platforms support different matches in the class-map, but as a general rule the following classification criteria are supported:

- Named and numbered access-lists - allows matching of IP addresses, TCP/UDP ports, IP protocol numbers, etc.
- Layer 3 packet length
- Layer 2 addresses - source/destination MAC address, Frame-Relay DLCI, etc.
- Packet marking - Layer 2 CoS, Layer 3 DSCP/IP precedence, Frame Relay DE, ATM CLP, MPLS EXP, etc.
- Network-Based Application Recognition (NBAR)
- Inverse logical matching (logical NOT)

When you apply a logical NOT to a nested class-map or multiple criteria in a single line, *De Morgan's law* applies, where NOT (X AND Y) = (NOT X) OR (NOT Y), while NOT (X OR Y) = (NOT X) AND (NOT Y).

Once classification is configured in a class-map, actions are defined for the different classes in a case sensitive *policy-map*. A policy map is an *ordered* list of class-maps with their corresponding actions, similar to how a route-map

Copyright © 2010 Internetwork Expert

works. The router matches packets entering/leaving the interface against all class-map entries in the respective input/output policy-map on the interface in a top-down fashion. This means that the first match in a class-map is used for classification, which implies that the order of the classes called in the policy-map is significant. The policy-map actions include marking, shaping, policing, assigning queue weight, compressing, etc. Any unclassified traffic in a policy-map falls into the *class-default* category, is covered in depth, along with the policy-map actions, in following sections.

Pitfall

Correct traffic flow classification within the class-map, and the correct order of operations in the policy-map, is key in the implementation of an MQC policy. In this task the question asks to classify traffic flows from web servers in VLAN 146, which means that they will be using *source port* 80 in their responses to clients. Additionally the SCAVENGER class-map, which matches IP Precedence 0 traffic from VLAN 146, may overlap other traffic classes, such as the HTTP class, which makes it important that SCAVENGER is called last in the policy-map to match any un-classified traffic up to that point.

Copyright © 2010 Internetwork Expert

Note

To verify this configuration first shutdown R5's Frame Relay link. Next, enable the HTTP server service on R1 and start transferring an IOS image from R1 to SW2, start an IP SLA jitter operation on R6 to source "voice-like" packets with the G.729 codec (60 bytes each), and finally send a large number of ICMP packets from R6 to R5, each larger than 1000 bytes.

```
R1:
ip http server
ip http path flash:

R5:
ip sla responder

R6:
ip sla monitor 1
 type jitter dest-ipaddr 155.1.45.5 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
!
ip sla monitor schedule 1 life forever start-time now
```

```
Rack1SW2#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-
mz.124-24.T.bin null:

Rack1R6#ping 155.1.45.5 repeat 100 size 1004 timeout 0

Type escape sequence to abort.
Sending 100, 1004-byte ICMP Echos to 155.1.45.5, timeout is 0 seconds:
 ..........................
 ..........................
Success rate is 0 percent (0/100)
```

Check the statistics to see the policy-map matches.

```
Rack1R4#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy output: SERIAL_LINK

    Class-map: VOICE (match-all)
      2006 packets, 128384 bytes
      30 second offered rate 20000 bps, drop rate 0 bps
      Match: access-group name VOICE
      Match: packet length min 60 max 60
      QoS Set
        dscp ef
          Packets marked 2006

    Class-map: HTTP (match-all)
      899 packets, 521420 bytes
      30 second offered rate 75000 bps, drop rate 0 bps
      Match: access-group name HTTP
      QoS Set
        precedence 2
          Packets marked 899

    Class-map: LARGE_ICMP (match-all)
      100 packets, 100800 bytes
      30 second offered rate 24000 bps, drop rate 24000 bps
      Match: protocol icmp
      Match: packet length min 1001
      drop

    Class-map: SCAVENGER (match-all)
      2 packets, 168 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: input-interface FastEthernet0/1
      Match: ip precedence 0
      QoS Set
        precedence 1
          Packets marked 2

    Class-map: class-default (match-any)
      8 packets, 1568 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

Note that all MQC configurations use the same unified syntax for configuration and verification.

Copyright © 2010 Internetwork Expert

10.34 MQC Bandwidth Reservations and CBWFQ

- Modify R4's MQC policy to meet the following requirements:
  - Set the total size of the MQC queue to 512 buffers
  - The traffic flows for the web servers in VLAN 146 and the IP Precedence 0 packets from VLAN 146 should be guaranteed 32Kbps each
  - Limit the sizes of the FIFO queues for the HTTP and IP Precedence 0 traffic classes to 16 and 24 packets respectively
  - All other unmatched traffic in the policy should run WFQ
  - Dynamic flows in this WFQ should start dropping when they reach 32 packets in length

Configuration

```
R4:
interface Serial 0/1/0
 bandwidth 128
 max-reserved-bandwidth 100
 no fair-queue
 hold-queue 512 out
!
policy-map SERIAL_LINK
 class VOICE
 class HTTP
  bandwidth 32
  queue-limit 16
 class SCAVENGER
  bandwidth 32
  queue-limit 24
 class class-default
  fair-queue
  queue-limit 32
```

Copyright © 2010 Internetwork Expert

Verification

Note

Class-Based Weighted Fair Queueing (CBWFQ) is the MQC equivalent of a combination of the legacy interface level **fair-queue** command and the custom-queue. CBWFQ is used to reserve a *minimum* amount of bandwidth in the output queue for a particular traffic flow in the case of congestion. The **bandwidth** statement used in the policy-map for the class is used to compute the scheduling weight of the traffic flow, relative to the configured interface bandwidth.

The logic of CBWFQ is that during congestion, a class with a bandwidth reservation of *ClassBandwidth* will have at least a *ClassBandwidth/InterfaceBandwidth* share of the total interface bandwidth. For this reason it is key that the interface level **bandwidth** statement is configured correctly, in respect to the access rate or guaranteed rate of the link in question whenever an MQC policy is applied. The **bandwidth** statement under the policy-map does not have a built-in policer, which means that if congestion is not occurring, the traffic flow matched by the class can use as much bandwidth as it wants. Based on this fact the CBWFQ reservation only becomes active once congestion occurs. Configuration of this feature is much more straightforward than the legacy custom-queue, as the bandwidth reserved is not based on a relative ratio like it is in the legacy version.

The feature is deemed Class-Based Weighted Fair Queueing, because the WFQ flow is manually defined through the MQC class-map. The weighting value for the flow is then defined through the **bandwidth** value under the respective policy-map. Once the policy-map is applied to the interface the entire software queue changes to CBWFQ. The CLI will not allow you to assign a service-policy with CBWFQ weights unless the interface is using FIFO queuing, which implies that CBWFQ is not compatible with any of the legacy queuing methods, such as custom queueing or priority queueing. These must be disabled explicitly before applying the service-policy.

By default, the sum of all configured bandwidth reservations cannot exceed 75% of the interface bandwidth value, and can be modified like the legacy RTP reservations with the **max-reserved-bandwidth** interface command. The system subtracts the bandwidth reserved for a class from the available interface bandwidth for the purpose of admission control. The default limits are designed to leave some best-effort bandwidth to the unclassified traffic, such as routing updates and layer 2 keepalives, which fall into the class-default of the policy-map.

Copyright © 2010 Internetwork Expert

Each class configured with a **bandwidth** statement under the policy-map has its own dedicated FIFO queue in the interface's CBWFQ conversation pool. The depth of each FIFO queue can be changed on a per-class basis with the **queue-limit** command. The overall WFQ settings, such as the CDT and the total queue size, can be set using the **queue-limit** under class-default, and the **hold-queue out** command at the interface level.

Recall that with legacy WFQ, flows are classified automatically. The scheduler shares the bandwidth in proportion to the flow's IP precedence value, normalized against the sum of other flows' precedences. Specifically if there are *N* active flows, and flow *i* has IP Precedence IPP(*i*), then any flow *k* is guaranteed a share as:

Share(k) = (IPP(k)+1)/(IPP(1)+IPP(2)…+IPP(N)+N)*100%

Note that each precedence value is shifted by 1 to ensure non-zero values. Since WFQ implements max-min sharing, any flow may claim unused interface bandwidth. Finally, the implementation uses a computation "weight" value based on the formula:

Weight(i) = 32384/(IPP(i)+1) **[Formula 1]**

This allows for the flow with higher IP precedence to have lower computation weight, and larger share of bandwidth. Remember that computation weights are inversely proportional to the actual weights you use to compute the share of the bandwidth, as the bandwidth is shared in the proportion [1/Weight(1):1/(Weight(2)… :1/Weight(N)]. For example, WFQ assigns a weight of 4048 to a flow with an IP precedence of 7, which is the maximum IPP value. This is an important fact that will be referenced again later.

With the new MQC format, CBWFQ retains the above computations for *unclassified* traffic only. Traffic flows that are explicitly matched in a class-map and have a bandwidth reservation configured are treated differently. To start, this class uses a separate flow queue with its own limits. Secondly, this flow is assigned a computation weight based on the following formula:

Weight(i) = Const*InterfaceBandwidth/Bandwith(i) **[Formula 2]**

Here *Bandwidth(i)* is the value configured under the respective class using the **bandwidth** keyword, and *InterfaceBandwidth* is the value configured under the main interface using the **bandwidth** keyword. The value of the constant *Const* depends on the number of flow queues active in CBWFQ, and varies from 1 to 64. The implications of the above formula's calculations are as follows.

Copyright © 2010 Internetwork Expert

Almost any manually configured class has a weight value significantly lower than any automatically classified flow. Recall that IP Precedence 7, the best value, has a best possible weight of 4048 for a dynamic conversation. However for a manually defined class, even in the worst case, *Const* equals 64 (e.g. with 32 flow-queues), the ratio of *interfaceBandwidth/Bandwidth* should be less than 0.02 to be comparable to 4048. This means that a class should have less than a 2% reservation of the interface's bandwidth, which is rare. Furthermore since any user-defined class has its own separate FIFO queue, we can conclude that CBWFQ isolates it from congestive discard drops performed across dynamic WFQ queues. Instead, each class queue is FIFO with a tail-drop discipline by default.

The next question is how CBWFQ shares bandwidth between the user-defined classes. Looking at the weight calculation we can assume that each class has a relative bandwidth share of

Share(i) = Bandwidth(i)/InterfaceBandwidth.

Recall that by default, the sum of all *Bandwidth(i)* reservation is no more than 75% of *InterfaceBandwidth*, due to the **`max-reserved-bandwidth`** defaults. So what happens with the remaining 25%? This is where class-default comes in.

As soon as the **`bandwidth`** keyword is specified under any user-defined class, the interface queue turns into CBWFQ. This means any unmatched flows that fall back into class-default are scheduled using dynamic WFQ weights. This means that automatic classification occurs, along with precedence-based weight assignment and sharing of the single buffer space of WFQ. This behavior is default, even if you did not configure **`fair-queue`** under the class-default.

In addition to dynamic flow queues, WFQ and CBWFQ both have the concept of Link Queues. The system uses these queues for critical control plane traffic, such as routing updates and layer 2 keepalives. Each Link Queue has computation weight of 1024, which is better than any dynamic queue's weight. By default, CBWFQ matches all control plane traffic to class-default, so it is reasonable to assume that some bandwidth (25% by default) is saved for those flows. This safeguard measure prevents user-defined classes from oversubscribing the interface bandwidth, and does not let the other important queues starve.

Copyright © 2010 Internetwork Expert

If you want to disable fair-queue for unclassified packets, an explicit **bandwidth** value for the class-default can be configured, which turns it into a single FIFO queue. For example, the following code snippet disables fair-queue for unclassified traffic, and gives it a static weight as relative to 96Kbps:

```
policy-map TEST
 class class-default
   bandwidth 96
```

Moreover if you do not define any classes other than class-default, and class-default has a bandwidth value defined, the entire interface queue essentially becomes a FIFO queue.

The following tables illustrate the weight values that are used for CBWFQ. Here *N* is the constant that defines the number of dynamic WFQ queues Their number is always a power of 2, and equals 2^N. Normally the IOS automatically calculates this based on the interface's bandwidth value, but it can be manually specified with the **fair-queue** command in class-default.

Copyright © 2010 Internetwork Expert

Conversation Numbers	CBWFQ Weight	Description
Below 2^N	32384/(IPP+1)	Used for automatic classification of dynamic WFQ Queues. Configurable via **fair-queue** command under "class-default".
2^N…2^N+7	1024	Link Queues. There are 8 conversations used to queue routing updates (marked as *PAK_PRIORITY* internally) and Layer 2 keepalives.
2^N+8	0	Priority queue which maps directly to legacy WFQ IP RTP Priority. Typically used for VoIP bearer traffic. CBWFQ polices this flow using a configurable token bucket procedure to ensure it does not starve other queues.
Above 2^N+8	Const*IntfBW/ClassBW OR RSVP flow weight	These conversations are used for user-defined traffic classes. Each class has its own FIFO queue and configurable queue depth. In addition, RSVP flows use the same range of conversation numbers.

The "N" constant	Number of dynamic Flows	CBWFQ constant "Const"
4	16	64
5	32	64
6	64	57
7	128	30
8	256	16
9	512	8
10	1024	4
11	2048	2
12	4096	1

Copyright © 2010 Internetwork Expert

In summary, the key point about CBWFQ is that it uses the same scheduling logic as the legacy WFQ, but user-configurable classes have a special low-weight, making them more important than any dynamic conversation. Thus, user-defined classes always get the guaranteed proportion of bandwidth, while flows using class-default may starve in the case of congestion.

For verification of this task shut down R5's Frame Relay link to force the traffic from VLAN 146 to the hosts behind R4 to take the path across the serial link. R6 should send IP SLA jitter probes to R5 across the serial link; this flow simulates voice traffic and has no explicit bandwidth set under the class. Next, configure R1 as an HTTP server, and set SW2 to transfer the IOS image from R1, oversubscribing the serial link. Lastly, generate a flow of ICMP packets from R6 to R5, simulating the SCAVENGER class traffic.

```
R1:
ip http server
ip http path flash:

R4:
interface Serial 0/1/0
 load-interval 30

R5:
ip sla responder
!
interface Serial 0/0/0
 shutdown

R6:
ip sla monitor 1
 type jitter dest-ipaddr 155.1.45.5 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
!
ip sla monitor schedule 1 life forever start-time now
```

```
Rack1SW2#copy http://admin:cisco@155.1.146.1/c2600-adventerprisek9-
mz.124-10.bin null:

Rack1R1#ping 155.1.45.5 repeat 100000 size 150

Type escape sequence to abort.
Sending 100000, 150-byte ICMP Echos to 155.1.45.5, timeout is 2
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Check the policy-map matches and observe the measured traffic rates. Note that the load-interval is set to 30 seconds to ensure more adaptive response to changes in traffic patterns.

```
Rack1R4#show policy-map interface serial 0/1/0
 Serial0/1/0
  Service-policy output: SERIAL_LINK

    Class-map: VOICE (match-all)
      173240 packets, 11087360 bytes
      30 second offered rate 23000 bps, drop rate 0 bps
      Match: access-group name VOICE
      Match: packet length min 60 max 60
      QoS Set
        dscp ef
          Packets marked 173241

    Class-map: HTTP (match-all)
      45477 packets, 26371423 bytes
      30 second offered rate 104000 bps, drop rate 0 bps
      Match: access-group name HTTP
      QoS Set
        precedence 2
          Packets marked 45477
      Queueing
        Output Queue: Conversation 41
        Bandwidth 32 (kbps)Max Threshold 16 (packets)
        (pkts matched/bytes matched) 6020/3491600
        (depth/total drops/no-buffer drops) 6/0/0

    Class-map: LARGE_ICMP (match-all)
      0 packets, 0 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: protocol icmp
      Match: packet length min 1001
      drop

    Class-map: SCAVENGER (match-all)
      38429 packets, 5905886 bytes
      30 second offered rate 15000 bps, drop rate 0 bps
      Match: input-interface FastEthernet0/1
      Match: ip precedence 0
      QoS Set
        precedence 1
          Packets marked 104852
      Queueing
        Output Queue: Conversation 42
        Bandwidth 32 (kbps)Max Threshold 24 (packets)
        (pkts matched/bytes matched) 2286/351274
        (depth/total drops/no-buffer drops) 0/0/0

    Class-map: class-default (match-any)
      908 packets, 205544 bytes
      30 second offered rate 0 bps, drop rate 18000 bps
      Match: any
      Queueing
        Flow Based Fair Queueing
        Maximum Number of Hashed Queues 32
        (total queued/total drops/no-buffer drops) 64/9916/0
```

Copyright © 2010 Internetwork Expert

Note that the conversation numbers for the user-defined classes are 41 and 42, which means that there are 32 dynamic queues plus 8 link queues. The IOS picked the number of dynamic queues automatically based on interface bandwidth of 128Kbps.

Next, the class for VOICE traffic demonstrates a measured bitrate close to 24Kbps. However, we also see the high drop rate under class-default, along with its queue stats. Since the VOICE class has no weight of its own, CBWFQ assigns it to a dynamic queue.

Two other user-configured classes (HTTP and SCAVENGER) have equal bandwidth weights configured, but the SCAVENGER class is not consuming all of its allocated bandwidth. This is why the HTTP class uses the unclaimed resources. Finally, since both user configurable classes have better weights than the WFQ dynamic queues, the VOICE traffic cannot get enough bandwidth. Thus, CBWFQ drops most of the VOICE class traffic.

Now let's see the CBWFQ queue contents:

```
Rack1R4#show queueing interface serial 0/1
Interface Serial0/1 queueing strategy: fair
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 11322
  Queueing strategy: Class-based queueing
  Output queue: 12/512/32/0 (size/max total/threshold/drops)
     Conversations  3/4/32 (active/max active/max total)
     Reserved Conversations 3/3 (allocated/max allocated)
     Available Bandwidth 40 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 6/256/0/0/0
  Conversation 41, linktype: ip, length: 580
  source: 155.1.146.1, destination: 155.1.58.8, id: 0x74CD, ttl: 254,
  TOS: 64 prot: 6, source port 80, destination port 11003

  (depth/weight/total drops/no-buffer drops/interleaves) 1/256/0/0/0
  Conversation 42, linktype: ip, length: 154
  source: 155.1.146.6, destination: 155.1.45.5, id: 0x874A, ttl: 254, prot: 1

  (depth/weight/total drops/no-buffer drops/interleaves) 5/5397/601/0/0
  Conversation 5, linktype: ip, length: 64
  source: 155.1.146.6, destination: 155.1.45.5, id: 0x02B8, ttl: 254,
  TOS: 184 prot: 17, source port 50700, destination port 16384
```

There are three flows active in the queue. The total queue length is 512, which is what we set the hold-queue to. A total of 12 packets are currently queued up. Note that packet sizes include layer 2 overhead, which is 4 bytes for HDLC.

Copyright © 2010 Internetwork Expert

The first displayed flow corresponds to the HTTP traffic, and has a maximum accumulated queue depth compared to the other flows. Its weight is 256, which corresponds to 25% of the interface bandwidth, or 32Kbps (note that CBWFQ does not take the IP Precedence of the HTTP traffic into account). The ICMP traffic has the same weight, while the flow corresponding to VOICE class has conversation number 5. This means it uses a dynamic queue for scheduling, and its weight of 5397 is actually 32384/(5+1) – the automatic weight corresponding to IP Precedence 5 traffic. We can calculate the guaranteed bandwidth shares based on observed weights using the following computations.

First, the bandwidth is shared in values inversely proportional to computation weights:

1/256:1/256:1/5397.

Normalize the proportion by dividing by the smallest number:

5397/256:5397/256:1 = 21:21:1.

Now the actual bandwidth shares are (roughly):

[21/(21+21+1), 21/(21+21+1), 1/(21+21+1)] * 100% = 49% 49% 2%.

Therefore the net effect is that the available bandwidth is almost equally divided between the two user-defined classes, and the dynamic conversation gets a much smaller share. To confirm this, verify the IP SLA statistics on R6, and note that conversation statistics for the VOICE traffic exhibits the largest number of packet drops.

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip sla monitor statistics 1
Round trip time (RTT)   Index 1
        Latest RTT: 6 ms
Latest operation start time: 04:09:04.672 UTC Fri Aug 15 2008
Latest operation return code: OK
RTT Values
        Number Of RTT: 29
        RTT Min/Avg/Max: 37/228/434 ms
Latency one-way time milliseconds
        Number of one-way Samples: 0
        Source to Destination one way Min/Avg/Max: 0/0/0 ms
        Destination to Source one way Min/Avg/Max: 0/0/0 ms
Jitter time milliseconds
        Number of Jitter Samples: 27
        Source to Destination Jitter Min/Avg/Max: 5/13/17 ms
        Destination to Source Jitter Min/Avg/Max: 1/1/3 ms
Packet Loss Values
        Loss Source to Destination: 655         Loss Destination to Source: 64
        Out Of Sequence: 0      Tail Drop: 252  Packet Late Arrival: 64
Voice Score Values
        Calculated Planning Impairment Factor (ICPIF): 40
MOS score: 2.75
Number of successes: 136
Number of failures: 18
Operation time to live: Forever
```

Even though the traffic is marked with IP Precedence 5, CBWFQ penalizes the flow as compared to the other user-defined classes. The user-defined flows always get what they ask for, and are always preferred over the dynamic conversations.

Copyright © 2010 Internetwork Expert

10.35 MQC Bandwidth Percent

- Modify R4's previous user-defined reservations to use a percentage reservation of 25% of the link bandwidth each, as compared to 32Kbps.

Configuration

```
R4:
policy-map SERIAL_LINK
 class HTTP
  no queue-limit
  no bandwidth
  bandwidth percent 25
 class SCAVENGER
  no queue-limit
  no bandwidth
  bandwidth percent 25
```

Verification

Note

Recall that CBWFQ does not use the absolute **bandwidth** value in a class to directly compute the scheduling weight, but instead it is computed relative to the interface's bandwidth value as *Weight(i) = Const*InterfaceBandwidth/Bandwith(i)*. Since this value is in reality a relative reservation in the first place, the calculation can be simplified by ignoring the interface level bandwidth, and expressing the reservation as a percentage. In this manner with the **bandwidth percent** command the weight calculation changes to:

Weight(i) = Const*100/Percent(i)

This implementation is useful in designs where a common template of reservation is applied to multiple links of differing bandwidth values. For example a standard policy-map could be defined to reserve 20% of link bandwidth for HTTP flows, and then applied to both 100Mbps FastEthernet and 45Mbps DS3. The resulting CBWFQ weights would be the same, but the actual bandwidth value differs based on the underlying physical link bandwidth.

Like the Kbps reservation through the **bandwidth** command, the sum of all configured **bandwidth percent** values in all classes of a policy-map cannot exceed the **max-reserved-bandwidth** configured on the respective interface. In addition, it is not possible to mix the **bandwidth** and **bandwidth percent** commands in the same policy-map; the units must be the same among classes.

Copyright © 2010 Internetwork Expert

Verification of this configuration is the same as the previous task. Shut down R5's Frame Relay link to force the traffic from VLAN 146 to the hosts behind R4 to take the path across the serial link. R6 should send IP SLA jitter probes to R5 across the serial link; this flow simulates voice traffic and has no explicit bandwidth set under the class. Next, configure R1 as an HTTP server, and set SW2 to transfer the IOS image from R1, oversubscribing the serial link. Lastly, generate a flow of ICMP packets from R6 to R5, simulating the SCAVENGER class traffic.

```
R1:
ip http server
ip http path flash:

R4:
interface Serial 0/1/0
 load-interval 30

R5:
ip sla responder
!
interface Serial 0/0/0
 shutdown

R6:
ip sla monitor 1
 type jitter dest-ipaddr 155.1.45.5 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
!
ip sla monitor schedule 1 life forever start-time now
```

```
Rack1SW2#copy http://admin:cisco@155.1.146.1/c1841-adventerprisek9-
mz.124-24.T.bin null:

Rack1R6#ping 155.1.45.5 repeat 100000 size 150

Type escape sequence to abort.
Sending 100000, 150-byte ICMP Echos to 155.1.45.5, timeout is 2
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Check the policy-map statistics as follows.

```
Rack1R4#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy output: SERIAL_LINK

    Class-map: VOICE (match-all)
      183205 packets, 11725120 bytes
      30 second offered rate 23000 bps, drop rate 0 bps
      Match: access-group name VOICE
      Match: packet length min 60 max 60
      QoS Set
        dscp ef
          Packets marked 183205

    Class-map: HTTP (match-all)
      50438 packets, 29248267 bytes
      30 second offered rate 105000 bps, drop rate 0 bps
      Match: access-group name HTTP
      QoS Set
        precedence 2
          Packets marked 50438
      Queueing
        Output Queue: Conversation 41
        Bandwidth 25 (%)
        Bandwidth 32 (kbps)Max Threshold 64 (packets)
        (pkts matched/bytes matched) 361/208844
        (depth/total drops/no-buffer drops) 2/0/0

    Class-map: LARGE_ICMP (match-all)
      0 packets, 0 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: protocol icmp
      Match: packet length min 1001
      drop

    Class-map: SCAVENGER (match-all)
      41195 packets, 6331150 bytes
      30 second offered rate 16000 bps, drop rate 0 bps
      Match: input-interface FastEthernet0/1
      Match: ip precedence 0
      QoS Set
        precedence 1
          Packets marked 107619
      Queueing
        Output Queue: Conversation 42
        Bandwidth 25 (%)
        Bandwidth 32 (kbps)Max Threshold 64 (packets)
        (pkts matched/bytes matched) 196/30114
        (depth/total drops/no-buffer drops) 1/0/0

    Class-map: class-default (match-any)
      965 packets, 218264 bytes
      30 second offered rate 0 bps, drop rate 17000 bps
      Match: any
      Queueing
        Flow Based Fair Queueing
        Maximum Number of Hashed Queues 32
        (total queued/total drops/no-buffer drops) 64/18239/0
```

This output displays the percent values along with inferred absolute bandwidth reservation, where bandwidth = percent*interface_bandwidth. Other than this the bandwidth shares remain the same. Next look at the queue content and note the CBWFQ weights.

```
Rack1R4#sh policy-map interface Serial 0/1/0

 Serial0/1/0

  Service-policy output: SERIAL_LINK

    Class-map: VOICE (match-all)
      68063 packets, 4356032 bytes
      30 second offered rate 24000 bps, drop rate 0 bps
      Match: access-group name VOICE
      Match: packet length min 60 max 60
      QoS Set
        dscp ef
          Packets marked 68063

    Class-map: HTTP (match-all)
      19390 packets, 11242130 bytes
      30 second offered rate 89000 bps, drop rate 0 bps
      Match: access-group name HTTP
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 5/1/0
      (pkts output/bytes output) 6581/3816444
      QoS Set
        precedence 2
          Packets marked 19390
      bandwidth 25% (32 kbps)

    Class-map: LARGE_ICMP (match-all)
      100 packets, 100800 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: protocol icmp
      Match: packet length min 1001
      drop

    Class-map: SCAVENGER (match-all)
      2105 packets, 319020 bytes
      30 second offered rate 8000 bps, drop rate 0 bps
      Match: input-interface FastEthernet0/1
      Match: ip precedence 0
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 637/97188
      QoS Set
        precedence 1
          Packets marked 2105
      bandwidth 25% (32 kbps)
```

Copyright © 2010 Internetwork Expert

```
Class-map: class-default (match-any)
  355 packets, 142676 bytes
  30 second offered rate 0 bps, drop rate 0 bps
  Match: any
  Queueing
  queue limit 32 packets
  (queue depth/total drops/no-buffer drops/flowdrops) 2/11/0/0
  (pkts output/bytes output) 60470/4809582
  Fair-queue: per-flow queue limit 8
```

Just as before, flows 41 and 42 are assigned weights of 256, while the VOICE flow with a weight of 5397 is heavily penalized as compared to the user-defined classes.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.36 MQC LLQ and Remaining Bandwidth Reservations

- Modify R4's policy to R5 as follows:
  - Configure Low Latency Queueing to allow for exactly one VoIP call to be prioritized
  - Additional VoIP calls should be allowed only if the link is not congested, but they should not be prioritized
  - Reserve 33% of the remaining bandwidth to the HTTP and SCAVENGER classes each
- Assume that VoIP calls are using the G.729 codec, which generates 50 packets per second with a Layer 3 size of 60 bytes.

Configuration

```
R4:
policy-map SERIAL_LINK
 class VOICE
  priority 27
 class HTTP
  no bandwidth
  bandwidth remaining percent 33
 class SCAVENGER
  no bandwidth
  bandwidth remaining percent 33
```

Verification

Note

The Low Latency Queuing (LLQ) feature is the MQC equivalent of the legacy IP RTP Priority, however classification is not limited to just RTP packets. Like IP RTP Priority, LLQ uses the special conversation number 2^N+8, where *N* defines the number of dynamic conversations. For example, if there are 32 (2^5) dynamic queues, then the LLQ conversation is number 40. This conversation has a weight value of 0, which means that it is always serviced first. In order to prevent starvation of other queues, the packets de-queued from the LLQ conversation are metered using a simple token bucket, with a configurable rate and burst size. Packets that exceed the token bucket are dropped (policed) during times of congestion, while if there is no congestion, exceeding traffic is not dropped, but it is simply not prioritized. For these reasons it is better to use the MQC LLQ feature in practical designs versus the legacy priority-queue or IP RTP Priority, because the legacy priority-queue does not have a policer, IP RTP Priority can not classify other traffic flows, and neither of them can be used in conjunction with other QoS features, such as a bandwidth reservation.

Copyright © 2010 Internetwork Expert

Multiple classes inside a single policy-map can use the priority keyword, but only one single priority queue exists. This design of multiple priority classes is used to ensure that one priority flow does not starve another priority flow. For example assume that VoIP and Video traffic are grouped together in one class-map, which has a **priority 128** value defined under a policy-map. If Video traffic is using 128Kbps, VoIP traffic can potentially be dropped (policed), or at least not be guaranteed priority. This is because both traffic flows must contend for the same 128Kbps rate. However if separate VoIP and Video class-maps are defined, each with a **priority 64** rate configured under the same policy-map, each flow is guaranteed priority up to 64Kbps. Although this still totals 128Kbps of priority, one class can't completely starve the other.

Note that the LLQ policer takes the layer 2 packet length into account, so in this case the HDLC overhead of 7 bytes is added to the 60 bytes of layer 3 VoIP payload, which results in a value close to 27Kbps (67 bytes/packet * 50 packets/second * 8 bits/byte = 26800bps) being reserved. The burst value for the token bucket can be configured manually, or automatically calculated by the IOS itself. A burst size of 1 or 1.5 seconds should be enough, since this is probably the maximum duration of a single spoken word in VoIP. See the *Further Learning* section following the verification for more info about computing LLQ sizes for VoIP calls.

Once the LLQ priority reservation is configured, the CBWFQ algorithm subtracts the reserved bandwidth of the priority queue from the interface's available bandwidth. Like in IP RTP Priority, the available bandwidth value defaults to 75% of the interface bandwidth, and can be modified with the **max-reserved-bandwidth** interface level command. The remaining bandwidth can be used to create a relative bandwidth reservation for other classes in the CBWFQ. By this logic, instead of configuring absolute bandwidth values - either numerically or in interface bandwidth percents - for other reservations, you can just specify relative shares of the bandwidth that remains after the priority queue finished its run. This method is seen in the above configuration through the **bandwidth remaining percent** command.

For example in this particular task the link between R4 and R5 was configured with a interface **bandwidth** value of 128Kbps, and a **max-reserved-bandwidth** of 100, meaning that 100% of 128Kbps is reservable. With the 27Kbps reservation for VoIP in the LLQ, 101Kbps is the remaining bandwidth. Since the HTTP and SCAVENGER classes reserve 33% of the remaining bandwidth, they are each allocated a minimum of 33Kbps ((128-27) * .33 = 33).

Copyright © 2010 Internetwork Expert

To verify this configuration generate three different traffic flows, like in the previous examples. First, shutdown the Frame Relay interface of R5 to force the traffic from VLAN 146 to hosts behind R5 to take path across the serial link. Configure R6 to send IP SLA jitter probes to R5; this flow simulates the voice traffic in the LLQ. Next, configure R1 as an HTTP server, and set SW2 to transfer the IOS image from R1, causing the serial link to be congested. Lastly, generate a flow of ICMP packets from R6 to R5, simulating the SCAVENGER class traffic.

```
R6:
ip http server
ip http path flash:

R4:
interface Serial 0/1/0
 load-interval 30

R5:
ip sla responder
!
interface Serial 0/0/0
 shutdown

R1:
ip sla monitor 1
 type jitter dest-ipaddr 155.1.45.5 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
!
ip sla monitor schedule 1 life forever start-time now
```

```
Rack1SW2#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-
mz.124-24.T.bin null:

Rack1R6#ping 155.1.45.5 repeat 100000 size 150

Type escape sequence to abort.
Sending 100000, 150-byte ICMP Echos to 155.1.45.5, timeout is 2
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Copyright © 2010 Internetwork Expert

Check the statistics for policy map matches. Note that the priority queue for VOICE traffic uses conversation 40, and its 30 seconds offered rate is 25Kbps. Since the voice packets use the priority queue now, the remaining classes have less bandwidth to share. Therefore the bandwidth measured for the HTTP class is only 78Kbps, compared to the higher value available seen in prior configurations without the LLQ. Also, note that SCAVENGER class does not use all its guaranteed bandwidth, so the HTTP class can use the remaining part.

```
Rack1R4#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy output: SERIAL_LINK

    Class-map: VOICE (match-all)
      26349 packets, 1686336 bytes
      30 second offered rate 25000 bps, drop rate 0 bps
      Match: access-group name VOICE
      Match: packet length min 60 max 60
      QoS Set
        dscp ef
          Packets marked 26349
      Queueing
        Strict Priority
        Output Queue: Conversation 40
        Bandwidth 27 (kbps) Burst 3400 (Bytes)
        (pkts matched/bytes matched) 26274/1681536
        (total drops/bytes drops) 0/0

    Class-map: HTTP (match-all)
      9491 packets, 5504156 bytes
      30 second offered rate 78000 bps, drop rate 0 bps
      Match: access-group name HTTP
      QoS Set
        precedence 2
          Packets marked 9491
      Queueing
        Output Queue: Conversation 41
        Bandwidth remaining 33 (%)Max Threshold 64 (packets)
        (pkts matched/bytes matched) 3092/1793048
        (depth/total drops/no-buffer drops) 6/0/0

    Class-map: LARGE_ICMP (match-all)
      0 packets, 0 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: protocol icmp
      Match: packet length min 1001
      drop
```

Copyright © 2010 Internetwork Expert

```
Class-map: SCAVENGER (match-all)
  8739 packets, 1343986 bytes
  30 second offered rate 19000 bps, drop rate 0 bps
  Match: input-interface FastEthernet0/1
  Match: ip precedence 0
  QoS Set
    precedence 1
      Packets marked 75162
  Queueing
    Output Queue: Conversation 42
    Bandwidth remaining 33 (%)Max Threshold 64 (packets)
    (pkts matched/bytes matched) 2843/437262
    (depth/total drops/no-buffer drops) 1/0/0

Class-map: class-default (match-any)
  139 packets, 31440 bytes
  30 second offered rate 0 bps, drop rate 0 bps
  Match: any
  Queueing
    Flow Based Fair Queueing
    Maximum Number of Hashed Queues 32
    (total queued/total drops/no-buffer drops) 0/0/0
```

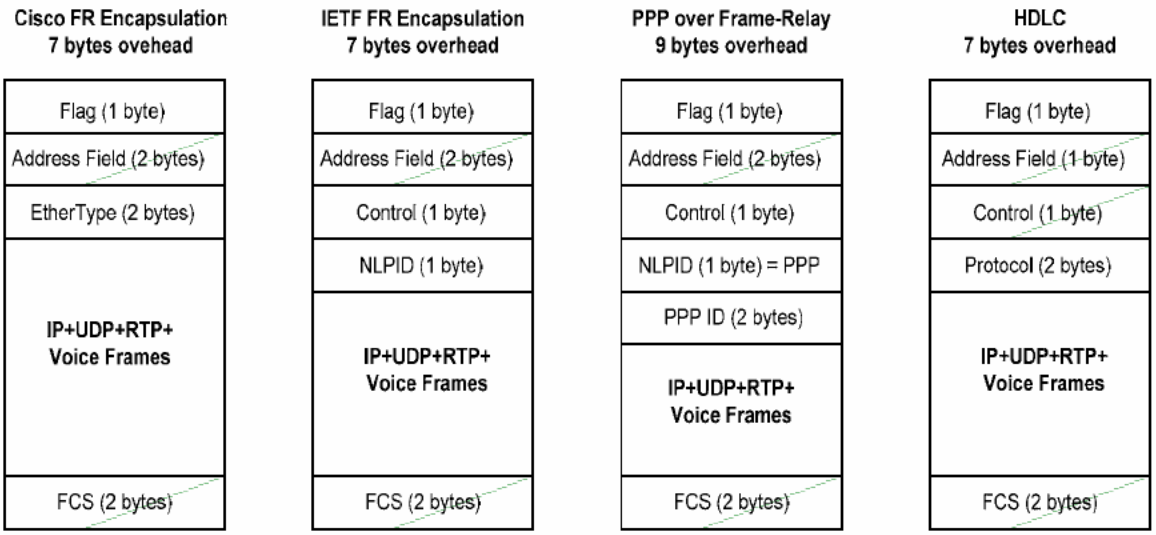
Further Learning

Computing voice bandwidth is usually required for scenarios where you provision an LLQ based on the number of calls and VoIP codec used. To accomplish this you need to account for codec payload rate, the Layer 3 overhead (RTP and UDP headers), and the Layer 2 overhead (Frame-Relay, Ethernet, HDLC etc. headers). Accounting for Layer 2 overhead is important since the LLQ policer takes this overhead in account when enforcing the maximum rate.

For this example let's consider two codecs for bandwidth computation, G.729 and G.711. By default both codecs generate 50 VoIP packets per second, however the codec framing rate is 10ms (100 packets per second). Thus each VoIP packet carries two frames with VoIP samples. The frame sizes are 10 bytes and 80 bytes for G.729 and G.711 respectively. Therefore G.729 generates (10*2)*50*8=8000bps and G.711 generates (80*2)*50*8=64000bps of payload rate.

The RTP header size is 12 bytes, and the UDP header size is 8 bytes. A typical IP header (with no options) is 20 bytes. Therefore, the Layer 3 overhead is 40 bytes, if we don't use header compression.

The following are the formats WAN frames commonly used to transport voice (with or without FRF.12/MLP fragmentation – voice packets are never fragmented).



Copyright © 2010 Internetwork Expert

As we can see both Cisco and IETF Frame-Relay encapsulations add 7 bytes of Layer 2 overhead to VoIP packets. The same holds true for HDLC encapsulation (which is not very common but added here for sake of completeness). PPP over Frame-Relay adds 9 bytes of overhead – the maximum overhead of all encapsulation types.

Using the information above, you can compute bandwidth usage for uncompressed voice traffic flow across any WAN connection. For example, let's compute the bandwidth consumption for 3 G.729 calls across Frame-Relay link with FRF.12 fragmentation. First off, FRF.12 does not fragment voice packets if set up properly, as the fragment size is greater than or equal to the VoIP packet. Thus we assume 7 bytes of Layer 2 overhead for any of the Frame Relay encapsulation types. Next, the size of the payload + Layer 3 overhead is 20 bytes + 40 bytes = 60 bytes. Based on the 50pps rate we end up with the bandwidth value of (20+40+7)*50*8=26800bps. If you want to use the G.711 codec, then replace the 20 bytes payload with 160 bytes. The result is (160+40+7)*50*8=82800bps.

Another thing to consider is IP/RTP/UDP header compression. Cisco's implementation reduces the total overhead of 40 bytes (12+8+20) down to 2 bytes (no UDP checksum). This limits the Layer 3 overhead to just 2 bytes. Based on this reduction from 40 to 2 bytes we could compute the bandwidth usage for a G.729 call over MLPoFR with UDP header compression as (20+2+9)*50*8=12400bps. The same computations for compressed G.729 over Frame Relay with or without FRF.12 yields (20+2+7)*50*8=11600bps.

For VoIP over Ethernet the Layer 2 overhead for is typically 18 bytes – 14 bytes for the Ethernet header and 4 bytes for FCS (32 bits). If the frame carries a VLAN tag, add another 4 bytes, for 22 bytes of total overhead. Note that you typically see the G.711 codec used over LAN links.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.37 MQC WRED

- Modify R4's policy so that the HTTP traffic class uses random detection for dropping as opposed to tail drop.
- Start dropping packets randomly when the average queue size is between 4 and 16.
- Drop 1 of every 4 packets when the average queue size reaches the maximum threshold.

Configuration

```
R4:
policy-map SERIAL_LINK
 class HTTP
  random-detect
  random-detect precedence 2 4 16 4
```

Verification

Note

CBWFQ supports three different drop policies, classic tail-drop, which is the default for user-defined classes, Congestive Discard for WFQ, and Random Early Detection. When you apply **`random-detect`** command under a user-defined class, it automatically removes **`queue-limit`** command and enforces RED as the drop policy. When using RED with CBWFQ, each flow is considered an individual FIFO queue. This is similar to flow-based WRED, though the big improvement is the ability to use random drop per flow, not per whole queue.

Similar to legacy per-interface WRED, you can tune settings per IP precedence value or per DSCP. In our case HTTP traffic uses IP precedence of 2, therefore we tune the RED settings for IP precedence 2.

To verify this configuration generate three traffic flows similar to before. The goal is to see random drops occur under the HTTP traffic class. To do this first shutdown the Frame Relay interface of R5 to force the traffic from VLAN146 to the hosts behind R5 take path across the serial link. R6 should generate IP SLA jitter probes to R5 across the serial link from R4 to R5. This flow simulates voice traffic. Next, configure R1 as an HTTP server and set SW2 to transfer the IOS image from R1, oversubscribing the serial link. Lastly, generate a flow of ICMP packets from R6 to R5, simulating the SCAVENGER class traffic.

Copyright © 2010 Internetwork Expert

```
R6:
ip http server
ip http path flash:

R4:
interface Serial 0/1/0
 load-interval 30

R5:
ip sla responder
!
interface Serial 0/0/0
 shutdown

R1:
ip sla monitor 1
 type jitter dest-ipaddr 155.1.45.5 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
!
ip sla monitor schedule 1 life forever start-time now
```

```
Rack1SW2#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-
mz.124-24.T.bin null:

Rack1R6#ping 155.1.45.5 repeat 100000 size 150

Type escape sequence to abort.
Sending 100000, 150-byte ICMP Echos to 155.1.45.5, timeout is 2
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Check the statistics for the service policy. Note the long detailed output for the HTTP class. It demonstrates the number of random drops occurred in the queue for IP traffic marked with IP precedence of 2.

```
Rack1R4#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy output: SERIAL_LINK

    Class-map: VOICE (match-all)
      255190 packets, 16332160 bytes
      30 second offered rate 23000 bps, drop rate 0 bps
      Match: access-group name VOICE
      Match: packet length min 60 max 60
      QoS Set
        dscp ef
          Packets marked 255190
      Queueing
        Strict Priority
        Output Queue: Conversation 40
        Bandwidth 26 (kbps) Burst 3200 (Bytes)
        (pkts matched/bytes matched) 67601/4326464
        (total drops/bytes drops) 0/0
```

```
    Class-map: HTTP (match-all)
      76985 packets, 44639911 bytes
      30 second offered rate 77000 bps, drop rate 2000 bps
      Match: access-group name HTTP
      QoS Set
        precedence 2
          Packets marked 76985
      Queueing
        Output Queue: Conversation 41
        Bandwidth remaining 33 (%)
        (pkts matched/bytes matched) 24482/14195104
        (depth/total drops/no-buffer drops) 0/16/0
         exponential weight: 9
         mean queue depth: 0

  class    Transmitted       Random drop      Tail drop    Minimum Maximum  Mark
           pkts/bytes        pkts/bytes       pkts/bytes    thresh  thresh  prob
      0       0/0               0/0              0/0           20      40  1/10
      1       0/0               0/0              0/0           22      40  1/10
      2     713/408916        16/9280            0/0            4      16  1/4
      3       0/0               0/0              0/0           26      40  1/10
      4       0/0               0/0              0/0           28      40  1/10
      5       0/0               0/0              0/0           30      40  1/10
      6       0/0               0/0              0/0           32      40  1/10
      7       0/0               0/0              0/0           34      40  1/10
   rsvp       0/0               0/0              0/0           36      40  1/10

<snip>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.38 MQC Dynamic Flows and WRED

- Modify the CBWFQ configuration on R4 to activate random drops for unclassified traffic's dynamic flows.
- Change the minimum and maximum RED thresholds for traffic with an IP precedence of one to 1 and 40 respectively.
- As the queue depth grows close to the maximum threshold, the probability of packet discard should be 25%.

Configuration

```
R4:
policy-map SERIAL_LINK
 class class-default
  fair-queue
  no queue-limit
  random-detect
  random-detect precedence 1   1     40    4
```

Verification

Note

There are two ways to enable WRED within class-default. The first is to configure a **bandwidth** reservation statement - turning the class's queue into a FIFO queue - and then enabling RED, and the second is to enable RED with WFQ. The second case activates RED dropping to replace Congestive Discard Threshold-based drops for dynamic flows. Think of this as a closer equivalent to flow-based RED, with automatic flow classification. All other WRED parameters remain the same as in the legacy case (e.g. averaging exponential factor, thresholds, etc).

Note that CBWFQ classifies the special *link queues* under class-default, which are therefore subject to WRED drop as soon as it's enabled. However, the default thresholds are high enough for the link queues to prevent the drops of critical control plane traffic.

To verify this configuration generate three traffic flows similar to before. The goal is to see random drops occur under the HTTP traffic class. To do this first shutdown the Frame Relay interface of R5 to force the traffic from VLAN146 to the hosts behind R5 take path across the serial link. R6 should generate IP SLA jitter probes to R5 across the serial link from R4 to R5. This flow simulates voice traffic. Next, configure R1 as an HTTP server and set SW2 to transfer the IOS image from R1, oversubscribing the serial link. Lastly, generate an ICMP traffic flow from SW1 to R5, marking it with IP precedence of 1 to feed packets into class-default.

Copyright © 2010 Internetwork Expert

```
R6:
ip http server
ip http path flash:

R4:
interface Serial 0/1/0
 load-interval 30

R5:
ip sla responder
!
interface Serial 0/0/0
 shutdown

R1:
ip sla monitor 1
 type jitter dest-ipaddr 155.1.45.5 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
!
ip sla monitor schedule 1 life forever start-time now

Rack1SW2#copy http://admin:cisco@155.1.146.1/c2600-adventerprisek9-
mz.124-10.bin null:

Rack1SW1#ping
Protocol [ip]:
Target IP address: 155.1.45.5
Repeat count [5]: 100000000
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface:
Type of service [0]: 32
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100000000, 100-byte ICMP Echos to 155.1.45.5, timeout is 2
seconds:
 ..
```

Verify the WRED statistics for class-default in your policy map. Note that no packets have been dropped due to the high maximum threshold (drop probability is inversely proportional to (max. thresh – min. thresh)). Also note matches for IP precedence 6 traffic, which corresponds to RIP updates that R4 sends to R5.

```
Rack1R4#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy output: SERIAL_LINK

<snip>

    Class-map: class-default (match-any)
      2354 packets, 452176 bytes
      30 second offered rate 2000 bps, drop rate 0 bps
      Match: any
      Queueing
        Flow Based Fair Queueing
        Maximum Number of Hashed Queues 32
        (total queued/total drops/no-buffer drops) 1/0/0
         exponential weight: 9

  class    Transmitted       Random drop      Tail drop    Minimum Maximum  Mark
           pkts/bytes        pkts/bytes       pkts/bytes    thresh  thresh  prob
      0      72/6400             0/0              0/0          20      40  1/10
      1     657/68328            0/0              0/0           1      40  1/4
      2       0/0                0/0              0/0          24      40  1/10
      3       0/0                0/0              0/0          26      40  1/10
      4       0/0                0/0              0/0          28      40  1/10
      5       0/0                0/0              0/0          30      40  1/10
      6      53/22048            0/0              0/0          32      40  1/10
      7       0/0                0/0              0/0          34      40  1/10
   rsvp       0/0                0/0              0/0          36      40  1/10
```

Copyright © 2010 Internetwork Expert

10.39 MQC WRED with ECN

- Modify the HTTP traffic class on R4 so that explicit congestion notification for TCP is used for RED dropping.

Configuration

```
R4:
policy-map SERIAL_LINK
 class HTTP
  random-detect ecn
  random-detect precedence 2   4     16    4
```

Verification

Note

TCP Explicit Congestion Notification (ECN), similar to how BECN and FECN work in Frame Relay, is used to signal the forthcoming of network congestion for TCP flows. Originally TCP detected network congestion based on packet loss, timeouts, and duplicate acknowledgements. This was usually the result of full queues and unconditional packet drops. TCP ECN allows the network to signal the receiver of the flow that the network is close to dropping packets. It's then up to the TCP receiver to decide how to react to this notification, which is usually that it signals the sender to slow down sending rate. The overall effect of TCP ECN is better performance, as compared to simple packet drops and slow start, as it allows the sender to respond fast than slow start would, and results in less time spent on the recovery from a packet loss.

TCP ECN works together with RED by changing the exceed action from random drop to ECN marking. Instead of randomly dropping a packet when the average queue depth grows above the minimum threshold, RED marks packet with the special ECN flag. This marking uses the two least-significant bits of the TOS bytes in the IP header. The Diff-Serv QoS model uses upper six bits of the TOS byte for DSCP/IP Precedence values. The lower two bits are known as ECN Capable Transport (ECT) and Congestion Experienced (CE). If a endpoint (sender or receiver) is ECN capable, it will set the ECT bit in packet headers to signal that it is capable to responding to congestion notifications. If RED is ECN enabled, and it's going to drop a packet randomly, the following actions apply.

First, ECN RED checks to see that either the ECT or CE bits are set. If both of the bits are zero, the sending or receiving endpoint is not ECN capable, and the packet is randomly dropped. If either of the bits are set (ECT or CE) the RED procedure does not drop the packet, but instead sets the other bit and transmits the packet with both bits set. If both ECT and CE bits are set, the packet is simply transmitted. If the queue is full, RED drops the packets irrespective of their markings.

Copyright © 2010 Internetwork Expert

To verify this configuration, enable the TCP ECN feature on both R1 and R5. Enable the HTTP server service on R1, and configure R5 to transfer a file from R1. Shutdown the Frame Relay interface of R5 to ensure that packets take path across the serial link between R4 and R5.

```
R1:
ip http server
ip http path flash:
!
ip tcp ecn

R5:
ip tcp ecn
!
interface Serial 0/0/0
 shutdown

Rack1R5#copy http://admin:cisco@155.1.146.1/c2600-adventerprisek9-
mz.124-10.bin null:
```

Copyright © 2010 Internetwork Expert

Check the statistics for the service policy. Note the long detailed output for the HTTP class. It demonstrates the number of random drops and ECN marks occurred in the queue for IP traffic marked with IP precedence of 2.

```
Rack1R4#show policy-map interface serial 0/1
 Serial0/1

  Service-policy output: SERIAL_LINK

<snip>

    Class-map: HTTP (match-all)
      13530 packets, 7740858 bytes
      5 minute offered rate 98000 bps, drop rate 0 bps
      Match: access-group name HTTP
      QoS Set
        precedence 2
          Packets marked 13530
      Queueing
        Output Queue: Conversation 41
        Bandwidth remaining 33 (%)
        (pkts matched/bytes matched) 11508/6580550
        (depth/total drops/no-buffer drops) 3/79/0
         exponential weight: 9
         explicit congestion notification
         mean queue depth: 4

  class    Transmitted       Random drop       Tail drop     Minimum Maximum  Mark
           pkts/bytes        pkts/bytes        pkts/bytes     thresh  thresh  prob
      0       0/0               0/0               0/0           20      40  1/10
      1       0/0               0/0               0/0           22      40  1/10
      2   10031/5714738        79/44285           0/0            4      16  1/4
      3       0/0               0/0               0/0           26      40  1/10
      4       0/0               0/0               0/0           28      40  1/10
      5       0/0               0/0               0/0           30      40  1/10
      6       0/0               0/0               0/0           32      40  1/10
      7       0/0               0/0               0/0           34      40  1/10
   rsvp       0/0               0/0               0/0           36      40  1/10


  class     ECN Mark
            pkts/bytes
      0        0/0
      1        0/0
      2      673/390340
      3        0/0
      4        0/0
      5        0/0
      6        0/0
      7        0/0
   rsvp        0/0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.40 MQC Class-Based Generic Traffic Shaping

- Configure MQC shaping on R6 to limit the sending rate on its link to VLAN 146 to 384Kbps.
- The link to VLAN 67 should be limited to 512Kbps.
- Use a burst interval of 20ms.

Configuration

```
R6:
policy-map SHAPE_VLAN146
 class class-default
   shape average 384000 7680
!
policy-map SHAPE_VLAN67
 class class-default
   shape average 512000 10240
!
interface FastEthernet 0/0.146
  service-policy output SHAPE_VLAN146
!
interface FastEthernet 0/0.67
  service-policy output SHAPE_VLAN67
```

Verification

Note

MQC based traffic-shaping allows Generic Traffic Shaping to be applied to any traffic class. This makes traffic-shaping's configuration modular and unified, without special variants for encapsulations like Frame Relay. Class-based GTS is similar to legacy GTS, but allows shaping based on class-maps, as opposed to access-list classification with legacy GTS, thus allowing any MQC classification options, such as NBAR, to define shaped traffic. Additionally MQC based shaping allows the shaping queue to be tuned, while legacy GTS supports just simple WFQ with no option to tune it.

All shaping parameters and the metering model remains the same as with the legacy GTS approach. However, class-based shaping unifies the previously separated GTS and FRTS syntax, as we'll see in further tasks. Note one subtle difference – FRTS by default assumes B_e=0, while GTS by default assumes B_e=B_c.

Like legacy GTS, MQC based GTS limits the average rate based on the full packet size, including layer 2 overhead. Though this is not important for many applications, it may become of concern on low-speed interfaces with small layer 3 packet payloads.

Copyright © 2010 Internetwork Expert

With the command **shape average** under the policy-map class configuration, the CIR, B_c, B_e, and buffer limit are defined. The buffer limit is the maximum size of the default WFQ queue. Since this task did not mention anything about B_e, the default value is used. B_c can be calculated the same way as with legacy GTS, where B_c=CIR*T_c/1000. In our case, T_c is 20ms, therefore the B_c values are 7680 and 10240 bits.

In this scenario we used class-based shaping to limit the sub-interfaces sending rate. This is a common use of GTS, and the effect is that each sub-interface now uses its own software queue, where as by default, all sub-interfaces share the software queue of their main interface. This also allows the use of separate QoS policies per sub-interface, due to the ability to tune shaper's queue (see the next task for more information on this).

To verify this configuration generate three traffic flows across R6 to see the class-based GTS in action. Below this is accomplished with R6 sending IP SLA jitter probes to R4, R6 acting as an HTTP server for R4, and an ICMP flow being sent from SW1 to R4.

```
R4:
ip sla responder

R6:
ip http server
ip http path flash:
!
! Configure SLA probe for jitter with 60 byte packets
!
ip sla monitor 1
 type jitter dest-ipaddr 155.1.146.4 dest-port 16384 codec g729a
 timeout 1000
 frequency 1
!
ip sla monitor schedule 1 life forever start-time now

Rack1R4#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-
mz.124-24.T.bin null:

Rack1SW1#ping 155.1.146.4 repeat 100000

Type escape sequence to abort.
Sending 100000, 150-byte ICMP Echos to 155.1.146.4, timeout is 2
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Copyright © 2010 Internetwork Expert

Use the following show command to verify your configuration and check the basic traffic-shaping statistics. Note the time interval and B_c/B_e values. B_c is "Sustain bits/int" and B_e is "Excess bits/int". The "Byte Limit" equals to B_c+B_e in bytes – the maximum amount of data that the shaper can send during the interval.

```
Rack1R6#show policy-map interface fastEthernet 0/0.146

FastEthernet0/0.146

  Service-policy output: SHAPE_VLAN146

    Class-map: class-default (match-any)
      65844 packets, 28003483 bytes
      5 minute offered rate 334000 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           384000/384000    1920   7680      7680      20        960

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      6         65838     27998907  56930     27085058  yes
```

Next view the shaping queue contents. Note that the queue appears to be WFQ, while in reality it is CBWFQ (you can tune it). The IOS computed the WFQ parameters based on the configured shaping rate (384Kbps), and ended up with 32 flow queues and a CDT value of 64. IOS takes the queue depth as the buffer limit configured in **shape** command.

One substantial difference from per-interface CBWFQ is that shaper queue has no notion of Link Queues – control plane traffic uses the same dynamic flows and WFQ computes weights based on its IP precedence values.

Note that all three traffic flows have equal weights in the CBWFQ, but different average packet sizes.

```
Rack1R6#show traffic-shape queue
<snip>

Traffic queued in shaping queue on FastEthernet0/0.146
 Traffic shape class: class-default
  Queueing strategy: weighted fair
  Queueing Stats: 5/1000/64/0 (size/max total/threshold/drops)
     Conversations  3/4/32 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 384 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 1/32384/0/0/0
  Conversation 18, linktype: ip, length: 78
  source: 155.1.146.6, destination: 155.1.146.4, id: 0x0151, ttl: 255,
  TOS: 0 prot: 17, source port 50359, destination port 16384

  (depth/weight/total drops/no-buffer drops/interleaves) 1/32384/0/0/0
  Conversation 25, linktype: ip, length: 118
  source: 155.1.67.7, destination: 155.1.146.4, id: 0x8C4B, ttl: 254, prot: 1

  (depth/weight/total drops/no-buffer drops/interleaves) 3/32384/0/0/0
  Conversation 2, linktype: ip, length: 1518
  source: 155.1.146.6, destination: 155.1.146.4, id: 0x8F7E, ttl: 255,
  TOS: 0 prot: 6, source port 80, destination port 55310
```

Based on this we can see that MQC based GTS performs the same as legacy GTS, but has the powerful classification options available via the MQC.

Copyright © 2010 Internetwork Expert

10.41 MQC Class-Based GTS and CBWFQ

- Change the queue for the shaper applied to R6's connection to VLAN 146 as follows.
    - Provide 32Kbps of priority treatment for small packets with a layer three size of 60 bytes (voice packets) using a 4000 byte burst value
    - Guarantee the HTTP traffic 256Kbps of the shaper's bandwidth
    - Unclassified traffic should receive fair-queue treatment

Configuration

```
R6:
!
! Define the class for voice traffic
!
class-map VOICE
 match packet length min 60 max 60
!
! Access-list to match HTTP traffic
!
ip access-list extended HTTP
 permit tcp any eq 80 any
!
! Class-map to match HTTP traffic
!
class-map HTTP
 match access-group name HTTP
!
! This policy map is the child policy
! to the shaping policy-map
!
policy-map CBWFQ
 class VOICE
   priority 32 4000
 class HTTP
   bandwidth 256
 class class-default
   fair-queue
!
! Nest the above-configured service-policy
!
policy-map SHAPE_VLAN146
  class class-default
    shape average 384000 7680
    service-policy CBWFQ
```

Copyright © 2010 Internetwork Expert

Verification

Note

The ability to configure the shaper's queue is one of the most powerful features of class-based shaping. Previously, GTS was able to use only WFQ, and the only parameter you could change was the WFQ buffer limit.

Using MQC syntax you can "nest" another service policy inside a class configured for shaping. This nested service policy defines "sub-classes" and their respective parameters, including CBWFQ settings. The most prominent feature is the ability to configure an LLQ conversation, allowing optimal voice traffic handling when limiting traffic rate on sub-interfaces.

When you nest a CBWFQ configuration inside a shaper, the amount of bandwidth you can allocate to nested classes equals to the shaper average rate (384Kbps in this scenario). The default 75% rule does not work for available bandwidth here. In addition, the link queues are no longer in play, since the queue does not apply to a physical interface. Therefore, you may want to define a separate class for control plane traffic and guarantee it some bandwidth.

As mentioned in the previous task, the common use of shaping is limiting sub-interfaces' sending rates. By nesting a CBWFQ policy, you can define flexible, per sub-interface queuing policies, comparable to the features you have for VC-based technologies such as ATM and Frame-Relay.

Note the priority queue bandwidth of 32Kbps. We get this bandwidth from the assumption of VoIP packet sizes of 60 bytes (G.729), plus 18 bytes overhead for an Ethernet header with 4 bytes of VLAN tag at 50 packets per second = (60+18)*50*8=31200bps. Remember that priority bandwidth must take layer 2 overhead in account.

To verify the configuration, configure three traffic sources. R6 should generate IP SLA jitter probes and R4 should respond. Set this probe to time out in 5 seconds, and use the ToS byte 224 (IP Precedence 7) for signaling. This is to provide that the IP SLA control connection has a weight high enough not to starve in shaper queue. In addition to that, R6 should act as a web-server and R4 should transfer a large file from it. Lastly, source an ICMP packet stream from SW1 to R4.

Copyright © 2010 Internetwork Expert

```
R4:
ip sla responder

R6:
ip http server
ip http path flash:
!
! Configure SLA probe for jitter with 60 byte packets
! Note the ToS byte corresponding to IP Pecedence of 7
!
ip sla monitor 1
 type jitter dest-ipaddr 155.1.146.4 dest-port 16384 codec g729a
 frequency 1
 timeout 1000
 tos 224
!
ip sla monitor schedule 1 life forever start-time now
```

```
Rack1R4#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-
mz.124-24.T.bin null:

Rack1SW1#ping 155.1.146.4 repeat 100000

Type escape sequence to abort.
Sending 100000, 150-byte ICMP Echos to 155.1.146.4, timeout is 2
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Check the statistics for policy map matches. Note that the voice packet rate is far below its average of 32Kbps. This is because the system classifies the IP SLA control connection using dynamic queues, based on IP Precedence of seven. However, even this is not enough to guarantee timely communication, as we will see later.

```
Rack1R6#show policy-map interface fastEthernet 0/0.146
 FastEthernet0/0.146

  Service-policy output: SHAPE_VLAN146

    Class-map: class-default (match-any)
      269715 packets, 88958085 bytes
      30 second offered rate 384000 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           384000/384000    1920   7680      7680      20        960

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      7         269209    88909805  141137    75436903  yes

      Service-policy : CBWFQ

        Class-map: VOICE (match-all)
          7018 packets, 547404 bytes
          30 second offered rate 3000 bps, drop rate 0 bps
          Match: packet length min 60 max 60
          Queueing
            Strict Priority
            Output Queue: Conversation 40
            Bandwidth 32 (kbps) Burst 4000 (Bytes)
            (pkts matched/bytes matched) 4001/312078
            (total drops/bytes drops) 486/37908

        Class-map: HTTP (match-all)
          13966 packets, 20215902 bytes
          30 second offered rate 380000 bps, drop rate 0 bps
          Match: access-group name HTTP
          Queueing
            Output Queue: Conversation 41
            Bandwidth 256 (kbps)Max Threshold 64 (packets)
            (pkts matched/bytes matched) 13916/20175173
        (depth/total drops/no-buffer drops) 3/0/0

        Class-map: class-default (match-any)
          2727 packets, 327338 bytes
          30 second offered rate 1000 bps, drop rate 0 bps
          Match: any
          Queueing
            Flow Based Fair Queueing
            Maximum Number of Hashed Queues 32
        (total queued/total drops/no-buffer drops) 4/0/0
```

Copyright © 2010 Internetwork Expert

Look at the traffic-shaper queue contents. Note the HTTP flows (source port 80) and its weight of 96. The next flow with weight 4048 corresponds to the IP SLA control channel. It has a low chance of competing with the greedy HTTP flow, and thus the scheduler constantly delays it. The next flow corresponds to RIP routing updates. Note that CBWFQ treats them as dynamic flows, not as the link conversation. The scheduler calculates the weight for this flow based strictly on the IP Precedence of 6 – 4626. The last flow corresponds to the stream of ICMP packets from SW1 to R4. This flow hardly has any chances to get there.

Note that ICMP packets have the size 118 in the queue. This is because their original layer 3 size was 100 bytes, and 18 bytes was added as Ethernet header overhead – 14 bytes for frame-header and 4 bytes for VLAN tag.

```
Rack1R6#show traffic-shape queue
<snip>

Traffic queued in shaping queue on FastEthernet0/0.146
 Traffic shape class: class-default
  Queueing strategy: weighted fair
  Queueing Stats: 8/1000/64/308 (size/max total/threshold/drops)
     Conversations  4/6/32 (active/max active/max total)
     Reserved Conversations 1/1 (allocated/max allocated)
     Available Bandwidth 102 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 3/96/0/0/0
  Conversation 41, linktype: ip, length: 1266
  source: 155.1.146.6, destination: 155.1.146.4, id: 0x915E, ttl: 255,
  TOS: 0 prot: 6, source port 80, destination port 48053

  (depth/weight/total drops/no-buffer drops/interleaves) 3/4048/0/0/0
  Conversation 29, linktype: ip, length: 98
  source: 155.1.146.6, destination: 155.1.146.4, id: 0x0000, ttl: 255,
  TOS: 224 prot: 17, source port 57870, destination port 1967

  (depth/weight/total drops/no-buffer drops/interleaves) 1/4626/0/0/0
  Conversation 2, linktype: ip, length: 230
  source: 155.1.146.6, destination: 224.0.0.9, id: 0x0000, ttl: 2,
  TOS: 192 prot: 17, source port 520, destination port 520

  (depth/weight/total drops/no-buffer drops/interleaves) 1/32384/0/0/0
  Conversation 25, linktype: ip, length: 118
  source: 155.1.67.7, destination: 155.1.146.4, id: 0x8741, ttl: 254, prot: 1
```

Copyright © 2010 Internetwork Expert

Now check the IP SLA probe statistics. Note the high number of probe failures. To improve this, we are going to give a special flow to the IP SLA control signaling. From the queue contents dumped above, you may note that it uses UDP destination port of 1967.

```
Rack1R1#show ip sla monitor statistics 1
Round trip time (RTT)   Index 1
        Latest RTT: 18 ms
Latest operation start time: 07:41:16.514 UTC Fri Aug 15 2008
Latest operation return code: OK
RTT Values
        Number Of RTT: 873
        RTT Min/Avg/Max: 2/18/94 ms
Latency one-way time milliseconds
        Number of one-way Samples: 0
        Source to Destination one way Min/Avg/Max: 0/0/0 ms
        Destination to Source one way Min/Avg/Max: 0/0/0 ms
Jitter time milliseconds
        Number of Jitter Samples: 745
        Source to Destination Jitter Min/Avg/Max: 1/14/83 ms
        Destination to Source Jitter Min/Avg/Max: 1/1/2 ms
Packet Loss Values
        Loss Source to Destination: 127         Loss Destination to Source: 0
        Out Of Sequence: 0      Tail Drop: 0    Packet Late Arrival: 0
Voice Score Values
        Calculated Planning Impairment Factor (ICPIF): 38
MOS score: 2.85
Number of successes: 81
Number of failures: 31
Operation time to live: Forever
```

Configure a special class to take care of the IP SLA signaling. Assign this class a bandwidth value of 24Kbps in the CBWFQ policy map.

R1:

```
ip access-list extended SLA_SIGNALING
 permit udp any any eq 1967
!
class-map SLA_SIGNALING
 match access-gorup name SLA_SIGNALING
!
policy-map CBWFQ
  class SLA_SIGNALING
   bandwidth 24
```

Copyright © 2010 Internetwork Expert

Look at the policy-map statistics now. Note the offered rate for the Voice packets (the load-interval has been set to 30 seconds for this output), and that HTTP traffic claimed almost all the bandwidth available after the VOICE traffic.

```
Rack1R1#show policy-map interface fastEthernet 0/0
 FastEthernet0/0

  Service-policy output: SHAPE_VLAN146

    Class-map: class-default (match-any)
      10148 packets, 6778660 bytes
      30 second offered rate 382000 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           384000/384000    1920   7680      7680      20        960

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      7         10151     6779014   10144     6776812   yes

      Service-policy : CBWFQ

        Class-map: VOICE (match-all)
          5716 packets, 445848 bytes
          30 second offered rate 30000 bps, drop rate 0 bps
          Match: packet length min 60 max 60
          Queueing
            Strict Priority
            Output Queue: Conversation 40
            Bandwidth 32 (kbps) Burst 4000 (Bytes)
            (pkts matched/bytes matched) 5712/445536
            (total drops/bytes drops) 0/0

        Class-map: HTTP (match-all)
          4350 packets, 6321484 bytes
          30 second offered rate 352000 bps, drop rate 0 bps
          Match: access-group name HTTP
          Queueing
            Output Queue: Conversation 41
            Bandwidth 256 (kbps)Max Threshold 64 (packets)
            (pkts matched/bytes matched) 4348/6319712
       (depth/total drops/no-buffer drops) 3/0/0

        Class-map: SLA_CONTROL (match-all)
          7 packets, 686 bytes
          30 second offered rate 0 bps, drop rate 0 bps
          Match: access-group name SLA_CONTROL
          Queueing
            Output Queue: Conversation 42
            Bandwidth 24 (kbps)Max Threshold 64 (packets)
            (pkts matched/bytes matched) 7/686
       (depth/total drops/no-buffer drops) 0/0/0
```

Copyright © 2010 Internetwork Expert

```
        Class-map: class-default (match-any)
          75 packets, 10642 bytes
          30 second offered rate 0 bps, drop rate 0 bps
          Match: any
          Queueing
            Flow Based Fair Queueing
            Maximum Number of Hashed Queues 32
         (total queued/total drops/no-buffer drops) 2/0/0
```

Now check the IP SLA statistics. Note that the number of failures is zero now, and the average RTT is within the range of shaper's Tc interval (20ms).

```
Rack1R1#show ip sla monitor statistics 1
Round trip time (RTT)   Index 1
        Latest RTT: 22 ms
Latest operation start time: 00:47:16.013 UTC Tue Aug 19 2008
Latest operation return code: OK
RTT Values
        Number Of RTT: 1000
        RTT Min/Avg/Max: 7/22/41 ms
Latency one-way time milliseconds
        Number of one-way Samples: 0
        Source to Destination one way Min/Avg/Max: 0/0/0 ms
        Destination to Source one way Min/Avg/Max: 0/0/0 ms
Jitter time milliseconds
        Number of Jitter Samples: 999
        Source to Destination Jitter Min/Avg/Max: 1/11/30 ms
        Destination to Source Jitter Min/Avg/Max: 1/1/2 ms
Packet Loss Values
        Loss Source to Destination: 0           Loss Destination to Source: 0
        Out Of Sequence: 0      Tail Drop: 0    Packet Late Arrival: 0
Voice Score Values
        Calculated Planning Impairment Factor (ICPIF): 11
MOS score: 4.06
Number of successes: 9
Number of failures: 0
Operation time to live: Forever
```

Copyright © 2010 Internetwork Expert

> **Note**
>
> Load the *QoS* initial configurations prior to starting.

10.42 MQC Single-Rate Three-Color Policer

- Configure R4 to meter incoming HTTP traffic on its link to VLAN 146 as follows.
  - If the traffic rate is less than 128Kbps, mark the packets with an IP precedence of one
  - If the traffic exceeds 128Kbps, mark the packets with an IP precedence of zero
  - Drop violating traffic
- Ensure the burst size is large enough to accommodate normal and excess burst durations of 200ms and 300ms at a rate of 128Kbps.

Configuration

```
R4:
ip access-list extended HTTP
 permit tcp any eq 80 any
!
class-map HTTP
 match access-group name HTTP
!
policy-map POLICE_VLAN146
 class HTTP
   police 128000 3200 4800
    conform-action set-prec-transmit 1
    exceed-action set-prec-transmit 0
    violate-action drop
!
interface FastEthernet 0/1
  service-policy input POLICE_VLAN146
```

Copyright © 2010 Internetwork Expert

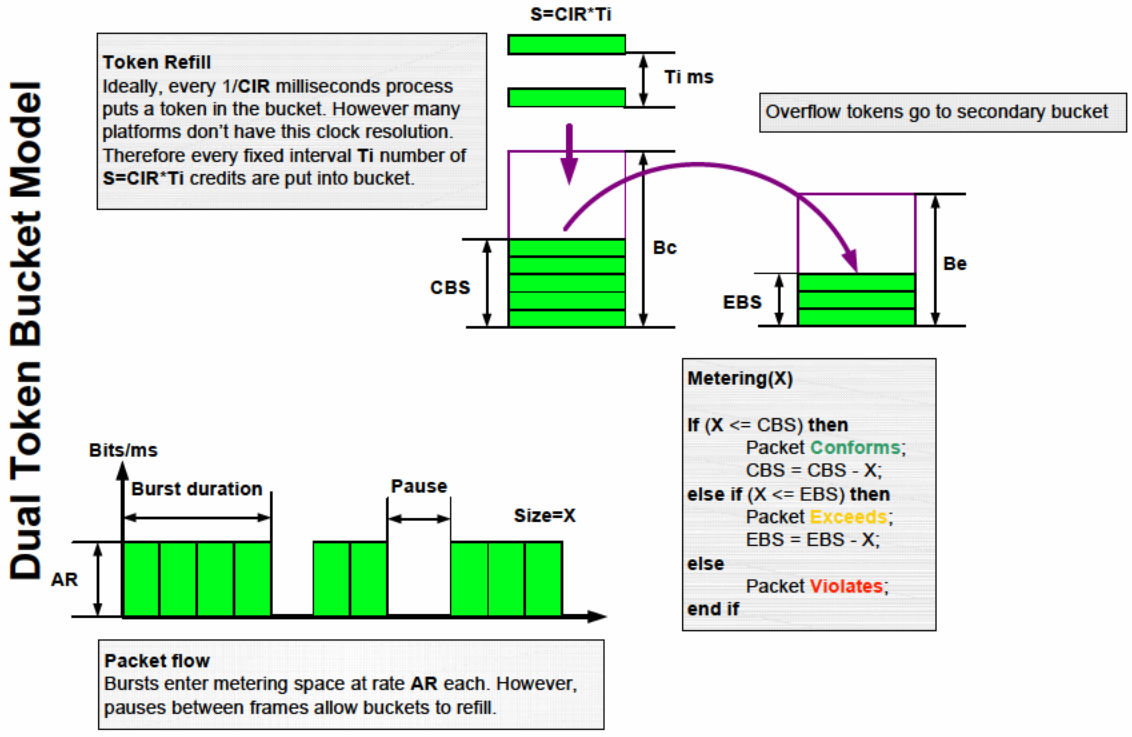
Verification

Note

A Single Rate Three Color Marker (Policer) or "srTCM" is the RFC-based implementation of the metering process. Compared to legacy CAR the following has changed.

First, the concept of Excess Burst (B_e) is no longer tied to a pseudorandom marking behavior. Rather, B_e is used as extra credit for periods of inactivity. Secondly, the policing process uses a Dual Token Bucket model, but the implementation is different. "Three color" in this case means that incoming traffic is metered against CIR using the configured burst sizes, and the result is a marking using either of three colors: Green (Conform), Yellow (Exceed), Red (Violate).

As usual, the token bucket exhibits the concept of a sliding metering window over the packet flow. However, this time process uses a secondary, Excess Burst bucket.



The Excess Burst bucket accumulates extra credit, spilled over from the Normal Bucket when it overfills. This only happens during long periods of pause that exceeds the averaging interval T_c. With just a single bucket, the process does not account for "extended" periods of silence. The Excess Burst bucket allows accumulating credits and using them when incoming burst exceeds the configured B_c value. Thus, B_e in this usage serves the similar purpose that the B_e functionality has with traffic-shaping. In fact, these two are complementary in the sense that srTCM's B_e properly "recognizes" the excessive bursting of traffic shaping.

In a classic token bucket model, a special timer fires every T_i interval, triggering the token refresh procedure. This method, however, limits the "resolution" of metering. At the same time, the amount of CPU operations needed to meter a packet is small. This is because the procedure uses only addition and subtraction operations, which consume little CPU cycles.

Cisco's implementation of the token bucket used for srTCM uses a different algorithm. This algorithm has "unlimited" precision, but consumes more CPU cycles per-packet. The process keeps track of three variables:

CBS – current normal burst size
EBS – current excess burst size
T0 – last packet arrival time.

Now imagine that a new packet of size "S" arrives at time "T".

Step 1:
The process computes accumulated credit:

Cr = CIR*(T-T0)

then adds this values to CBS:

```
if ((CBS + Cr) <= Bc) then
     CBS = CBS + Cr;
else
     CBS = Bc;
end if
```

Step 2:
The process computes a new EBS size if there are enough credits:

```
If (CBS + Cr > Bc) then
      EBS = CBS + Cr – Bc;
end

if (EBS > Be) then
      EBS = Be;
```

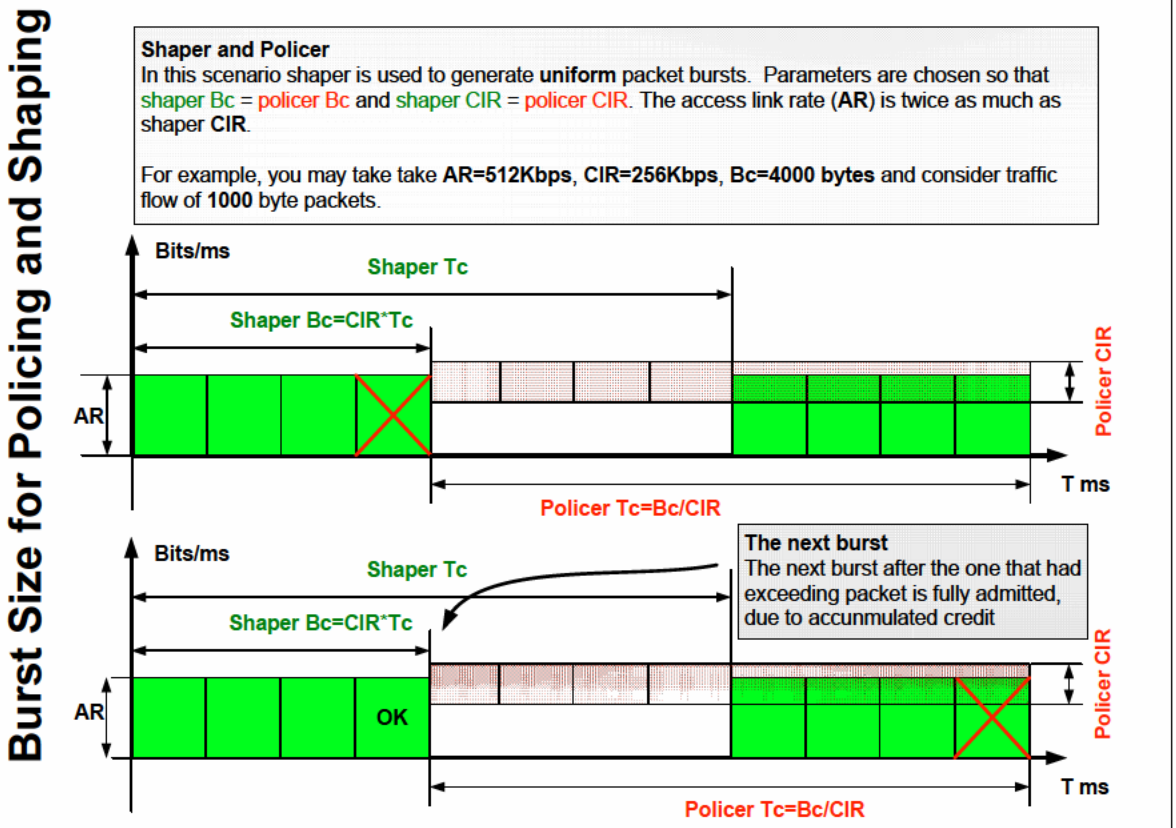
Step 3:
The process compares packet size to accumulated credits and performs marking:

```
if ( S < CBS ) then
    Packet Conforms;
    CBS = CBS – S;
else if (S < EBS) then
    Packet Exceeds;
    EBS = EBS – S;
else
    Packet Violates;
end if
```

This implementation does not use any hardware timer, and thus the "precision" is not limited. However, additional multiplication operations may become CPU consuming on intense packet flows.

For an optimal burst size selection, if you are using srTCM to perform traffic admission at the network edge, ensure your policing burst sizes match at least one packet size more than the burst size configured at the customer side. This is needed because now the router meters traffic with extra precision, and the effect illustrated on the next figure (shaper Bc = policer Bc) may occur.

Copyright © 2010 Internetwork Expert



Note that the policer may mark the beginning of the second burst as exceeding. This is because the policer B_c exactly equals to shaper B_c. In this situation when the sliding window hits the beginning of the next burst it has no capacity to accommodate another packet. The packet marked as exceeding does not count for the sliding window content, and thus the next burst will have all four packets admitted. For this ideal simulation, the policer B_c should be larger than traffic burst size by at least one packet. This brings us to the formula:

Policer B_c = Shaper Burst Size + Average Packet Size

In addition to that, it makes sense to configure burst size proportional to the average packet size as well, to ensure more precise matching. However, this is not necessary, especially with larger bursts.

Note that the above effect was insignificant with CAR, due to the "lower" resolution feature. Next, if you are using policing to limit the traffic rate, pick up the burst sizes to reduce the negative impact on TCP traffic. This has been discussed in the section dedicated to rate limiting with CAR.

Copyright © 2010 Internetwork Expert

In this particular example the burst sizes are based on the time intervals as follows:

Bc = 128000*0,2/8=3200 bytes
Be = 128000*0,3/8=4800 bytes

To test the policer simulate an HTTP traffic flow from R1 to R4. At the same time, pre-shape the traffic rate from R1 to R4 using GTS. Set the shaper's burst size to 12800 bits (1600 bytes), which is half of the policer's burst.

```
R1:
ip http server
ip http path flash:
!
interface FastEthernet 0/0
  traffic-shape rate 128000 12800 0

R4:
interface FastEthernet 0/1
  load-interval 30

Rack1R4#copy http://admin:cisco@155.1.146.1/c2600-adventerprisek9-
mz.124-10.bin null:
```

Check the policy-map statistics.

```
Rack1R4#show policy-map interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy input: POLICE_VLAN146

    Class-map: HTTP (match-all)
      1245 packets, 1786990 bytes
      30 second offered rate 127000 bps, drop rate 0 bps
      Match: access-group name HTTP
      police:
          cir 128000 bps, bc 3200 bytes, be 4800 bytes
        conformed 1241 packets, 1780934 bytes; actions:
          set-prec-transmit 1
        exceeded 4 packets, 6056 bytes; actions:
          set-prec-transmit 0
        violated 0 packets, 0 bytes; actions:
          drop
        conformed 127000 bps, exceed 0 bps, violate 0 bps

    Class-map: class-default (match-any)
      124 packets, 11184 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

Note the small number of exceeding packets. Now change the shaper burst size to be equal to the burst size of the policer (25600 bits) and repeat the metering on the same file transfer.

```
R1:
interface FastEthernet 0/0
  traffic-shape rate 128000 25600 0

Rack1R4#show policy-map interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy input: POLICE_VLAN146

    Class-map: HTTP (match-all)
      2510 packets, 2949429 bytes
      30 second offered rate 127000 bps, drop rate 0 bps
      Match: access-group name HTTP
      police:
          cir 128000 bps, bc 3200 bytes, be 4800 bytes
        conformed 2254 packets, 2575997 bytes; actions:
          set-prec-transmit 1
        exceeded 256 packets, 373432 bytes; actions:
          set-prec-transmit 0
        violated 0 packets, 0 bytes; actions:
          drop
        conformed 111000 bps, exceed 16000 bps, violate 0 bps

    Class-map: class-default (match-any)
      108 packets, 16776 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

The amount of packets incorrectly marked as exceeding is around 10%, even though shaper burst equals the policer burst. This is the demonstration of the effect described above.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.43 MQC Hierarchical Policers

- Configure R4 to limit the aggregate rate of HTTP traffic entering the connection to VLAN 146 to 128Kbps.
- Transmit conforming packets, mark exceeding packets with an IP precedence of zero, and drop violating packets.
- For HTTP traffic flows from R1 and R6, limit the rate to 64Kbps for each flow.
- For these second-level policers, set the conform action to set the IP precedence to 1 and transmit, the exceed action to set the IP precedence to 0 and transmit, and the violate action to set the IP precedence to 0 and transmit.

Configuration

```
R4:
!
! All HTTP traffic
!
ip access-list extended HTTP
 permit tcp any eq 80 any
!
class-map HTTP
 match access-group name HTTP

!
! Traffic from R1 and R6 respectively
!
ip access-list extended FROM_R1
 permit ip host 155.1.146.1 any
!
ip access-list extended FROM_R6
 permit ip host 155.1.146.6 any
!
!
!
class-map FROM_R1
 match access-group name FROM_R1
!
class-map FROM_R6
 match access-group name FROM_R6

!
! Nested policers (subrate)
!
policy-map SUBRATE_POLICER
 class FROM_R1
  police 64000 3200 4800
   conform-action set-prec-transmit 1
   exceed-action set-prec-transmit 0
   violate-action set-prec-transmit 0
 class FROM_R6
  police 64000 3200 4800
```

Copyright © 2010 Internetwork Expert

```
   conform-action set-prec-transmit 1
   exceed-action set-prec-transmit 0
   violate-action set-prec-transmit 0

!
! Policer configuration using MQC syntax.
!
policy-map POLICE_VLAN146
 class HTTP
   police 128000 3200 4800
    conform-action transmit
    exceed-action set-prec-transmit 0
    violate-action drop
   service-policy SUBRATE_POLICER
!
interface FastEthernet 0/1
  service-policy input POLICE_VLAN146
```

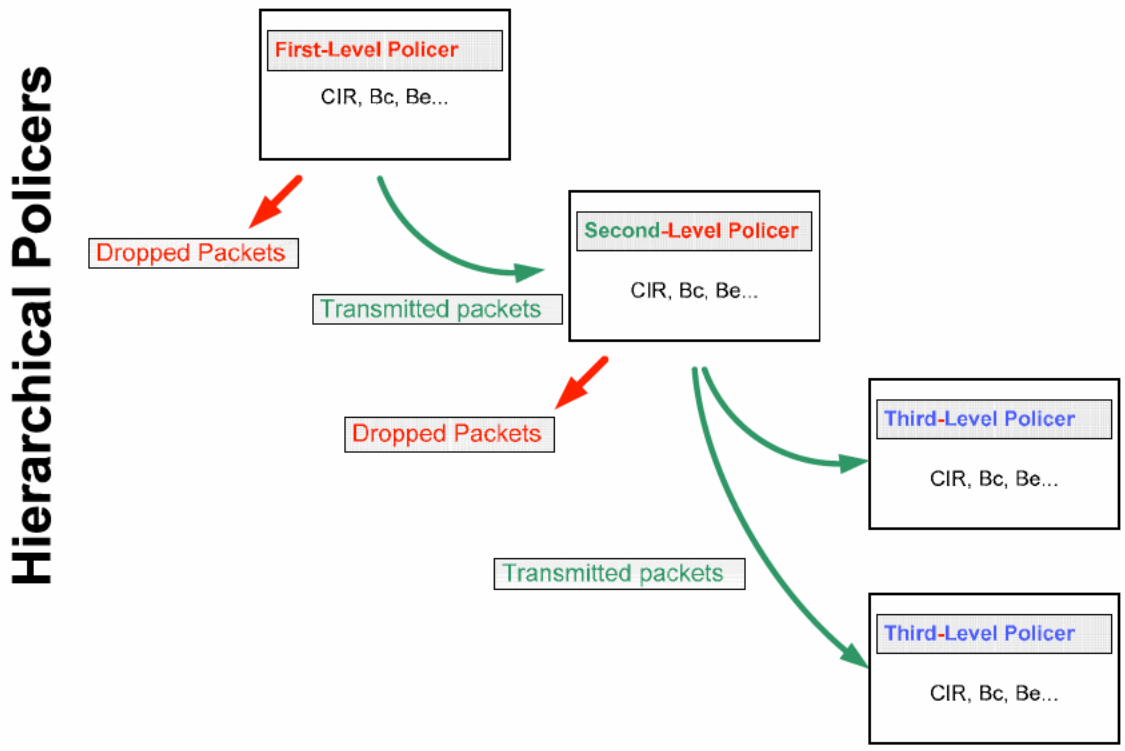
Verification

Note

MQC allows the nesting of up to three levels of policing. The syntax uses nested service-policies, where the parent policy defines the aggregate rate, and the child policies define the sub-rates for each class. The effect is similar to the cascaded rate-limit statements used in legacy CAR, however, note the following differences.

First, with MQC hierarchical policing, the upper level policers are applied first before the nested policers. With CAR, you usually configure the “aggregate” statement in the end of the other “sub-rate” statements. Secondly, every policer could be a single rate or two-rate policer. Furthermore, the MQC allows for a more natural and intuitive grouping of classes and policing statements.

Copyright © 2010 Internetwork Expert



Start verification by generating a single traffic flow from R1 down to SW2 across R4. To accomplish this, you need to shut down the Frame-Relay interface of R5. The general idea is that link between R4 and R5 is set to 128Kbps and the traffic flows will eventually oversubscribe it. The task configuration ensures that aggregate traffic across the serial link never exceeds 128Kbps, but allows every flow to send traffic above the guaranteed 64Kbps. Generate an HTTP traffic flow from R1 to SW2 and pre-shape it down to 128Kbps on R6. Ensure the traffic shaper burst smaller than the configured policer burst by at least one average packet size to avoid the mis-marking behavior as seen in the previous task.

Copyright © 2010 Internetwork Expert

```
R1:
ip http server
ip http path flash:
!
! Bc = 12800 bits = 1600 bytes
!
interface FastEthernet 0/0
  traffic-shape rate 128000 12800 0

R4:
interface FastEthernet 0/1
  load-interval 30

R5:
interface Serial 0/0/0
 shutdown
```

```
Rack1SW2#copy http://admin:cisco@155.1.146.1/c2600-adventerprisek9-
mz.124-10.bin null:
```

Note that traffic from R1 to SW2 exceeds the nested policer rate and thus the second-level policer marks almost half of the traffic with IP Precedence 0.

```
Rack1R4#show policy-map interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy input: POLICE_VLAN146

    Class-map: HTTP (match-all)
      2428 packets, 1432520 bytes
      30 second offered rate 126000 bps, drop rate 0 bps
      Match: access-group name HTTP
      police:
          cir 128000 bps, bc 3200 bytes, be 4800 bytes
        conformed 2428 packets, 1432520 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          set-prec-transmit 0
        violated 0 packets, 0 bytes; actions:
          drop
        conformed 123000 bps, exceed 0 bps, violate 0 bps

      Service-policy : SUBRATE_POLICER

        Class-map: FROM_R1 (match-all)
          2428 packets, 1432520 bytes
          30 second offered rate 126000 bps, drop rate 0 bps
          Match: access-group name FROM_R1
          police:
              cir 64000 bps, bc 3200 bytes, be 4800 bytes
            conformed 1231 packets, 726290 bytes; actions:
              set-prec-transmit 1
            exceeded 0 packets, 0 bytes; actions:
              set-prec-transmit 0
            violated 1197 packets, 706230 bytes; actions:
              set-prec-transmit 0
            conformed 63000 bps, exceed 0 bps, violate 62000 bps
```

Copyright © 2010 Internetwork Expert

```
        Class-map: FROM_R6 (match-all)
          0 packets, 0 bytes
          30 second offered rate 0 bps, drop rate 0 bps
          Match: access-group name FROM_R6
          police:
              cir 64000 bps, bc 3200 bytes, be 4800 bytes
            conformed 0 packets, 0 bytes; actions:
              set-prec-transmit 1
            exceeded 0 packets, 0 bytes; actions:
              set-prec-transmit 0
            violated 0 packets, 0 bytes; actions:
              set-prec-transmit 0
            conformed 0 bps, exceed 0 bps, violate 0 bps

        Class-map: class-default (match-any)
          0 packets, 0 bytes
          30 second offered rate 0 bps, drop rate 0 bps
          Match: any

    Class-map: class-default (match-any)
      10 packets, 860 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

Now add another traffic flow, from R6 down to SW4 and across the serial link between R4 and R5.

```
R6:
ip http server
ip http path flash:
!
interface FastEthernet 0/0.146
  traffic-shape rate 128000 12800 0
```

```
Rack1SW4#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-mz.124-
24.T.bin null:
```

Verify the policer statistics once again. Note that now both second-level policers meter input rate close to 64Kbps with no violating traffic.

```
Rack1R4#show policy-map interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy input: POLICE_VLAN146

    Class-map: HTTP (match-all)
      11514 packets, 6792149 bytes
      30 second offered rate 126000 bps, drop rate 0 bps
      Match: access-group name HTTP
      police:
          cir 128000 bps, bc 3200 bytes, be 4800 bytes
        conformed 11513 packets, 6791559 bytes; actions:
          transmit
        exceeded 2 packets, 1180 bytes; actions:
          set-prec-transmit 0
        violated 0 packets, 0 bytes; actions:
          drop
        conformed 126000 bps, exceed 0 bps, violate 0 bps

      Service-policy : SUBRATE_POLICER

        Class-map: FROM_R1 (match-all)
          10295 packets, 6074050 bytes
          30 second offered rate 63000 bps, drop rate 0 bps
          Match: access-group name FROM_R1
          police:
              cir 64000 bps, bc 3200 bytes, be 4800 bytes
            conformed 5817 packets, 3432030 bytes; actions:
              set-prec-transmit 1
            exceeded 0 packets, 0 bytes; actions:
              set-prec-transmit 0
            violated 4479 packets, 2642610 bytes; actions:
              set-prec-transmit 0
            conformed 63000 bps, exceed 0 bps, violate 0 bps

        Class-map: FROM_R6 (match-all)
          1219 packets, 718099 bytes
          30 second offered rate 63000 bps, drop rate 0 bps
          Match: access-group name FROM_R6
          police:
              cir 64000 bps, bc 3200 bytes, be 4800 bytes
            conformed 1216 packets, 716329 bytes; actions:
              set-prec-transmit 1
            exceeded 3 packets, 1770 bytes; actions:
              set-prec-transmit 0
            violated 0 packets, 0 bytes; actions:
              set-prec-transmit 0
            conformed 63000 bps, exceed 0 bps, violate 0 bps

        Class-map: class-default (match-any)
          0 packets, 0 bytes
          30 second offered rate 0 bps, drop rate 0 bps
          Match: any

    Class-map: class-default (match-any)
      68 packets, 8728 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

This is due to the fact the Serial interface runs WFQ, and shares bandwidth equally between two flows.

```
Rack1R4#show interfaces serial 0/1/0
Serial0/1/0 is up, line protocol is up
  Hardware is PowerQUICC Serial
  Internet address is 155.1.45.4/24
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
     reliability 255/255, txload 20/255, rxload 2/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  Last input 00:00:01, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: weighted fair
  Output queue: 12/1000/64/0 (size/max total/threshold/drops)
     Conversations  2/3/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 1158 kilobits/sec
  5 minute input rate 18000 bits/sec, 53 packets/sec
  5 minute output rate 124000 bits/sec, 27 packets/sec
     37685 packets input, 2352426 bytes, 0 no buffer
     Received 785 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     18965 packets output, 10363861 bytes, 0 underruns
     0 output errors, 0 collisions, 3 interface resets
     0 unknown protocol drops
     0 output buffer failures, 0 output buffers swapped out
     6 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.44 MQC Two-Rate Three-Color Policer

- Modify the first-level policy-map applied directly to R4's connection to VLAN 146 to remove the aggregate policer of 128Kbps.
- Modify the second-level policy-map to replace the single-rate policers with two-rate policers.
- Use the CIR value of 64Kbps and PIR value of 128Kbps.
- Use the values of CIR*400ms and PIR*200ms for normal and excess burst sizes.
- Set the conform action to set IP precedence 1 and transmit, the exceed action to set IP precedence of 0 and transmit, and the violate action to drop.

Configuration

```
R4:
policy-map SUBRATE_POLICER
 class FROM_R1
  police cir 64000 bc 3200 pir 128000 be 6400
   conform-action set-prec-transmit 1
   exceed-action set-prec-transmit 0
   violate-action drop
 class FROM_R6
  police cir 64000 bc 3200 pir 128000 be 6400
   conform-action set-prec-transmit 1
   exceed-action set-prec-transmit 0
   violate-action drop
!
policy-map POLICE_VLAN146
 class HTTP
   no police
   service-policy SUBRATE_POLICER
!
interface FastEthernet 0/1
  service-policy input POLICE_VLAN146
```

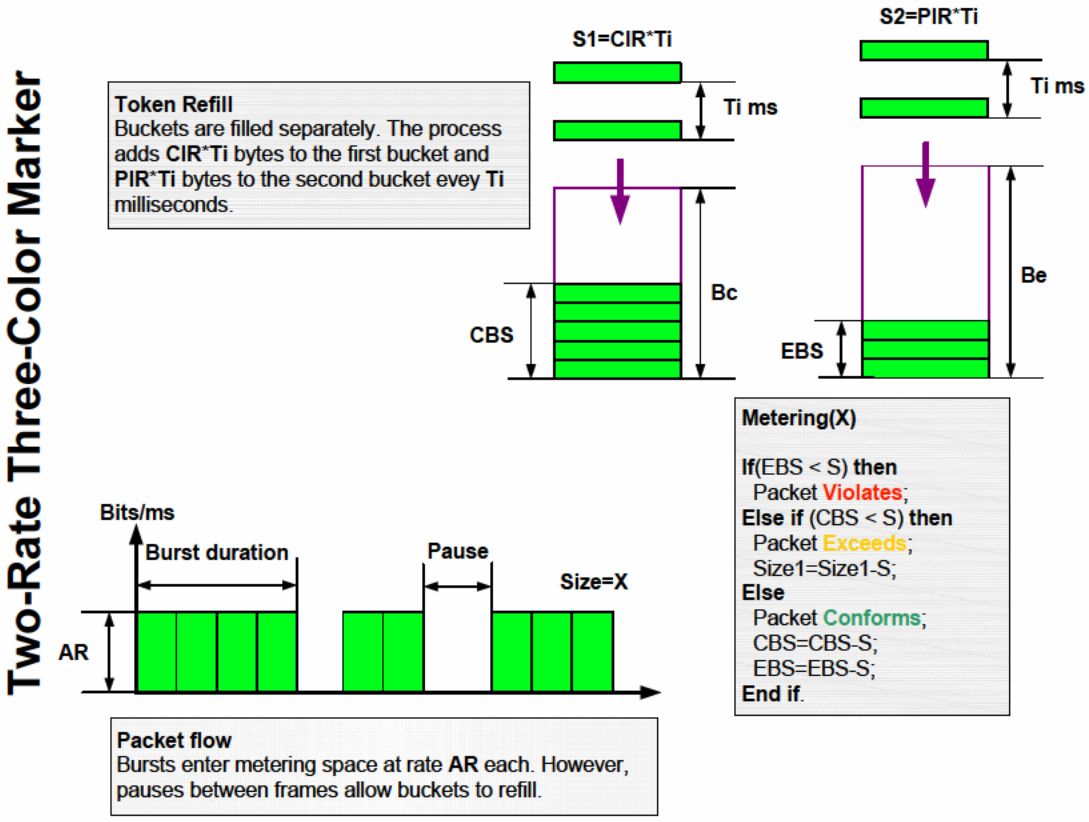
Copyright © 2010 Internetwork Expert

Verification

Note

The two-rate policer implements an RFC-based procedure to meter traffic rate against two pre-configured rates, average and peak. The RFC names this procedure as a two-rate three-color marker, or "trTCM". Three colors correspond to "Conform", "Exceed" and "Violate" actions of the policer.

Four configurable parameters apply to the two-rate policer: CIR, PIR, B_c, B_e. This time B_c bounds the average bursts, and B_e bounds the peak burst. Two-rate policer runs two token buckets at the same time. Each bucket has its own fill rate, as illustrated below.



The process refills each bucket separately, at its own rate. The first bucket fills at rate CIR, while the second bucket fills at rate PIR. As usual, a special timer fires every T_i interval and triggers the token refresh procedure. This, however, limits the "resolution" of metering procedure.

Copyright © 2010 Internetwork Expert

The alternate implementation, used by Cisco with trTCM, is similar to the implementation of srTCM. The algorithm uses the following variables:

CBS – current normal burst size (initially set to B_c)
EBS – current excess burst size (initially set to B_e)
T0 – last packet arrival time.

Now imagine that a new packet of size "S" arrives at time "T".

Step 1:
The process computes the accumulated credit and new burst sizes:

```
Cr1 = CIR*(T-T0);
Cr2 = PIR*(T-T0);
```

then adds these values to CBS and EBS:

```
if ((CBS + Cr1) <= Bc) then
     CBS = CBS + Cr1;
else
     CBS = Bc;
end if

if ((EBS + Cr2) <= Be) then
     EBS = EBS + Cr2;
else
     EBS = Be;
end if
```

Step 2:
The process compares the packet size to accumulated credits:

```
if ( S > EBS ) then
    Packet Violates;
else if (S > CBS) then
    Packet Exceeds;
    EBS = EBS – S;
else
    Packet Conforms;
    EBS = EBS – S;
    CBS = CBS – S;
end if.
```

Copyright © 2010 Internetwork Expert

Effectively the two-rate policer runs two sliding windows over the packet stream, metering against two bitrates using two separate averaging intervals. This procedure is common on Service Provider edges that offer oversubscription to customers, based on PIR and CIR rates.

To verify this generate a single traffic flow from R1 down to SW2 across R4. To accomplish this, you need to shut down the Frame-Relay interface of R5. The general idea is that link between R4 and R5 is set to 128Kbps and the traffic flows will eventually oversubscribe it. The task configuration ensures that an individual flow across the serial link never exceeds 128Kbps, but allows every flow to send traffic above guaranteed 64Kbps.

Generate an HTTP traffic flow from R1 to SW2 and pre-shape it down to 128Kbps on R6. Ensure the traffic shaper burst is smaller than the configured policer burst by at least one average packet size.

```
R1:
ip http server
ip http path flash:
!
! Bc = 12800 bits = 1600 bytes
!
interface FastEthernet 0/0
  traffic-shape rate 128000 12800 0

R4:
interface FastEthernet 0/1
  load-interval 30

R5:
interface Serial 0/0/0
 shutdown
```

```
Rack1SW2#copy http://admin:cisco@155.1.146.1/c2600-adventerprisek9-
mz.124-10.bin null:
```

Copyright © 2010 Internetwork Expert

Note that traffic from R1 to SW2 exceeds the policer average rate but not the peak rate. Therefore, the policer marks half of the packets as exceeding. Since the traffic is TCP based, it adapts to the maximum allowable rate.

```
Rack1R4#show policy-map interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy input: POLICE_VLAN146

    Class-map: HTTP (match-all)
      2331 packets, 1373671 bytes
      30 second offered rate 121000 bps, drop rate 0 bps
      Match: access-group name HTTP

      Service-policy : SUBRATE_POLICER

        Class-map: FROM_R1 (match-all)
          2331 packets, 1373671 bytes
          30 second offered rate 121000 bps, drop rate 0 bps
          Match: access-group name FROM_R1
          police:
              cir 64000 bps, bc 3200 bytes
              pir 128000 bps, be 6400 bytes
            conformed 1183 packets, 696351 bytes; actions:
              set-prec-transmit 1
            exceeded 1148 packets, 677320 bytes; actions:
              set-prec-transmit 0
            violated 0 packets, 0 bytes; actions:
              drop
            conformed 62000 bps, exceed 61000 bps, violate 0 bps

        Class-map: FROM_R6 (match-all)
          0 packets, 0 bytes
          30 second offered rate 0 bps, drop rate 0 bps
          Match: access-group name FROM_R6
          police:
              cir 64000 bps, bc 3200 bytes
              pir 128000 bps, be 6400 bytes
            conformed 0 packets, 0 bytes; actions:
              set-prec-transmit 1
            exceeded 0 packets, 0 bytes; actions:
              set-prec-transmit 0
            violated 0 packets, 0 bytes; actions:
              drop
            conformed 0 bps, exceed 0 bps, violate 0 bps

        Class-map: class-default (match-any)
          0 packets, 0 bytes
          30 second offered rate 0 bps, drop rate 0 bps
          Match: any

    Class-map: class-default (match-any)
      1081 packets, 318426 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

Now add another traffic flow from R6 down to SW4 and across the serial link between R4 and R5.

```
R6:
ip http server
ip http path flash:
!
interface FastEthernet 0/0.146
  traffic-shape rate 128000 12800 0

Rack1SW4#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-mz.124-
24.T.bin null:
```

Verify the policer statistics once again. Now both policers meter the same average rate, and there is no exceeding traffic. This is because the oversubscribed serial link between R4 and R5 uses fair queuing and divides the bandwidth equally between the two flows.

Copyright © 2010 Internetwork Expert

```
Rack1R4#show policy-map interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy input: POLICE_VLAN146

    Class-map: HTTP (match-all)
      15682 packets, 9247646 bytes
      30 second offered rate 127000 bps, drop rate 0 bps
      Match: access-group name HTTP

      Service-policy : SUBRATE_POLICER

        Class-map: FROM_R1 (match-all)
          14228 packets, 8392901 bytes
          30 second offered rate 63000 bps, drop rate 0 bps
          Match: access-group name FROM_R1
          police:
              cir 64000 bps, bc 3200 bytes
              pir 128000 bps, be 6400 bytes
            conformed 7917 packets, 4669411 bytes; actions:
              set-prec-transmit 1
            exceeded 6311 packets, 3723490 bytes; actions:
              set-prec-transmit 0
            violated 0 packets, 0 bytes; actions:
              drop
            conformed 63000 bps, exceed 0 bps, violate 0 bps

        Class-map: FROM_R6 (match-all)
          1454 packets, 854745 bytes
          30 second offered rate 63000 bps, drop rate 0 bps
          Match: access-group name FROM_R6
          police:
              cir 64000 bps, bc 3200 bytes
              pir 128000 bps, be 6400 bytes
            conformed 1451 packets, 852975 bytes; actions:
              set-prec-transmit 1
            exceeded 3 packets, 1770 bytes; actions:
              set-prec-transmit 0
            violated 0 packets, 0 bytes; actions:
              drop
            conformed 63000 bps, exceed 0 bps, violate 0 bps

        Class-map: class-default (match-any)
          0 packets, 0 bytes
          30 second offered rate 0 bps, drop rate 0 bps
          Match: any

    Class-map: class-default (match-any)
      1190 packets, 341980 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

Note that traffic never violated the peak rate condition, thanks to the fact that the link between R4 and R5 is 128Kbps.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.45 MQC Peak Shaping

- Configure R1 and R6 to shape HTTP traffic to a peak rate of 128Kbps as it is sent out to VLAN 146.
- Use B_c and B_e bursts based on a 10ms interval.

Configuration

```
R1:
access-list 180 permit tcp any eq 80 any
!
class-map HTTP
 match access-group 180
!
policy-map POLICY_VLAN146_OUT
  class HTTP
    shape peak 64000 6400 6400
!
interface FastEthernet 0/0
  service-policy output POLICY_VLAN146_OUT

R6:
access-list 180 permit tcp any eq 80 any
!
class-map HTTP
 match access-group 180
!
policy-map POLICY_VLAN146_OUT
  class HTTP
    shape peak 64000 6400 6400
!
interface FastEthernet 0/0.146
  no traffic-shape rate
  service-policy output POLICY_VLAN146_OUT
```

Verification

Note

Peak shaping may look confusing at first sight, however, its function becomes clear once you think of oversubscription. As we discussed before, oversubscription means selling customers more bandwidth than a network can supply, hoping that not all connections would use their maximum sending rate at the same time. With oversubscription, a traffic contract usually specifies three parameters: PIR, CIR and T_c – peak rate, committed rate, and averaging time interval for rate measurements. The SP allows customers to send traffic at rates up to PIR, but only guarantees CIR rate in case of network congestion.

Copyright © 2010 Internetwork Expert

Inside the network, the SP uses any of the max-min scheduling procedures to implement bandwidth sharing in such manner that oversubscribed traffic has lower preference than conforming traffic. Additionally, the SP generally assumes that customers respond to notifications of traffic congestion in the network (either explicit, such as FECN/BECN/TCP ECN or implicit such as packet drops in TCP) by slowing down sending rate.

Commonly customers implement traffic shaping to conform to their traffic contract, and the provider uses traffic policing to enforce the contract. If a contract specifies PIR, then it makes sense for customer to shape traffic at PIR rate. However, this makes difficult to deduce the CIR value just by looking at the router configuration. In some circumstances, like with Frame-Relay networks, a secondary parameter, known as minCIR, may help to understand the configuration quickly. In general, it would benefit to see CIR and PIR in the shaping configuration at the same time. This is exactly the idea behind shape peak. When you configure **shape peak <CIR> <Bc> <Be>** the actual maximum sending rate is limited to PIR = CIR*(1+B_e/B_c).

That is, each time interval T_c=B_c/CIR the shaper allows sending up to B_c+B_e bits of data. By default, if you omit the value for B_e, it equals to B_c and thus PIR=2*CIR by default. However, due to a cosmetic IOS bug in the show output, this is NOT reflected unless you explicitly specify the Be value in command line. With shape peak configured this way, you can see both CIR as the “average rate” and PIR as the “target rate” when issuing “show policy-map” command.

```
Rack1R6#show policy-map interface fastEthernet 0/0.146
 FastEthernet0/0.146
<snip>
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           128000/64000     1600   6400      6400      100       1600
...
```

All other shaping functions remain the same as with the classic GTS - shape peak is just more suited for use with oversubscription scenarios. Also, in Frame-Relay networks you may want to use configuration similar to the following to respond to congestion notifications:

```
shape peak <CIR> <Bc> <Be>
shape adaptive <CIR>
```

Copyright © 2010 Internetwork Expert

To verify the configuration, ensure that traffic from VLAN146 goes across R4 to reach R5. Then, start transferring a file from R1 to SW2.

```
R1 & R6:
ip http server
ip http path flash:

Rack1SW2#copy http://admin:cisco@155.1.146.1/c2600-adventerprisek9-
mz.124-10.bin null:
```

Check the policer statistics on R4 (remember we have two-rate policers admitting traffic flows from R1 and R6). Note that the policer considers half of the packets as exceeding, and that the offered rate is close to 128Kbps (peak).

```
Rack1R4#show policy-map interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy input: POLICE_VLAN146

    Class-map: HTTP (match-all)
      20451 packets, 12066090 bytes
      30 second offered rate 126000 bps, drop rate 0 bps
      Match: access-group name HTTP

      Service-policy : SUBRATE_POLICER

        Class-map: FROM_R1 (match-all)
          20451 packets, 12066090 bytes
          30 second offered rate 126000 bps, drop rate 0 bps
          Match: access-group name FROM_R1
          police:
              cir 64000 bps, bc 3200 bytes
              pir 128000 bps, be 6400 bytes
            conformed 11113 packets, 6556670 bytes; actions:
              set-prec-transmit 1
            exceeded 9338 packets, 5509420 bytes; actions:
              set-prec-transmit 0
            violated 0 packets, 0 bytes; actions:
              drop
            conformed 64000 bps, exceed 62000 bps, violate 0 bps

        Class-map: FROM_R6 (match-all)
          0 packets, 0 bytes
          30 second offered rate 0 bps, drop rate 0 bps
          Match: access-group name FROM_R6
          police:
              cir 64000 bps, bc 3200 bytes
              pir 128000 bps, be 6400 bytes
            conformed 0 packets, 0 bytes; actions:
              set-prec-transmit 1
            exceeded 0 packets, 0 bytes; actions:
              set-prec-transmit 0
            violated 0 packets, 0 bytes; actions:
              drop
            conformed 0 bps, exceed 0 bps, violate 0 bps
<snip>
```

Copyright © 2010 Internetwork Expert

Check the shaper statistics on R1. Note the average and target traffic rates. Also, note that shaping is no longer active, although some packets have been delayed. This is because the initial stage of TCP transmission is usually aggressive, and it takes time to adapt to the enforced rates and stabilize the sending rate. The sustained sending rate is close to 128Kbps.

```
Rack1R1#show policy-map interface fastEthernet 0/0
 FastEthernet0/0

  Service-policy output: POLICY_VLAN146_OUT

    Class-map: HTTP (match-all)
      3225 packets, 1897929 bytes
      30 second offered rate 124000 bps, drop rate 0 bps
      Match: access-group 180
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           128000/64000     1600   6400      6400      100       1600

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      0         3225      1897929   348       205320    no

    Class-map: class-default (match-any)
      29 packets, 4378 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

Now start the second file transfer from R6 to SW4.

```
Rack1SW4#copy http://admin:cisco@155.1.146.6/c1841-adventerprisek9-mz.124-
24.T.bin null:
```

Check the accumulated statistics on R4 once again. Note that now both classes conform to their guaranteed rate, thanks to the fair-sharing of the serial link between R4 and R5.

```
Rack1R4#show policy-map interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy input: POLICE_VLAN146

    Class-map: HTTP (match-all)
      35113 packets, 20715559 bytes
      30 second offered rate 126000 bps, drop rate 0 bps
      Match: access-group name HTTP

      Service-policy : SUBRATE_POLICER

        Class-map: FROM_R1 (match-all)
          29986 packets, 17691740 bytes
          30 second offered rate 63000 bps, drop rate 0 bps
          Match: access-group name FROM_R1
          police:
              cir 64000 bps, bc 3200 bytes
              pir 128000 bps, be 6400 bytes
            conformed 18466 packets, 10894940 bytes; actions:
              set-prec-transmit 1
            exceeded 11520 packets, 6796800 bytes; actions:
              set-prec-transmit 0
            violated 0 packets, 0 bytes; actions:
              drop
            conformed 63000 bps, exceed 0 bps, violate 0 bps

        Class-map: FROM_R6 (match-all)
          5127 packets, 3023819 bytes
          30 second offered rate 63000 bps, drop rate 0 bps
          Match: access-group name FROM_R6
          police:
              cir 64000 bps, bc 3200 bytes
              pir 128000 bps, be 6400 bytes
            conformed 5124 packets, 3022049 bytes; actions:
              set-prec-transmit 1
            exceeded 3 packets, 1770 bytes; actions:
              set-prec-transmit 0
            violated 0 packets, 0 bytes; actions:
              drop
            conformed 63000 bps, exceed 0 bps, violate 0 bps

        Class-map: class-default (match-any)
          0 packets, 0 bytes
          30 second offered rate 0 bps, drop rate 0 bps
          Match: any
```

Copyright © 2010 Internetwork Expert

Now check the statistics for traffic shapers at R1 and R6. This time, both shapers are inactive, and offered rate is close to 64Kbps (the CIR value).

```
Rack1R6#show policy-map interface fastEthernet 0/0.146
 FastEthernet0/0.146

  Service-policy output: POLICY_VLAN146_OUT

    Class-map: HTTP (match-all)
      19407 packets, 10699248 bytes
      5 minute offered rate 58000 bps, drop rate 0 bps
      Match: access-group 180
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 19407/11527758
      shape (peak) cir 64000, bc 6400, be 6400
      target shape rate 128000

    Class-map: class-default (match-any)
      156 packets, 87376 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 156/48360

Rack1R1#show policy-map interface fastEthernet 0/0
 FastEthernet0/0

 Service-policy output: POLICY_VLAN146_OUT

    Class-map: HTTP (match-all)
      33062 packets, 19505469 bytes
      30 second offered rate 63000 bps, drop rate 0 bps
      Match: access-group 180
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           128000/64000     1600   6400      6400      100       1600

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      0         33062     19505469  2632      1552858   no

    Class-map: class-default (match-any)
      7641 packets, 7385752 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

10.46 MQC Percent-Based Policing

- Hard code R1's FastEthernet link to a speed of 10Mbps.
- Limit the traffic entering this link to 10% of this rate with a burst value of 125ms.

Configuration

```
R1:
policy-map POLICE_VLAN146
 class class-default
   police rate percent 10 burst 125 ms
!
interface FastEthernet 0/0
 speed 10
 service-policy input POLICE_VLAN146
```

Verification

Note

Percent based policing allows specifying the policer rate as a percentage of the interface speed. At the same time, you may configure burst sizes in milliseconds duration. The actual burst size will be *InterfaceSpeed*BurstDuration* based on the interface's speed. This method does not use "absolute" values but rather relative settings based on interface parameters. The drawback is limited range of speed values, since CLI does not allow fractional percent values. Thus, you may only step by one percent configuring policed rates.

The metering procedure remains the same as with srTCM or trTCM, you just change the way to set configuration parameters.

To verify, send a stream of 1000 byte ICMP packets from R4 to R1 and shape it to 2Mbps. Select the burst size for the shaper to be less than the policer's burst by approximately one packet size (1000+14 bytes). You may learn the policer burst size by using the **`show policy-map`** command.

Based on that, the shaper burst is (15625-1014)*8 = 117000 bps, but we'll make it a bit smaller, around 116000 bps.

Copyright © 2010 Internetwork Expert

```
R4:
interface FastEthernet 0/1
 traffic-shape rate 2000000 116000
```

```
Rack1R4#ping 155.1.146.1 size 1000 timeout 0 repeat 100000000
```

Check the policer statistics to view number of conforming and exceeding packets. Note the amount is approximately the same, and conformed rate is getting close to 1Mbps (which is 10% of interface bandwidth), however we cannot reach the exact metering precision due to small burst size mismatches and jitter.

```
Rack1R1#show policy-map interface fastEthernet 0/0
 FastEthernet0/0

  Service-policy input: POLICE_VLAN146

    Class-map: class-default (match-any)
      40769 packets, 41308718 bytes
      30 second offered rate 1791000 bps, drop rate 889000 bps
      Match: any
      police:
          rate 10 % burst 125 ms
          rate 1000000 bps, burst 15625 bytes
        conformed 20535 packets, 20791442 bytes; actions:
          transmit
        exceeded 20234 packets, 20517276 bytes; actions:
          drop
        conformed 900000 bps, exceed 889000 bps
```

Copyright © 2010 Internetwork Expert

10.47 MQC Header Compression

- Enable MQC based RTP header compression on the serial link between R1 and R3.
- Assume that RTP packets use the port range 16384-32767.
- Allocate enough priority bandwidth to allow 2 G.729 compressed calls over this link, for a total of 24Kbps.

Configuration

```
R1:
class-map VOICE_BEARER
 match ip rtp 16384 16383
!
policy-map SERIAL_LINK_OUT
 class VOICE_BEARER
   priority 24
   compression header ip rtp
!
interface Serial 0/1
 bandwidth 128
 service-policy output SERIAL_LINK_OUT

R3:
class-map VOICE_BEARER
 match ip rtp 16384 16383
!
policy-map SERIAL_LINK_OUT
 class VOICE_BEARER
   priority 24
   compression header ip rtp
!
interface Serial 1/2
 bandwidth 128
 clock rate 128000
 service-policy output SERIAL_LINK_OUT
```

Copyright © 2010 Internetwork Expert

Verification

Note

The functionality of MQC based RTP header compression is the same as in the legacy case. It is important that the MQC policer for the priority queue accounts for the effect of header compression, therefore when configuring bandwidth for the priority queue, base your calculations on compressed packet sizes.

The RTP header compression process reduces IP/UDP/RTP headers from 40 bytes down to 2 bytes, as Cisco does not preserve the UDP checksum. In the case of G.729 and HDLC the resulting bandwidth for one calls is (2*10+2+7)*50*8 = 11600 bps.

The computation assumes the fact that G.729 generates 50pps with 20 bytes of payload each. The total overhead for HDLC is 7 bytes. We round up the value in bps to the next whole number, which is 12Kbps. Two calls will then consume 24Kbps.

Note that MQC based TCP header compression can be configured in this same manner.

```
Rack1R1#show policy-map interface serial 0/1
 Serial0/1

  Service-policy output: SERIAL_LINK_OUT

    Class-map: VOICE_BEARER (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: ip rtp 16384 16383
      Queueing
        Strict Priority
        Output Queue: Conversation 40
        Bandwidth 24 (kbps) Burst 600 (Bytes)
        (pkts matched/bytes matched) 0/0
        (total drops/bytes drops) 0/0
      compress:
          header ip rtp
          UDP/RTP (compression on, IPHC, RTP)
            Sent:    0 total, 0 compressed,
                     0 bytes saved, 0 bytes sent
                     rate 0 bps

    Class-map: class-default (match-any)
      108 packets, 23328 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

```
Rack1R3# show policy-map interface serial 1/2
 Serial1/2

  Service-policy output: SERIAL_LINK_OUT

    Class-map: VOICE_BEARER (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: ip rtp 16384 16383
      Queueing
        Strict Priority
        Output Queue: Conversation 40
        Bandwidth 24 (kbps) Burst 600 (Bytes)
        (pkts matched/bytes matched) 0/0
        (total drops/bytes drops) 0/0
      compress:
          header ip rtp
          UDP/RTP (compression on, IPHC, RTP)
            Sent:    0 total, 0 compressed,
                     0 bytes saved, 0 bytes sent
                     rate 0 bps


    Class-map: class-default (match-any)
      91 packets, 15148 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Use the legacy commands to verify the configuration as well.

```
Rack1R1#show ip rtp header-compression
RTP/UDP/IP header compression statistics:
We're compressing using MQC profiles, use the
MQC commands to see the stats for each class.
  Interface Serial0/1 (compression on, IPHC, RTP)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 32 rx slots, 32 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 32 free
contexts
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.48 Using Class-Based GTS for FRTS

- Configure R3 and R5 to shape traffic going out to the Frame Relay network using MQC based GTS.
- R3's policy should match PVC 305, and shape to an average rate of 256Kbps.
- R5's policy should match PVC 503 with a peak shaping rate of 128Kbps, and match PVC 502 with a peak shaping rate of 256Kbps.
- R3 should use a Bc and Be of 12800 and 6400 for the shaper to R5.
- R5 should use a Bc and Be of 6400 for the shaper to R3, and a Bc and Be of 12800 for the shaper to R2.

Configuration

```
R3:
class-map DLCI_305
 match fr-dlci 305
!
policy-map POLICY_FR_OUT
 class DLCI_305
  shape average 256000 12800 6400
!
interface Serial 1/0
 service-policy output POLICY_FR_OUT

R5:
class-map DLCI_502
 match fr-dlci 502
!
class-map DLCI_503
 match fr-dlci 503
!
policy-map POLICY_FR_OUT
 class DLCI_503
  shape peak 128000 6400 6400
 class DLCI_502
  shape peak 256000 12800 12800
!
interface Serial 0/0/0
  service-policy output POLICY_FR_OUT
```

Copyright © 2010 Internetwork Expert

Verification

Note

You can use Class-Based GTS (CBTS) to simulate Frame-Relay traffic shaping. The core of this functionality is the **`match fr-dlci`** command that allows matching a specific DLCI and classifying traffic on per-VC basis. However, some important functionality is missing, for example the **`shape adaptive`** command does not work. Unlike legacy GTS, CBTS allows shaping on per-VC basis. Some of the differences between legacy GTS, legacy FRTS, and CBTS are as follows.

- You can apply legacy GTS per interface or sub-interface only, and the queuing strategy is fixed to WFQ. You may apply CBTS to sub-interfaces as well as to selected PVCs.
- FRTS turns interface queue into FIFO, but allows using PQ/CQ/WFQ/CBWFQ as per-VC queues. With CBTS, interface queue is either FIFO or CBWFQ. However, you may also use the legacy queuing methods. The per-VC queue is limited to CBWFQ with CBTS.
- CBTS does not affect PVCs not matched by configured class-maps. Legacy FRTS enforces a default rate on all PVCs bound to an interface.
- Legacy FRTS allows per-VC fragmentation, while CBTS only supports per-interface fragmentation.
- CBTS does not support adaptive Frame-Relay traffic shaping.

Copyright © 2010 Internetwork Expert

This configuration can be verified as follows:

```
Rack1R5#show policy-map interface serial 0/0/0

 Serial0/0/0

  Service-policy output: POLICY_FR_OUT

    Class-map: DLCI_503 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: fr-dlci 503
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           256000/128000    1600   6400      6400      50        1600

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      0         0         0         0         0         no

    Class-map: DLCI_502 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: fr-dlci 502
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           512000/256000    3200   12800     12800     50        3200

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      0         0         0         0         0         no

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Note the PIR/CIR rate in the output above. Now verify traffic-shaping queues. Note that CBWFQ bandwidth bases on the CIR value, not the PIR rate (128 and 256K respectively).

Copyright © 2010 Internetwork Expert

```
Rack1R5#show traffic-shape queue
Traffic queued in shaping queue on Serial0/0/0
 Traffic shape class: DLCI_503
  Queueing strategy: weighted fair
  Queueing Stats: 0/1000/64/0 (size/max total/threshold/drops)
     Conversations  0/0/16 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 128 kilobits/sec


 Traffic shape class: DLCI_502
  Queueing strategy: weighted fair
  Queueing Stats: 0/1000/64/0 (size/max total/threshold/drops)
     Conversations  0/0/32 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 256 kilobits/sec
```

```
Rack1R3#show policy-map interface serial 1/0

 Serial1/0

  Service-policy output: POLICY_FR_OUT

    Class-map: DLCI_305 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: fr-dlci 305
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           256000/256000    2400   12800     6400      50        1600

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      0         0         0         0         0         no

    Class-map: class-default (match-any)
      645 packets, 8385 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

```
Rack1R3#show traffic-shape queue
Traffic queued in shaping queue on Serial1/0
 Traffic shape class: DLCI_305
  Queueing strategy: weighted fair
  Queueing Stats: 0/1000/64/0 (size/max total/threshold/drops)
     Conversations  0/0/32 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 256 kilobits/sec
```

Copyright © 2010 Internetwork Expert

10.49 MQC Based Frame-Relay DE-Marking

- Modify R5's shaper so that the Frame Relay Discard Eligible bit is set on all frames going out DLCI 502.

Configuration

```
R5:
policy-map POLICY_FR_OUT
 class DLCI_502
   shape peak 256000 12800 12800
   set fr-de
!
interface Serial 0/0/0
  service-policy output POLICY_FR_OUT
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Unlike the legacy DE-marking procedure, MQC-based DE marking applies to all packet switching paths, including CEF. In this particular case, we apply DE-marking to all traffic that exits DLCI 502. To verify this generate some packets over DLCI 502 and see if they are marked.

```
Rack1R5#show policy-map interface serial 0/0/0

 Serial0/0/0

  Service-policy output: POLICY_FR_OUT

    Class-map: DLCI_503 (match-all)
      7 packets, 1672 bytes
      5 minute offered rate 1000 bps, drop rate 0 bps
      Match: fr-dlci 503
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           256000/128000    1600   6400      6400      50        1600

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      0         6         1616      0         0         no

    Class-map: DLCI_502 (match-all)
      17 packets, 2712 bytes
      5 minute offered rate 1000 bps, drop rate 0 bps
      Match: fr-dlci 502
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           512000/256000    3200   12800     12800     50        3200

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      0         16        2656      0         0         no
      QoS Set
        fr-de
          Packets marked 17

    Class-map: class-default (match-any)
      20 packets, 3423 bytes
      5 minute offered rate 2000 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

10.50 Using MQC CBWFQ with Legacy FRTS

- Configure legacy FRTS on R2's connection to DLCI 205 as follows:
  - Shape to the average rate of 512Kbps
  - Use an interval of 10ms
  - Adaptively shape down to 256Kbps if BECNs are received
- Configure an MQC policy inside this legacy shaper as follows:
  - Guarantee 64Kbps of PVC bandwidth to HTTP flows
  - Other unmatched flows should be guaranteed 192Kbps

Configuration

```
R2:
!
ip http server
ip http path flash:
ip tcp mss 100
!
class-map HTTP
 match protocol http
!
policy-map CBWFQ
 class HTTP
  bandwidth 64
 class class-default
  bandwidth 192
!
map-class frame-relay DLCI_205
 frame-relay cir 512000
 frame-relay bc 25600
 frame-relay be 0
 frame-relay mincir 256000
 frame-relay adaptive-shaping becn
 service-policy output CBWFQ
!
interface Serial 0/0
 frame-relay traffic-shaping
 frame-relay interface-dlci 205
  class DLCI_205
```

Copyright © 2010 Internetwork Expert

Verification

Note

Even though the preferred way of configuring Frame-Relay Traffic Shaping and CBWFQ is through MQC Based FRTS, a CBWFQ policy may still be applied as a per-VC queuing policy by calling it from a legacy FRTS map-class. The MQC policy specifies CBWFQ weights using the unified MQC syntax. In this case, we use NBAR to classify HTTP traffic.

CBWFQ as the per-VC queue is an extension of using simple WFQ or WFQ with IP RTP priority. You may use all CBWFQ features, including LLQ, however the legacy FRTS still performs the shaping function. This means that the system automatically enforces a 56Kbps rate on all PVCs unless you explicitly assign map-classes for them.

One important feature for per-VC CBWFQ is that IOS deduces the "available bandwidth" for CBWFQ based on the **mincir** settings. By default minCIR=CIR/2, so you may need to adjust this value if you do not do oversubscription.

To verify the policy-map status and statistics you may no longer use the **show policy-map** command. Instead, the **show frame-relay pvc** command is used to see the policy-map attached to the respective PVC.

```
Rack1R2#show frame-relay pvc 205

PVC Statistics for interface Serial0/0 (Frame Relay DTE)

DLCI = 205, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0

  input pkts 14337         output pkts 7194         in bytes 4527272
  out bytes 1982084        dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 12778         out DE pkts 0
  out bcast pkts 7184      out bcast bytes 1981044
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  Shaping adapts to BECN
  pvc create time 1d01h, last time pvc status changed 16:25:16
  cir 512000    bc 25600     be 0         byte limit 3200   interval 50
  mincir 256000    byte increment 3200  Adaptive Shaping BECN
  pkts 16        bytes 4416      pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  service policy CBWFQ
 Serial0/0: DLCI 205 -
```

```
 Service-policy output: CBWFQ

    Class-map: HTTP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol http
      Queueing
        Output Queue: Conversation 41
        Bandwidth 64 (kbps)Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0

    Class-map: class-default (match-any)
      7 packets, 1932 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
        Output Queue: Conversation 42
        Bandwidth 192 (kbps)Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
  Output queue size 0/max total 600/drops 0
```

Generate some HTTP traffic across R2 and add another ICMP flow to verify if traffic matches the configured policy.

```
Rack1R5#copy http://admin:cisco@155.1.0.2/c2600-adventerprisek9-mz.124-10.bin
null:
Loading http://***********@155.1.0.2/c2600-adventerprisek9-mz.124-10.bin !!!!!

Rack1SW2#ping 155.1.0.2 size 500 repeat 10000 timeout 1

Type escape sequence to abort.
Sending 10000, 500-byte ICMP Echos to 155.1.0.2, timeout is 1 seconds:
!!!!!
```

Verify queue statistics once again and see traffic matches for both classes in the PVC service-policy.

```
Rack1R2#show frame-relay pvc 205

PVC Statistics for interface Serial0/0 (Frame Relay DTE)

DLCI = 205, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0

  input pkts 22008          output pkts 12045        in bytes 5466236
  out bytes 7718533         dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                  out bytes dropped 0
  in FECN pkts 0            in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0           in DE pkts 20449         out DE pkts 0
  out bcast pkts 7375       out bcast bytes 2033760
  5 minute input rate 10000 bits/sec, 10 packets/sec
  5 minute output rate 61000 bits/sec, 6 packets/sec
  Shaping adapts to BECN
  pvc create time 1d02h, last time pvc status changed 16:54:44
  cir 512000    bc 25600     be 0         byte limit 3200   interval 50
  mincir 256000    byte increment 3200  Adaptive Shaping BECN
  pkts 4867      bytes 5740865   pkts delayed 2         bytes delayed 3008
  shaping inactive
  traffic shaping drops 0
  service policy CBWFQ
 Serial0/0: DLCI 205 -

  Service-policy output: CBWFQ

    Class-map: HTTP (match-all)
      3584 packets, 5146633 bytes
      5 minute offered rate 52000 bps, drop rate 0 bps
      Match: protocol http
      Queueing
        Output Queue: Conversation 41
        Bandwidth 64 (kbps)Max Threshold 64 (packets)
        (pkts matched/bytes matched) 2/3008
        (depth/total drops/no-buffer drops) 0/0/0

    Class-map: class-default (match-any)
      1282 packets, 600300 bytes
      5 minute offered rate 7000 bps, drop rate 0 bps
      Match: any
      Queueing
        Output Queue: Conversation 42
        Bandwidth 192 (kbps)Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
  Output queue size 0/max total 600/drops 0
```

Copyright © 2010 Internetwork Expert

10.51 MQC Compatible FRF.12 Fragmentation

- Configure R2, R3, and R5 to fragment packets to 480 bytes as they are sent out to the Frame Relay network.
- Enable fragmentation on the main Frame-Relay interfaces of R3 and R4.
- Enable fragmentation under the map-class for DLCI 205 on R2.

Configuration

```
R2:
map-class frame-relay DLCI_205
 frame-relay fragment 480

R3:
interface Serial 1/0
  frame-relay fragment 480 end-to-end

R5:
interface Serial 0/0/0
  frame-relay fragment 480 end-to-end
```

Verification

Note

The optimal fragment size ensures the fragment serialization delay of 10ms. Based on the *slowest* link speed we can calculate the fragment size as:

FragSize = 384000*0,01/8 = 480 bytes

MQC only allows enabling FRF.12 fragmentation at the interface level. This command is incompatible with legacy FRTS enabled at interface level. FRF.12 with MQC performs the same function it does with legacy FRTS. However, this time fragment-interleaving queue is hidden, and you cannot observe it with show commands. MQC restricts this feature for use with the whole interface only – you cannot enable it per VC. However, it makes perfect sense, since enabling fragmentation on just one PVC is not effective, until you enable it on all other PVCs.

Copyright © 2010 Internetwork Expert

To verify, ensure you send and receive fragmented packets on all participating routers.

```
Rack1R5#show frame-relay fragment
interface                dlci frag-type  size in-frag    out-frag   dropped-
frag
Se0/0/0                  501  end-to-end 480  0          130        0
Se0/0/0                  502  end-to-end 480  4378       4506       0
Se0/0/0                  503  end-to-end 480  0          130        0
Se0/0/0                  504  end-to-end 480  0          130        0
Se0/0/0                  513  end-to-end 480  0          0          0

Rack1R2#show frame-relay fragment
interface                dlci frag-type  size in-frag    out-frag   dropped-
frag
Se0/0                    205  end-to-end 480  4814       4837       0

Rack1R3#ping 155.1.0.5 size 500

Type escape sequence to abort.
Sending 5, 500-byte ICMP Echos to 155.1.0.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 260/271/316 ms

Rack1R3#show frame-relay fragment
interface                dlci frag-type  size in-frag    out-frag   dropped-
frag
Se1/0                    301  end-to-end 480  0          0          0
Se1/0                    302  end-to-end 480  0          0          0
Se1/0                    304  end-to-end 480  0          0          0
Se1/0                    305  end-to-end 480  328        10         0
```

Note that we had to stream large ICMP packets off R3 to increase the fragment counters, as the RIP updates are to small to fragment them.

Copyright © 2010 Internetwork Expert

10.52 MQC Based Frame-Relay Traffic Shaping

- Modify the traffic-shaping configuration on R5 to remove the classes matching the Frame-Relay DLCIs.
- Configure MQC shaping on R5 to apply per-VC as follows, but do not enable legacy FRTS on the main interface
    - Shape peak with CIR values 128000 and 256000 for DLCI 503 and DLCI 502
    - Select a B_c value based on a T_c of 50ms, and a B_e value so that PIR=2*CIR
    - Set the Frame-Relay DE bit for traffic going out DLCI 502
- Replace the interface-based fragmentation with per-VC fragmentation.

Configuration

```
R5:
policy-map SHAPE_DLCI_503
 class class-default
  shape peak 128000 6400 6400
  shape adaptive 128000
!
policy-map SHAPE_DLCI_502
 class class-default
  shape peak 256000 12800 12800
  shape adaptive 256000
  set fr-de
!
! MQC based FRTS still uses map-classes
! but applies shaping using CBTS commands
! inside a service-policy.
! Fragmentation also applies inside a map-class
!
map-class frame-relay DLCI_503
 frame-relay fragment 480
 service-policy output SHAPE_DLCI_503
!
map-class frame-relay DLCI_502
 frame-relay fragment 480
 service-policy output SHAPE_DLCI_502
!
! We no longer configure fragmentation at the interface level
!
interface Serial 0/0/0
 no frame-relay fragment 480 end-to-end
 no service-policy output POLICY_FR_OUT
 frame-relay interface-dlci 503
  class DLCI_503
 frame-relay interface-dlci 502
  class DLCI_502
```

Copyright © 2010 Internetwork Expert

Verification

Note

MQC based Frame-Relay traffic shaping is a hybrid form of legacy FRTS and Class-Based Traffic Shaping (CBTS). Instead of using MQC-based syntax for classification, it relies on map-classes bound to the interface's DLCIs. MQC statements specify the shaping parameters, such as average/peak rate. Additionally, it is possible to enable Frame-Relay adaptive traffic shaping and FECN reflection, which CBTS does not support. Additionally, you should not use the **frame-relay traffic-shaping** interface level command with the MQC-based FRTS.

As in the case of CBTS, you may adjust shaper queue by applying a second-level policy map with CBWFQ parameters. This is similar to the use of CBWFQ with legacy FRTS but this time you use MQC syntax to define shaping parameters.

In this scenario, we oversubscribe both PVCs, thus we use the **shape peak** command combined with **shape adaptive** to configure a PVC to send at PIR, while ensuring it throttles down to CIR in case of congestion notification.

The verification commands are slightly different from the ones you use to verify legacy FRTS, as seen below.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show frame-relay pvc 502

PVC Statistics for interface Serial0/0 (Frame Relay DTE)

DLCI = 502, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0

  input pkts 29950          output pkts 46932        in bytes 22814609
  out bytes 11804996        dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0            in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0           in DE pkts 0             out DE pkts 45369
  out bcast pkts 18637      out bcast bytes 5888192
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 1d07h, last time pvc status changed 21:56:46
  fragment type end-to-end fragment size 480
  service policy SHAPE_DLCI_502
Serial0/0: DLCI 502 -

  Service-policy output: SHAPE_DLCI_502

    Class-map: class-default (match-any)
      26 packets, 8216 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           512000/256000    3200   12800     12800     50        3200

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        BECN   0         1019      976568    0         0         no
      QoS Set
        fr-de
          Packets marked 26
  Output queue size 0/max total 600/drops 0
```

```
Rack1R5#show frame-relay pvc 503

PVC Statistics for interface Serial0/0 (Frame Relay DTE)

DLCI = 503, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0

  input pkts 107786        output pkts 99529        in bytes 9809512
  out bytes 51081671       dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 164
  out bcast pkts 18639     out bcast bytes 5888824
  5 minute input rate 5000 bits/sec, 9 packets/sec
  5 minute output rate 57000 bits/sec, 6 packets/sec
  pvc create time 1d07h, last time pvc status changed 21:56:51
  fragment type end-to-end fragment size 480
  service policy SHAPE_DLCI_503
 Serial0/0: DLCI 503 -

  Service-policy output: SHAPE_DLCI_503

    Class-map: class-default (match-any)
      818 packets, 954140 bytes
      5 minute offered rate 30000 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           256000/128000    1600   6400      6400      50        1600

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        BECN   0         0         0         0         0         no
  Output queue size 0/max total 600/drops 0
```

Note the CIR/PIR values in this output, as well as "Adapt Active" field, which tells that PVCs will respond to BECN notifications. FRF.12 fragmentation can be verified as follows.

```
Rack1R5#show frame-relay fragment
interface              dlci frag-type  size in-frag    out-frag   dropped-frag
Se0/0/0                502  end-to-end 480  0          734        0
Se0/0/0                503  end-to-end 480  0          19         0
```

Copyright © 2010 Internetwork Expert

10.53 Voice Adaptive Traffic Shaping

- Since the provider oversubscribes both PVCs on R5's interface, voice quality may degrade when sending traffic at rates over the CIR. Since it is undesirable to turn off oversubscription and lose this extra bandwidth, provide a solution to throttle down the sending rates on R3 and R5 to the CIR, but only when the IOS detects traffic in the LLQ queue.
- Leave the shaping time interval at 50ms for simplicity.
- Assume that VoIP traffic is already marked as DSCP EF.
- This traffic should be prioritized up to 64Kbps on R2, R3, and R5.

Configuration

```
R2:
!
! R2 uses legacy FRTS
!
class-map VOICE_BEARER
 match dscp ef
!
policy-map CBWFQ
 class class-default
  bandwidth 128
 class VOICE_BEARER
  priority 64
!
map-class frame-relay DLCI_205
 service-policy output CBWFQ

R3:
!
! R3 uses CBTS for traffic shaping
!
class-map VOICE_BEARER
 match dscp ef
!
policy-map LLQ
 class VOICE_BEARER
  priority 64
!
policy-map POLICY_FR_OUT
 class DLCI_305
  shape average 256000 12800 6400
  service-policy LLQ
```

Copyright © 2010 Internetwork Expert

```
R5:
!
! R5 uses MQC-based FRTS, VATS applies here
!
class-map VOICE_BEARER
 match dscp ef
!
policy-map LLQ
 class VOICE_BEARER
  priority 64
!
! Add LLQ to shaper queue
! Enable VATS in the class
!
policy-map SHAPE_DLCI_503
 class class-default
  shape peak 128000 6400 6400
  shape adaptive 128000
  shape fr-voice-adapt
  service-policy LLQ
!
policy-map SHAPE_DLCI_502
 class class-default
  shape peak 256000 12800 12800
  shape adaptive 256000
  shape fr-voice-adapt
  service-policy LLQ
```

Verification

Note

Voice Adaptive Traffic Shaping (VATS) is only available with MQC-based Frame-Relay traffic shaping, and only with LLQ enabled in shaper-queue. Thus, in our scenario, only R5 is capable of doing VATS.

The goal of VATS is to throttle the PVC's sending rate once the system detects voice packets in the priority queue. The VATS feature becomes useful in oversubscription scenarios, where the customer constantly sends traffic over the CIR. In cases when the LLQ enqueues voice packets, the system signals the shaper to slow its sending rate. This ensures that router only sends at the guaranteed rate, and the provider will never drop or delay any voice packets.

The syntax for the command applied under policy-map class is **shape fr-voice-adapt [deactivation** <*N*>**]**, where *N* specifies the number of seconds required to deactivate shaping after the last packet has been seen in priority queue (LLQ). The default value is 30 seconds.

Copyright © 2010 Internetwork Expert

To verify this scenario, adjust the PIR for DLCI 503 to 56Kbps, and the CIR to 32Kbps. This will match them properly against the connection's physical rate of 64Kbps. Next, generate a flow of ICMP packets marked with DSCP EF from SW2 to R3 to simulate voice traffic. Note that the TOS byte of 184 corresponds to DSCP 46 (EF). In addition to that, start transferring the IOS image from R5 down to R3 to cause congestion in the shaper.

```
R5:
policy-map LLQ
 class VOICE_BEARER
  priority 32
!
policy-map SHAPE_DLCI_503
 class class-default
  shape peak 32000 6400 4800
  shape adaptive 32000
  shape fr-voice-adapt
  service-policy LLQ

Rack1SW2#ping
Protocol [ip]:
Target IP address: 155.1.0.3
Repeat count [5]: 10000000
Datagram size [100]: 60
Timeout in seconds [2]: 1
Extended commands [n]: y
Source address or interface:
Type of service [0]: 184
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 10000000, 60-byte ICMP Echos to 155.1.0.3, timeout is 1
seconds:
!!!!!!!!!!

R5:
ip http server
ip http path flash:

Rack1R3#copy http://admin:cisco@155.1.0.5/c1841-adventerprisek9-mz.124-
24.T.bin null:
Loading http://***********@155.1.0.5/c1841-adventerprisek9-mz.124-
24.T.bin !!!
```

Copyright © 2010 Internetwork Expert

Now check the policy-map statistics on R5. Note that the shaper's offered rate is close to 32Kbps, since the system activated VATS on this PVC. Every new voice packet refreshes VATS times back to 30 seconds.

```
Rack1R5#show policy-map interface serial 0/0/0
 Serial0/0/0: DLCI 503 -

  Service-policy output: SHAPE_DLCI_503

    Class-map: class-default (match-any)
      737754 packets, 73689969 bytes
      5 minute offered rate 32000 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
            56000/32000     1400   6400      4800      200       1400

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                        Delayed   Delayed   Active
        BECN   2         737752    73688401  3132      267876    yes
        Voice Adaptive Shaping active, time left 29 secs

      Service-policy : LLQ

        Class-map: VOICE_BEARER (match-all)
          555731 packets, 35566784 bytes
          5 minute offered rate 12000 bps, drop rate 0 bps
          Match:  dscp ef (46)
          Queueing
            Strict Priority
            Output Queue: Conversation 24
            Bandwidth 32 (kbps) Burst 800 (Bytes)
            (pkts matched/bytes matched) 555731/35566784
            (total drops/bytes drops) 0/0

        Class-map: class-default (match-any)
          182023 packets, 38123185 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: any
```

Copyright © 2010 Internetwork Expert

Now stop the flow of ICMP packets from SW2 (this flow simulated the voice traffic), and wait for the system to deactivate VATS. Observe shaping statistics once again after this delay:

```
Rack1R5#show policy-map interface serial 0/0/0
 Serial0/0: DLCI 503 -

  Service-policy output: SHAPE_DLCI_503

    Class-map: class-default (match-any)
      739663 packets, 75354437 bytes
      5 minute offered rate 53000 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
            56000/32000     1400   6400      4800      200       1400

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                        Delayed   Delayed   Active
        BECN   2         739660    75352301  4898      1747468   yes
        Voice Adaptive Shaping inactive

      Service-policy : LLQ

        Class-map: VOICE_BEARER (match-all)
          556296 packets, 35602944 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match:  dscp ef (46)
          Queueing
            Strict Priority
            Output Queue: Conversation 24
            Bandwidth 32 (kbps) Burst 800 (Bytes)
            (pkts matched/bytes matched) 556296/35602944
            (total drops/bytes drops) 0/0

        Class-map: class-default (match-any)
          183367 packets, 39751493 bytes
          5 minute offered rate 53000 bps, drop rate 0 bps
          Match: any
```

Note that the shaping rate is now getting close to 56Kbps, since VATS is inactive. As soon as voice packets are back, the system will return to shaping back to CIR.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.54 Voice Adaptive Fragmentation

- Configure R5 to activate fragmentation, but only if voice packets are present in the LLQ queue.

Configuration

```
R5:
interface Serial 0/0/0
 frame-relay fragmentation voice-adaptive
```

Verification

> **Note**
>
> Fragmentation is useful only in the presence of VoIP packets – otherwise it just wastes router resources and link bandwidth due to additional overhead. Voice Adaptive Fragmentation (VAF) allows fragmenting of large data packets only in the presence of voice packets in the LLQ. Based on this the VAF feature helps in situations when processing a high load of fragmented packets becomes CPU intensive.
>
> VAF only works with MQC-based FRTS, and supports fragmentation configured in a map-class (legacy style) or at the interface level (MQC-compatible style). The syntax for interface-level command is **frame-relay fragmentation voice-adaptive [deactivation** <*N*>**]**, where *N* specifies the number of seconds required to deactivate fragmentation after the last packet has been seen in priority queue (LLQ). The default value is 30 seconds.
>
> To verify this scenario adjust the PIR for DLCI 503 to 56Kbps and the CIR to 32Kbps. This will match them properly against the connection's physical rate of 64Kbps. After that, generate a flow of ICMP packets marked with DSCP EF from SW2 to R3 to simulate voice traffic. Note that TOS byte of 184 corresponds to DSCP 46 (EF).

```
R5:
policy-map LLQ
 class VOICE_BEARER
  priority 32
!
policy-map SHAPE_DLCI_503
 class class-default
  shape peak 32000 6400 4800
  shape adaptive 32000
!
map-class frame-relay DLCI_503
 frame-relay fragment 80
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#ping
Protocol [ip]:
Target IP address: 155.1.0.3
Repeat count [5]: 10000000
Datagram size [100]: 60
Timeout in seconds [2]: 1
Extended commands [n]: y
Source address or interface:
Type of service [0]: 184
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 10000000, 60-byte ICMP Echos to 155.1.0.3, timeout is 1 seconds:
!!!!!!!!!!
```

Check that VAF is active on PVC 503 now:

```
Rack1R5#show frame-relay pvc 503

PVC Statistics for interface Serial0/0/0 (Frame Relay DTE)

DLCI = 503, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0

  input pkts 769006        output pkts 767494       in bytes 74548828
  out bytes 87236829       dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 9361      out bcast bytes 2957736
  5 minute input rate 12000 bits/sec, 24 packets/sec
  5 minute output rate 13000 bits/sec, 24 packets/sec
  pvc create time 12:01:37, last time pvc status changed 12:01:37
  fragment type end-to-end fragment size 80 adaptive active, time left 29 secs
```

Now reset the Frame-Relay statistics and send packets larger than the configured fragment size across the PVC. Verify that the system fragmented the packets:

```
Rack1R5#clear frame-relay counter
Rack1R5#show frame-relay fragment
interface              dlci frag-type  size in-frag    out-frag   dropped-frag
Se0/0/0                  502  end-to-end 480  0          2          0
Se0/0/0                503  end-to-end 80   0          9          0

Rack1R5#ping 155.1.0.3 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 155.1.0.3, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 56/64/368 ms
```

```
Rack1R5#show frame-relay fragment
interface               dlci frag-type  size in-frag    out-frag   dropped-
frag
Se0/0/0                   502  end-to-end 480  0          6          0
Se0/0/0                 503  end-to-end 80   0          227        0
```

Now stop the flow of ICMP packets from SW2 (this flow simulated the voice traffic) and wait for the system to deactivate VAF. Observe fragments statistics again:

```
Rack1R5#show frame-relay pvc 503

PVC Statistics for interface Serial0/0 (Frame Relay DTE)

DLCI = 503, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0

  input pkts 1040          output pkts 1135         in bytes 95088
  out bytes 120326         dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 168       out bcast bytes 53088
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 12:15:38, last time pvc status changed 12:15:38
  fragment type end-to-end fragment size 80 adaptive inactive

Rack1R5#show frame-relay fragment
interface               dlci frag-type  size in-frag    out-frag   dropped-
frag
Se0/0/0                   502  end-to-end 480  0          0          0
Se0/0/0                 503  end-to-end 80   0          0          0

Rack1R5#ping 155.1.0.3 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 155.1.0.3, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 56/59/81 ms
Rack1R5#show frame-relay fragment
interface               dlci frag-type  size in-frag    out-frag   dropped-
frag
Se0/0/0                 502  end-to-end 480  0          0          0
Se0/0/0                 503  end-to-end 80   0          0          0
```

Based on this we can conclude that fragmentation only occurs when voice packets are present in the LLQ.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.55 MLPPP LFI over Frame Relay

- Configure PPP over Frame Relay on the back-to-back Serial link between R4 and R5 using DLCI 100.
- Configure FRTS on both routers assuming a PVC CIR of 256Kbps, and an interface rate of 512Kbps.
- Create two class-maps named VOICE_BEARER and VOICE_SIGNALING matching the DSCP values of EF and CS3 respectively on both routers.
- Create a policy-map named MLPPP_LLQ and configure it as follows:
  - VOICE_BEARER class should provide enough priority bandwidth to accommodate two G.729 calls (56Kbps)
  - VOICE_SIGNALING class should receive 5% of the remaining bandwidth
  - Class-default should receive 95% of the remaining bandwidth
- Enable PPP multilink and interleave on the virtual-templates and configure delay of 10ms.
- Apply the service-policy to the virtual-templates.

Configuration

```
R4:
class-map match-all VOICE_BEARER
 match  dscp ef
!
class-map match-all VOICE_SIGNALING
 match  dscp cs3
!
! 56Kbps for 2 G.729 calls
!
policy-map MLPPP_LLQ
 class VOICE_BEARER
  priority 56
 class VOICE_SIGNALING
  bandwidth remaining percent 5
 class class-default
  bandwidth remaining percent 95
!
! Use MQC FRTS for traffic-shaping
!
policy-map SHAPE_DLCI_100
 class class-default
  shape average 256000 2560 0
!
map-class frame-relay DLCI_100
 service-policy output SHAPE_DLCI_100
!
interface Serial 0/1/0
 encapsulation frame-relay
 no keepalive
 no ip address
```

Copyright © 2010 Internetwork Expert

```
 frame-relay interface-dlci 100 ppp Virtual-Template 1
  class DLCI_100

!
! Apply LLQ to the virtual-template
! Adjust bandwidth to 512Kbps
!
interface Virtual-Template 1
 ip address 155.1.45.4 255.255.255.0
 bandwidth 512
 service-policy output MLPPP_LLQ
 ppp multilink
 ppp multilink interleave
 ppp multilink fragment delay 10

R5:
class-map match-all VOICE_BEARER
 match  dscp ef
!
class-map match-all VOICE_SIGNALING
 match  dscp cs3
!
! 56Kbps for voice traffic
!
policy-map MLPPP_LLQ
 class VOICE_BEARER
  priority 56
 class VOICE_SIGNALING
  bandwidth remaining percent 5
 class class-default
  bandwidth remaining percent 95
!
map-class frame-relay DLCI_100
 frame-relay cir 256000
 frame-relay mincir 25600
 frame-relay bc 2560
 frame-relay be 0
!
interface Serial 0/1/0
 encapsulation frame-relay
 no keepalive
 no ip address
 frame-relay traffic-shaping
 frame-relay interface-dlci 100 ppp Virtual-Template 1
  class DLCI_100
!
! Apply LLQ to the virtual-template
! Adjust bandwidth to 512Kbps
!
interface Virtual-Template 1
 ip address 155.1.45.5 255.255.255.0
 bandwidth 512
 service-policy output MLPPP_LLQ
 ppp multilink
 ppp multilink interleave
 ppp multilink fragment delay 10
```

Copyright © 2010 Internetwork Expert

Verification

Note

MLPPP (Multilink PPP) LFI (Link Fragmentation and Interleaving) provides a way to implement a fragmentation and interleaving scheme supported across both Frame-Relay and ATM PVCs. As both Frame-Relay and ATM are considered legacy technologies nowadays, the need for a common fragmentation scheme does not seem to be important. However, many installations still utilize Frame-Relay and ATM interworking schemes, and which is why Cisco IOS still supports this feature.

Keep in mind the following key points about MLPPP.

1) MLPPP requires Frame-Relay traffic shaping for PVCs. Although MLPPP would work without FRTS, it will not provide proper QoS properties, since queuing will not engage at the PPP level.

2) MLPPP requires the LLQ at virtual-interface level. LLQ is essential for interleaving, since it ensures that scheduler services VoIP packets ahead of others.

3) Calculate the fragment size based on interface physical rate, not the PVC CIR. In our case, we simply changed the virtual-template interface bandwidth to match physical rate.

4) PPP performs fragmentation and interleaving using PPP multilink feature. MLPPP never fragments packets smaller than the configured fragments size, but rather send them as regular PPP packets interleaved with fragments of larger packets. The interleaving procedure uses special interface-level dual-FIFO queue, similar to the one used in FRF.12 process.

For voice bandwidth calculation, take into account the MLPPPoFR encapsulation overhead of 9 bytes: 1 byte for FR flag, 2 bytes for Frame-Relay header, 1 byte for PPP LLC Control field, 1 byte for PPP NLPID and 2 bytes for Frame-Relay FCS.

In our case, voice packets are uncompressed G.729 which takes 60 bytes for Payload + IP/UDP/RTP headers, 9 bytes for overhead and have a 50 pps rate. Thus the total Layer 2 bandwidth per call is:

Bandwidth = (60+9)*50*8=27600 bps

Multiply it by 2 to get approximately 56Kbps.

Copyright © 2010 Internetwork Expert

When provisioning CBWFQ bandwidth values, note that we scaled the bandwidth available on the PVC (CIR) to the physical interface rate. This may require scaling the classes' absolute bandwidth values to match the new interface bandwidth, or simply use bandwidth allocation based on remaining percents. As for voice bandwidth, remember the extra overhead added by MLPPP header. We demonstrated the calculations in configuration section.

To verify your configuration, issue the following commands on R4 side (MQC-based FRTS configuration):

```
Rack1R4#show policy-map interface serial 0/1/0
 Serial0/1/0: DLCI 100 -

  Service-policy output: SHAPE_DLCI_100

    Class-map: class-default (match-any)
      19 packets, 2484 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           256000/256000    320    2560      0         10        320

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      0         19        2484      0         0         no

Rack1R4#show ppp multilink

Virtual-Access3, bundle name is Rack1R5
  Endpoint discriminator is Rack1R5
  Bundle up for 01:43:46, total bandwidth 512, load 1/255
  Receive buffer limit 12192 bytes, frag timeout 1000 ms
  Interleaving enabled
    0/0 fragments/bytes in reassembly list
    0 lost fragments, 0 reordered
    0/0 discarded fragments/bytes, 0 lost received
    0x2B3 received sequence, 0x2B3 sent sequence
  Member links: 1 (max not set, min not set)
    Vi1, since 01:43:47, 640 weight, 630 frag size
No inactive multilink interfaces

Rack1R4#show policy-map interface virtual-access 3
 Virtual-Access3

  Service-policy output: MLPPP_LLQ

    Class-map: VOICE_BEARER (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:  dscp ef (46)
      Queueing
        Strict Priority
        Output Queue: Conversation 136
        Bandwidth 56 (kbps) Burst 1400 (Bytes)
        (pkts matched/bytes matched) 0/0
        (total drops/bytes drops) 0/0
```

Copyright © 2010 Internetwork Expert

```
    Class-map: VOICE_SIGNALING (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:  dscp cs3 (24)
      Queueing
        Output Queue: Conversation 137
        Bandwidth remaining 5 (%)Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0

    Class-map: class-default (match-any)
      13 packets, 4082 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
        Output Queue: Conversation 138
        Bandwidth remaining 95 (%)Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
```

Note the CIR value, fragment size and policy-map attached to virtual bundle interface. The **`show ppp multilink`** command displays the bundle interface "virtual-access3" and the member link "virtual-access1". The policy map applies to the bundle interface, while member link queue is FIFO.

Now repeat the similar verification on the side configured for Frame-Relay traffic shaping.

```
Rack1R5#show frame-relay pvc 100

PVC Statistics for interface Serial0/1/0 (Frame Relay DTE)

DLCI = 100, DLCI USAGE = LOCAL, PVC STATUS = STATIC, INTERFACE = Serial0/1

  input pkts 2037          output pkts 2152         in bytes 266496
  out bytes 400154         dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 154       out bcast bytes 48664
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 01:59:33, last time pvc status changed 01:59:33
  Bound to Virtual-Access1 (up, cloned from Virtual-Template1)
  cir 256000    bc 2560      be 0         byte limit 320    interval 10
  mincir 25600     byte increment 320   Adaptive Shaping none
  pkts 1998      bytes 351490    pkts delayed 8         bytes delayed 468
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/40, 0 drop, 0 dequeued
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ppp multilink

Virtual-Access3, bundle name is Rack1R4
  Endpoint discriminator is Rack1R4
  Bundle up for 01:48:01, total bandwidth 512, load 1/255
  Receive buffer limit 12192 bytes, frag timeout 1000 ms
  Interleaving enabled
    0/0 fragments/bytes in reassembly list
    1 lost fragments, 1 reordered
    0/0 discarded fragments/bytes, 0 lost received
    0x2CE received sequence, 0x2CF sent sequence
  Member links: 1 (max not set, min not set)
    Vi1, since 01:48:01, 640 weight, 630 frag size
No inactive multilink interfaces

Rack1R5#show policy-map interface Virtual-Access 3
 Virtual-Access3

  Service-policy output: MLPPP_LLQ

    Class-map: VOICE_BEARER (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:  dscp ef (46)
      Queueing
        Strict Priority
        Output Queue: Conversation 136
        Bandwidth 56 (kbps) Burst 1400 (Bytes)
        (pkts matched/bytes matched) 0/0
        (total drops/bytes drops) 0/0

    Class-map: VOICE_SIGNALING (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:  dscp cs3 (24)
      Queueing
        Output Queue: Conversation 137
        Bandwidth remaining 5 (%)Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0

    Class-map: class-default (match-any)
      34 packets, 15436 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
        Output Queue: Conversation 138
        Bandwidth remaining 95 (%)Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
```

Now identify the interface-level interleaving queue, for example on R4. Note that the queue is essentially the same priority queue seen with FRF.12:

```
Rack1R4#show interfaces serial 0/1/0
Serial0/1 is up, line protocol is up
  Hardware is PowerQUICC Serial
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation FRAME-RELAY, loopback not set
  Keepalive not set
  LMI DLCI 1023  LMI type is CISCO  frame relay DTE
  FR SVC disabled, LAPF state down
  Broadcast queue 0/64, broadcasts sent/dropped 39/0, interface broadcasts 38
  Last input 02:04:14, output 00:00:07, output hang never
  Last clearing of "show interface" counters 02:16:55
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 10
  Queueing strategy: dual fifo
  Output queue: high size/max/dropped 0/256/0
  Output queue: 0/128 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     2491 packets input, 462842 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     1 input errors, 0 CRC, 1 frame, 0 overrun, 0 ignored, 0 abort
     2488 packets output, 322180 bytes, 0 underruns
     0 output errors, 0 collisions, 2 interface resets
     0 output buffer failures, 0 output buffers swapped out
     14 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up

Rack1R4#show queueing interface serial 0/1/0
Interface Serial0/1 queueing strategy: priority

Output queue utilization (queue/count)
        high/0 medium/0 normal/183 low/0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.56 QoS Pre-Classify

- Create a GRE tunnel between R1 and R6 over VLAN 146.
- Route traffic between the Loopback0 subnets of R1 and R6 across the tunnel using static routes.
- Limit the rate of traffic leaving the VLAN 146 interface of R6 to 256Kbps.
- Configure R6 to guarantee 64Kbps of priority bandwidth to IP traffic between the Loopback0 subnets of R1 and R6 marked with DSCP EF on the VLAN146 interface.

Configuration

```
R1:
interface Tunnel0
 tunnel source 155.1.146.1
 tunnel destination 155.1.146.6
 ip unnumbered FastEthernet 0/0
!
ip route 150.1.6.0 255.255.255.0 Tunnel0

R6:
interface Tunnel0
 tunnel source 155.1.146.6
 tunnel destination 155.1.146.1
 ip unnumbered FastEthernet 0/0.146
 qos pre-classify
!
ip route 150.1.1.0 255.255.255.0 Tunnel0
!
ip access-list extended LOOPBACKS
 permit ip 150.1.6.0 0.0.0.255 150.1.1.0 0.0.0.255
!
class-map LOOPBACKS_DSCP_EF
 match access-group name LOOPBACKS
 match dscp ef
!
policy-map LLQ
 class LOOPBACKS_DSCP_EF
!
policy-map SHAPE_VLAN_146
 class class-default
  shape average 256000
  service-policy LLQ
!
interface FastEthernet 0/0.146
 service-policy output SHAPE_VLAN_146
```

Copyright © 2010 Internetwork Expert

Verification

Note

From a QoS classification perspective, traffic routed inside tunnels, such as GRE or IPsec, may pose a serious problem. Imagine a situation where multiple tunnels go across the same physical interface of a router, in addition to non-tunneled traffic leaving the same interface. Additionally you need to limit the aggregate outgoing traffic rate, and share bandwidth between tunneled and non-tunneled traffic. If you apply a service policy to the physical interface, the policy will treat traffic inside the tunnel as a single flow, sourced off the tunnel endpoints addresses. Since the classification takes into account only the top-most header, the system is not able to distinguish traffic flows inside the tunnel. A number of solutions exist to make the router aware of the "tunneled" traffic structure.

1) ToS reflection. When IOS encapsulates an IP packet with a tunnel header, it copies the ToS byte to the tunnel header. This makes possible to distinguish traffic "classes" between the tunnel endpoints. However, specific flows remain "hidden" from the physical interface level policy, limiting the usefulness of flow-based schedulers like WFQ.

2) Applying service-policy to the tunnel interface. IOS allows applying GTS (Generic Traffic Shaping) and CBTS (Class Based Shaping) to GRE/IPIP tunnels. Using CBTS you may shape traffic going across the tunnel and tune the shaper queue the way you want. This method has its limitations. First, not all tunnel technologies support CBTS. For example, you cannot apply CBTS inside an IPsec tunnel or apply it to Multipoint GRE tunnel. More seriously, even if the previous issue could be resolved, you still cannot create one unified aggregate policy for all traffic. That is, you need to limit each tunnel up to its maximum rate and apply CBWFQ inside the tunnel. In addition, you have to configure some policy at the physical interface level, e.g. a bandwidth reservation for tunnel traffic flow. This makes the configuration process complicated.

The last solution is QoS pre-classification. When you turn on this feature on a tunnel interface (GRE/mGRE, IPIP, IPsec, Virtual-Template) you no longer need to apply a service policy inside the tunnel interface. Thanks to QoS pre-classification, the service-policy applied at the interface level can "see" the tunnel encapsulated packets as is they cross the interface without any encapsulation. However, the physical interface level policy still accounts for tunnel header overhead, thus allowing for fair scheduling.

Copyright © 2010 Internetwork Expert

The QoS pre-classify feature is useful in scenarios similar to the described above, e.g. you need to share the physical interface between tunneled and non-tunneled traffic. With pre-classification you apply the service-policy to the physical interface and enable the feature on all tunnel interfaces, treating all traffic (from a QoS prospective) like it is not encapsulated.

To verify this, generate packets from the Loopback0 of R6 to the Loopback0 of R1, since these packets are tunnel encapsulated.

```
Rack1R6#ping
Protocol [ip]:
Target IP address: 150.1.1.1
Repeat count [5]: 100
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: 150.1.6.6
Type of service [0]: 184
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 150.1.1.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.6.6
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 1/3/8 ms

Rack1R6#show ip route 150.1.1.1
Routing entry for 150.1.1.0/24
  Known via "static", distance 1, metric 0 (connected)
  Redistributing via rip
  Advertised by rip
  Routing Descriptor Blocks:
  * directly connected, via Tunnel0
      Route metric is 0, traffic share count is 1

Rack1R6#show policy-map interface fastEthernet 0/0.146
 FastEthernet0/0.146

  Service-policy output: SHAPE_VLAN_146

    Class-map: class-default (match-any)
      150 packets, 29092 bytes
      5 minute offered rate 2000 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           256000/256000    1984   7936      7936      31        992

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      0         150       29092     0         0         no
```

```
Service-policy : LLQ

  Class-map: VOICE (match-all)
    100 packets, 14200 bytes
    5 minute offered rate 2000 bps, drop rate 0 bps
    Match:  dscp ef (46)
    Queueing
      Output Queue: Conversation 41
      Bandwidth 64 (kbps)Max Threshold 64 (packets)
      (pkts matched/bytes matched) 0/0
  (depth/total drops/no-buffer drops) 0/0/0

  Class-map: LOOPBACKS_DSCP_EF (match-all)
    0 packets, 0 bytes
    5 minute offered rate 0 bps
    Match: access-group name LOOPBACKS
    Match:  dscp ef (46)

  Class-map: class-default (match-any)
    50 packets, 14892 bytes
    5 minute offered rate 0 bps, drop rate 0 bps
    Match: any
```

Note the size of the ICMP packets classified by the service-policy. A total of 14200 bytes for 100 packets means a size of 142 bytes for each packet. Subtract 18 bytes for the tagged Ethernet frame, and the remaining 24 bytes are the for GRE encapsulation header. This means that QoS pre-classification correctly accounts for the tunnel overhead.

Copyright © 2010 Internetwork Expert

> **Note**
>
> Load the *QoS* initial configurations prior to starting.

10.57 RSVP and WFQ

- Temporarily Shutdown R4's and R5's Frame Relay connection.
- Configure the link bandwidth between R4 and R5 as 128Kbps.
- Enable RSVP on the connections to VLAN 146 on R1, R4, and R6.
- Allow RSVP to use the maximum possible bandwidth on the link between R4 and R5, but do not allow it to reserve more than 64Kbps per flow.

Configuration

```
R1:
interface FastEthernet 0/0
 ip rsvp bandwidth

R4:
interface FastEthernet 0/1
 ip rsvp bandwidth
!
interface Serial 0/1/0
 bandwidth 128
 ip rsvp bandwidth 96 64

R5:
interface Serial 0/1/0
 bandwidth 128
 clock rate 128000
 ip rsvp bandwidth 96 64
!
interface Serial 0/0/0
 shutdown

R6:
interface FastEthernet 0/0
 ip rsvp bandwidth
!
interface FastEthernet 0/0.146
 ip rsvp bandwidth
```

Copyright © 2010 Internetwork Expert

Verification

Note

RSVP, or Resource Reservation Protocol, is the core of Integrated Services (IntServ) QoS model, also known as the end-to-end QoS model. The general idea is to mimic the behavior of circuit-switched networks, using RSVP as a signaling protocol to request certain QoS capabilities from the network. Compare this to a PSTN cloud, where the network reserves resources for a call for the duration of the conversation and releases them as the call ends. This in contrast with classic packet switched network (PSN) behavior, where the network statistically multiplexes packets and provides per-hop treatment for each packet, irrespective of the "flow" the packet belongs to. The latter behavior evolved into the Differentiated Services (DiffServ) QoS model.

Central to RSVP is the concept of a "flow" – a unidirectional stream of packets. A flow could be either one-to-one or one-to-many - e.g. if a sender streams a multicast flow with many receivers. When a host X wants to stream a flow toward host Y (Y could be a multicast address), it sends RSVP request (PATH message) to the nearest router, asking to establish QoS-aware path downstream to Y. The request contains information describing the flow the sender is going to originate. The router compares the parameters in the message with its available resources and, if everything is okay, forwards the request downstream to the router one hop closer to Y. Eventually, when this chain of requests reaches Y, it responds with a reservation request (RESV message) towards host X.

It is up to the receiver to request specific QoS properties from the network (e.g. specify the bitrate and quality of service class). After that, every router in path checks if it can provide the requested QoS parameters, and forwards the requests upstream to the sender. When sender receives the reservation message, it knows that it can now send data along the QoS-capable path. Of course, each router on the path must pre-configure proper instrumentation to provide the requested QoS parameters. Note that since flows are unidirectional it requires installing two reservations for each direction of the traffic if the packet exchange is duplex.

Within the Integrated Services model, the network may provide the following classes of service: Best Effort, Guaranteed Rate, and Controlled Load. Best effort flows do not require any specific QoS treatment. Guaranteed rate provides a flow with bandwidth and delay guarantee – this is the strictest type of service. Controlled load is the type of service you may expects on a lightly loaded network – it provides you the bandwidth but no guarantees on delay.

Copyright © 2010 Internetwork Expert

The RSVP reservation request contains two structures called ‘Flowspec’ and ‘Filterspec’. These two structures describe the reservation that receiver requests from the network. The ‘Flowspec’ consists of ‘Rspec’ and ‘Tspec’ structures. ‘Rspec’ (reservation specification) defines the class of services that reservation demands and ‘Tspec’ (traffic specification) defines token bucket parameter for traffic metering – such as average rate and burst size. Note that those structures are the part of the requests sent from the receiver upstream to the sender. RSVP path requests (originated by sender) also contain ‘Tspec’ structure, so that routers on the way may check their resources to see if they can accommodate the new flow’s bitrate.

The ‘Filterspec’ structure defines the senders filter. Essentially, it specifies the sources that are eligible to use the reservation installed by the receiver. Usually, communications are one to one. However, sometimes you may want to send RESV message to multiple senders and make them share a single reservation. For example, this may be the case of many-to-many conference. Three types of filter exist in RSVP: *Fixed Filter* (FF), *Shared Explicit* (SE), and *Wildcard Filter* (WF). FF allows only one explicitly specified source to use the reservation, and the ‘Tspec’ structure parameters (rate, burst etc) apply only to that single flow. If there are multiple senders, the receiver needs to establish a reservation for every one of them. SE allows multiple explicitly specified sources to use the same reservation. The receiver specifies the IP addresses of the sources in the reservation message. WF allows any sender to use the reservation, without explicitly specifying the IP addresses.

Note that RSVP is inherently poorly scalable, in the sense that number of flows grows as N^2 where N is the number of endpoints. However, if the receivers use multicast addressing, RSVP supports feature knows as “reservation merging”. That is, if many recipients sharing the same multicast group create reservations for the same sender, RVSP will merge all reservations into one with the largest bandwidth, and propagate it upstream the tree towards the sender. This feature makes RSVP scalable, when working with multicast. Other enhancements to RSVP also allow scalability by adding “aggregate” reservations; the standard RSVP implementation does not support them.

RSVP is a soft-state protocol – that is, a router would keep an RSVP reservation only when it keeps receiving RSVP PATH/RESV messages. By default, all routers send those messages asynchronously, every 30 seconds. That is, if a router has particular RSVP state, it would keep sending PATH and RESV messages for it. If it does not receive PATH from upstream or RESV from downstream node for some it, the router will time out the state.

Copyright © 2010 Internetwork Expert

RSVP configuration is simple, with only a single command needed to specify the RSVP bandwidth and the per-flow bandwidth using the syntax **`ip rsvp bandwidth <total-bw> <per-flow-bw>`**. Note that you need this command to enable RSVP as well as specify bandwidth available to RSVP flows. Without RSVP enabled on interface, a router will not accept nor originate RSVP messages on the interface. If you omit any parameters to the above-described command, the default is to use 75% of interface bandwidth for RSVP and allow the same amount per-flow. Note that you cannot set the RSVP bandwidth to a value higher than 75% of the interface configured bandwidth.

Also note that RSVP needs WFQ/CBWFQ as the interface queue to provide required QoS parameters. On interfaces that do not support WFQ/CBWFQ, RSVP performs admission control only. The system uses 'Filterspec' structure parameters to define traffic classifier for WFQ/CBWFQ flow. Look at this as an enhancement to automatic logic of WFQ flows classification. In fact, RSVP flows are very similar to CBWFQ user-defined classes. They even share the same flow numbers with user-defined classes (refer to the task on CBWFQ bandwidth reservation for details). So what is the formula for WFQ weigh of an RSVP flow? Here is the procedure for weigh calculation:

1) The RSVP flow with maximum bandwidth reservation has WFQ weight of 6.
2) All other flows, have weights based on the formula:

Weight = 6*max_reservation_bw/flow_reservation_bw

Where "max_reservation_bw" is the maximum RSVP bandwidth reservation on the link and "flow_reservation_bw" is the RSVP bandwidth reservation for the flow in question.

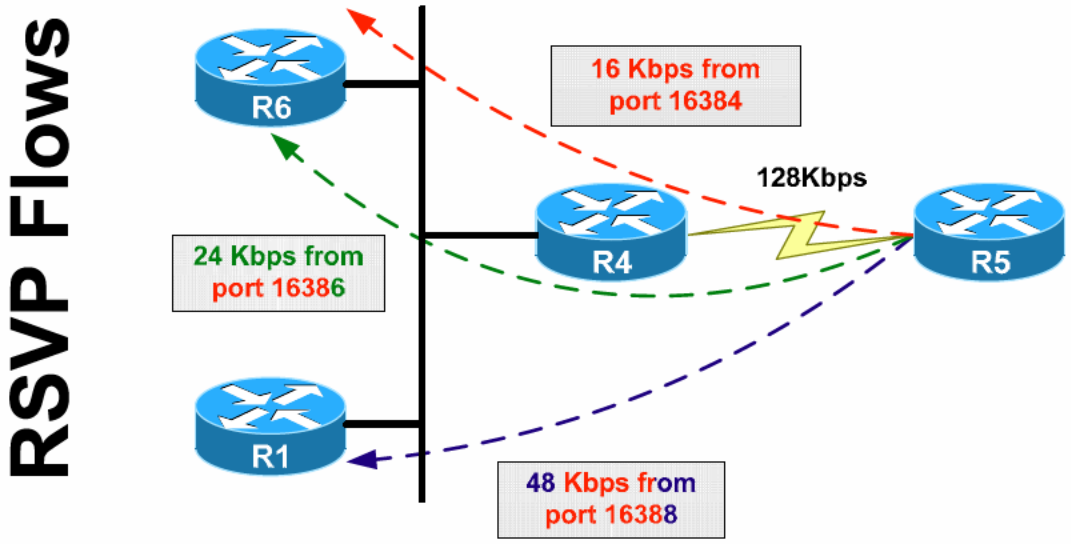
For example if there are two reservations of 12Kbps and 36Kbps, then the second flow will have a WFQ weight of 6, and the first flow will have a WFQ weight of 18.

We see that those weights are small. So how does WFQ ensure that RSVP flows do not monopolize the WFQ bandwidth? The answer is that RSVP also installs special per-flow token-bucket meters, configured in accordance to 'Tspec' parameters. The scheduler serves packets that *exceed* the configured parameters like if they have the default WFQ weight corresponding to IP Precedence of 0 (the highest weight of 32384, lowest scheduling priority).

Note that WFQ subtracts the bandwidth reserved by RSVP from the bandwidth available on the interface, so you may see the approximate amount of bandwidth remaining to other flows on the link.

Copyright © 2010 Internetwork Expert

To verify how RSVP works, consider the diagram below. R5 is the sender for 3 RSVP sessions going down to VLAN146 hosts. R6 reserves bandwidth for two sessions of 16Kbps and 24Kbps, while R1 reserves just one session of 48Kbps. Both sessions going to R6 will ask for guaranteed rate and R1 will ask for controlled load services



We source the first flow from UDP port 16384, second flow from UDP port of 16386 and the third flow from UDP port of 16388.

Note that in the configuration below burst sizes are in kilobytes. The burst size is taken as "Rate*1s" which is 6Kbyte for 48Kbps, 3Kbyte for 24Kbps, and 2Kbyte for 16Kbps. You will see the reason for selecting the averaging interval of 1s later, when we start configuring traffic sources.

```
R1:
ip rsvp reservation-host 150.1.1.1 150.1.5.5 UDP 16388 16388 FF LOAD 48 6

R5:
ip rsvp sender-host 150.1.1.1 150.1.5.5 UDP 16388 16388 48 6
ip rsvp sender-host 150.1.6.6 150.1.5.5 UDP 16384 16384 16 2
ip rsvp sender-host 150.1.6.6 150.1.5.5 UDP 16384 16386 24 3

R6:
ip rsvp reservation-host 150.1.6.6 150.1.5.5 UDP 16384 16384 FF RATE 16 2
ip rsvp reservation-host 150.1.6.6 150.1.5.5 UDP 16384 16386 FF RATE 24 3
```

Copyright © 2010 Internetwork Expert

Confirm that R5 has installed the RSVP reservations originated from R6 and R1 on the Serial interface to R4. Note the WFQ weights assigned to the flows as well as policing parameters for each flow. Also, look at the interface RSVP status, to see the amount of already allocated bandwidth.

```
Rack1R5#show ip rsvp installed
RSVP: Serial0/1/0
BPS     To              From            Protoc DPort  Sport  Weight Conversation
48K     150.1.1.1       150.1.5.5       UDP    16388  16388  6      265
16K     150.1.6.6       150.1.5.5       UDP    16384  16384  18     266
24K     150.1.6.6       150.1.5.5       UDP    16384  16386  12     267

Rack1R5#show ip rsvp interface detail

 Se0/1/0:
   Interface State: Up
   Bandwidth:
     Curr allocated: 88K bits/sec
     Max. allowed (total): 128K bits/sec
     Max. allowed (per flow): 64K bits/sec
     Max. allowed for LSP tunnels using sub-pools: 0 bits/sec
     Set aside by policy (total): 0 bits/sec
   Admission Control:
     Header Compression methods supported:
       rtp (36 bytes-saved), udp (20 bytes-saved)
   Traffic Control:
     RSVP Data Packet Classification is ON via CEF callbacks
   Signalling:
     DSCP value used in RSVP msgs: 0x3F
     Number of refresh intervals to enforce blockade state: 4
     Number of missed refresh messages: 4
     Refresh interval: 30
   Authentication: disabled
```

We have three WFQ flows, with conversation numbers 265, 266 and 267. That means that there are 256 dynamic WFQ flows on the interface. The weights are 6, 12, and 18 for flows that reserved 48K, 24K, and 16K respectively.

Now we configure three traffic sources, to simulate sender traffic. We will use the IP SLA jitter operation to generate bursts of 100-byte packets every 1 second. By sending 2000 bytes every 1 second we would achieve the 16Kbps average rate. Similarly, bursts of 4000 bytes and 6000 bytes result in average rates of 24Kbps and 48Kbps.

Note that IP SLA operation sends UDP packets. The combined size of IP + UDP header is 20 + 8 bytes. The HDLC overhead is 4 bytes. Thus, to get 100 bytes of layer 2 size, we need the payload size of 68 bytes (68 + 28 + 4=100). In addition, we set the inter-packet interval to the minimum 1ms, to generate packet bursts.

Copyright © 2010 Internetwork Expert

The three operations correspond to the three RSVP reservations:

```
R1:
ip sla monitor responder

R6:
ip sla responder

R5:
ip sla monitor 1
 type jitter dest-ipaddr 150.1.6.6 dest-port 16384 source-ipaddr 150.1.5.5
source-port 16384 num-packets 20 interval 1
 request-data-size 68
 timeout 500
 frequency 1
ip sla monitor schedule 1 life forever start-time now

ip sla monitor 2
 type jitter dest-ipaddr 150.1.6.6 dest-port 16384 source-ipaddr 150.1.5.5
source-port 16386 num-packets 30 interval 1
 request-data-size 68
 timeout 500
 frequency 1
ip sla monitor schedule 2 life forever start-time now

ip sla monitor 3
 type jitter dest-ipaddr 150.1.1.1 dest-port 16388 source-ipaddr 150.1.5.5
source-port 16388 num-packets 60 interval 1
 request-data-size 68
 timeout 500
 frequency 1
ip sla monitor schedule 3 life forever start-time now
```

Copyright © 2010 Internetwork Expert

Now create a special policy-map on R4 to meter the input rate for RSVP reserved flows. Start transferring the IOS image from R5 to R6 across the slow serial link to see that WFQ guarantees reservations even when with congested link.

```
R4:
ip access-list extended FLOW_1
 permit udp host 150.1.5.5 eq 16384 any
!
ip access-list extended FLOW_2
 permit udp host 150.1.5.5 eq 16386 any
!
ip access-list extended FLOW_3
 permit udp host 150.1.5.5 eq 16388 any
!
class-map match-all FLOW_1
 match access-group name FLOW_1
!
class-map match-all FLOW_3
 match access-group name FLOW_3
!
class-map match-all FLOW_2
 match access-group name FLOW_2
!
policy-map METER
 class FLOW_1
 class FLOW_2
 class FLOW_3
!
interface Serial0/1/0
 service-policy input METER
 load-interval 30

R5:
ip http server
ip htt path flash:
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#$copy http://admin:cisco@150.1.5.5/c2600-adventerprisek9-
mz.124-10.bin null:
Loading http://************@150.1.5.5/c2600-adventerprisek9-mz.124-
10.bin !!!

Rack1R4#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy input: METER

    Class-map: FLOW_1 (match-all)
      124181 packets, 11182928 bytes
      30 second offered rate 16000 bps
      Match: access-group name FLOW_1

    Class-map: FLOW_2 (match-all)
      151863 packets, 14924736 bytes
      30 second offered rate 24000 bps
      Match: access-group name FLOW_2

    Class-map: FLOW_3 (match-all)
      298729 packets, 29432584 bytes
      30 second offered rate 48000 bps
      Match: access-group name FLOW_3

    Class-map: class-default (match-any)
      1776 packets, 553343 bytes
      30 second offered rate 33000 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

See that the metered rate is the same that we reserved, even though the HTTP packets overload the link. Now look at the contents of the WFQ queue on R5:

```
Rack1R5#show queueing interface serial 0/1/0
Interface Serial0/1/0 queueing strategy: fair
  Input queue: 1/75/0/0 (size/max/drops/flushes); Total output drops:
15470
  Queueing strategy: weighted fair
  Output queue: 27/1000/64/15470 (size/max total/threshold/drops)
     Conversations  2/7/256 (active/max active/max total)
     Reserved Conversations 3/3 (allocated/max allocated)
     Available Bandwidth 8 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 20/6/0/0/0
  Conversation 265, linktype: ip, length: 100
  source: 150.1.5.5, destination: 150.1.1.1, id: 0x000D, ttl: 255,
  TOS: 0 prot: 17, source port 16388, destination port 16388

  (depth/weight/total drops/no-buffer drops/interleaves) 7/32384/0/0/0
  Conversation 70, linktype: ip, length: 580
  source: 150.1.5.5, destination: 155.1.146.6, id: 0xA68F, ttl: 255,
  TOS: 0 prot: 6, source port 80, destination port 51793
```

Note that RSVP flow for port 16388 has weigh of 6, since it's the maximum reservation flow on the interface. Also, note the remaining bandwidth and the number of dynamic flows on the interface. Now look at the detailed statistics for installed RSVP flows:

```
Rack1R5#show ip rsvp installed detail

RSVP: Serial0/1/0 has the following installed reservations
RSVP Reservation. Destination is 150.1.1.1. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16388, Source port is 16388
  Traffic Control ID handle: 0B00040E
  Created: 23:52:56 UTC Sat Mar 2 2002
  Admitted flowspec:
    Reserved bandwidth: 48K bits/sec, Maximum burst: 6K bytes, Peak rate: 48K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow:
    WFQ on hw idb Se0/1:  RESERVED queue 265.  Weight: 6, BW 48 kbps
  Conversation supports 1 reservations [0xD000400]
  Data given reserved service: 44760 packets (4296960 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 746 seconds
  Long-term average bitrate (bits/sec): 46061 reserved, 0 best-effort
  Policy: INSTALL. Policy source(s): Default
```

Copyright © 2010 Internetwork Expert

```
RSVP Reservation. Destination is 150.1.6.6. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16384, Source port is 16384
  Traffic Control ID handle: 15000409
  Created: 23:52:57 UTC Sat Mar 2 2002
  Admitted flowspec:
    Reserved bandwidth: 16K bits/sec, Maximum burst: 2K bytes, Peak rate: 16K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow:
    WFQ on hw idb Se0/1:  RESERVED queue 266.  Weight: 18, BW 16 kbps
  Conversation supports 1 reservations [0x36000402]
  Data given reserved service: 14960 packets (1436160 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 748 seconds
  Long-term average bitrate (bits/sec): 15351 reserved, 0 best-effort
  Policy: INSTALL. Policy source(s): Default

RSVP Reservation. Destination is 150.1.6.6. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16384, Source port is 16386
  Traffic Control ID handle: C100040F
  Created: 23:52:57 UTC Sat Mar 2 2002
  Admitted flowspec:
    Reserved bandwidth: 24K bits/sec, Maximum burst: 3K bytes, Peak rate: 24K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow:
    WFQ on hw idb Se0/1:  RESERVED queue 267.  Weight: 12, BW 24 kbps
  Conversation supports 1 reservations [0x6900040B]
  Data given reserved service: 22530 packets (2162880 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 750 seconds
  Long-term average bitrate (bits/sec): 23047 reserved, 0 best-effort
  Policy: INSTALL. Policy source(s): Default
```

Copyright © 2010 Internetwork Expert

Note the long-term average bit-rates for each flow. They are close to the values we set in the reservations: 16K, 24K, and 48K. Now check RSVP states in R4. Note that "Resource Provider" is set to "None", and thus the router actually does nothing to provide the required QoS treatment on the FastEthernet interface. However, commonly no one needs that, since high-speed interfaces usually provide more bandwidth than a router can forward, and the congestion points are WAN links.

```
Rack1R4#show ip rsvp installed detail

RSVP: FastEthernet0/1 has the following installed reservations
RSVP Reservation. Destination is 150.1.1.1. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16388, Source port is 16388
  Traffic Control ID handle: 0400040C
  Created: 18:13:28 UTC Fri Mar 1 2002
  Admitted flowspec:
    Reserved bandwidth: 48K bits/sec, Maximum burst: 6K bytes, Peak rate: 48K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow: None
  Conversation supports 1 reservations [0x36000406]
  Data given reserved service: 47481 packets (4558176 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 791 seconds
  Long-term average bitrate (bits/sec): 46073 reserved, 0 best-effort
  Policy: INSTALL. Policy source(s): Default
RSVP Reservation. Destination is 150.1.6.6. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16384, Source port is 16384
  Traffic Control ID handle: 0A000404
  Created: 18:13:13 UTC Fri Mar 1 2002
  Admitted flowspec:
    Reserved bandwidth: 16K bits/sec, Maximum burst: 2K bytes, Peak rate: 16K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow: None
  Conversation supports 1 reservations [0x800040A]
  Data given reserved service: 16161 packets (1551456 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 808 seconds
  Long-term average bitrate (bits/sec): 15359 reserved, 0 best-effort
  Policy: INSTALL. Policy source(s): Default
RSVP Reservation. Destination is 150.1.6.6. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16384, Source port is 16386
  Traffic Control ID handle: 03000403
  Created: 18:13:35 UTC Fri Mar 1 2002
  Admitted flowspec:
    Reserved bandwidth: 24K bits/sec, Maximum burst: 3K bytes, Peak rate: 24K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow: None
  Conversation supports 1 reservations [0x7000402]
  Data given reserved service: 23615 packets (2267040 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 787 seconds
  Long-term average bitrate (bits/sec): 23067 reserved, 0 best-effort
  Policy: INSTALL. Policy source(s): Default
```

Copyright © 2010 Internetwork Expert

Finally, observe the flow of RSVP messages on R4. Note the periodic refresh messages for all three flows and the default refresh interval value of 30 seconds.

```
Rack1R4#debug ip rsvp

RSVP 150.1.5.5_16386->150.1.6.6_16384[0.0.0.0]: Received Path message from
155.1.45.5 (on Serial0/1/0)
RSVP session 150.1.6.6_16384[0.0.0.0]: Received RESV for 150.1.6.6
(FastEthernet0/1) from 155.1.146.6
RSVP 150.1.5.5_16384->150.1.6.6_16384[0.0.0.0]: reservation found--processing
possible change: 848EA964
RSVP 150.1.5.5_16384->150.1.6.6_16384[0.0.0.0]: No change in reservation
RSVP 150.1.5.5_16384->150.1.6.6_16384[0.0.0.0]: Refresh RESV, req=84917B78,
refresh interval=30000mSec [cleanup timer is not awake]
RSVP 150.1.5.5_16384->150.1.6.6_16384[0.0.0.0]: Sending Resv message to
155.1.45.5
RSVP session 150.1.1.1_16388[0.0.0.0]: Received RESV for 150.1.1.1
(FastEthernet0/1) from 155.1.146.1
RSVP 150.1.5.5_16388->150.1.1.1_16388[0.0.0.0]: reservation found--processing
possible change: 848EA824
RSVP 150.1.5.5_16388->150.1.1.1_16388[0.0.0.0]: No change in reservation
RSVP 150.1.5.5_16386->150.1.6.6_16384[0.0.0.0]: Refresh Path psb = 8490C884
refresh interval = 30000mSec
RSVP 150.1.5.5_16386->150.1.6.6_16384[0.0.0.0]: Sending Path message to
155.1.146.6
RSVP session 150.1.6.6_16384[0.0.0.0]: Received RESV for 150.1.6.6
(FastEthernet0/1) from 155.1.146.6
RSVP 150.1.5.5_16386->150.1.6.6_16384[0.0.0.0]: reservation found--processing
possible change: 848EAAA4
RSVP 150.1.5.5_16386->150.1.6.6_16384[0.0.0.0]: No change in reservation
```

Copyright © 2010 Internetwork Expert

Finally, let's see how WFQ deals with flows that exceed their reservations. Re-configure the 16Kbps flow to send 32Kbps instead. All you need is double the number of packets send per second. In addition to that, ensure that the file transfer between R5 and R6 is still ongoing.

```
R5:
no ip sla 1
udp-jitter 150.1.6.6 16384 source-ip 150.1.5.5 source-port 16384 num-packets 40
interval 4
 request-data-size 68
 timeout 5000
 threshold 100
 frequency 15

ip sla monitor schedule 1 life forever start-time now


Rack1R6#$copy http://admin:cisco@150.1.5.5/c1841-adventerprisek9-mz.124-
24.T.bin null:
Loading http://***********@150.1.5.5/c1841-adventerprisek9-mz.124-24.T.bin !!!

Rack1R4#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy input: METER

    Class-map: FLOW_1 (match-all)
      1303558 packets, 128241332 bytes
      30 second offered rate 32000 bps
      Match: access-group name FLOW_1

    Class-map: FLOW_2 (match-all)
      1894879 packets, 188349552 bytes
      30 second offered rate 23000 bps
      Match: access-group name FLOW_2

    Class-map: FLOW_3 (match-all)
      3729529 packets, 371635916 bytes
      30 second offered rate 48000 bps
      Match: access-group name FLOW_3

    Class-map: class-default (match-any)
      104206 packets, 55282992 bytes
      30 second offered rate 19000 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

Now we see that FLOW_1 traffic rate is 32Kbps – just as we configured it. The explanation to that fact is that RSVP scheduler re-assigns the exceeding traffic to a flow queue with a very high weigh 32384. However, this weight is on par with the weight of the HTTP traffic flow. Thus, they share the bandwidth remaining on the link in equal proportions. The bandwidth remaining on the link is approximately 40Kbps, so it's enough for FLOW_1 to get up to 32Kbps.

Let's look at the WFQ queue contents on R5 once again:

```
Rack1R5#show queueing interface serial 0/1/0
Interface Serial0/1/0 queueing strategy: fair
  Input queue: 2/75/0/0 (size/max/drops/flushes); Total output drops:
473
  Queueing strategy: Class-based queueing
  Output queue: 17/1000/64/473 (size/max total/threshold/drops)
     Conversations  2/10/256 (active/max active/max total)
     Reserved Conversations 3/3 (allocated/max allocated)
     Available Bandwidth 8 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 10/32384/0/0/0
  Conversation 213, linktype: ip, length: 100
  source: 150.1.5.5, destination: 150.1.6.6, id: 0x001F, ttl: 255,
  TOS: 0 prot: 17, source port 16384, destination port 16384

  (depth/weight/total drops/no-buffer drops/interleaves) 6/32384/0/0/0
  Conversation 223, linktype: ip, length: 580
  source: 150.1.5.5, destination: 155.1.146.6, id: 0x0A5C, ttl: 255,
  TOS: 0 prot: 6, source port 80, destination port 54751
```

We see that the snapshot of the queue contents containing flows for "exceeding" part of the violating flow and the flow for HTTP traffic. They both have the same weight of 32384. Now look at the statistics for the RSVP flow and note the exceeding traffic that uses best-effort services:

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip rsvp installed detail

RSVP: Serial0/1/0 has the following installed reservations
<snip>

RSVP Reservation. Destination is 150.1.6.6. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16384, Source port is 16384
  Traffic Control ID handle: 0500040E
  Created: 00:45:33 UTC Fri Mar 1 2002
  Admitted flowspec:
    Reserved bandwidth: 16K bits/sec, Maximum burst: 2K bytes, Peak
rate: 16K bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow:
    WFQ on hw idb Se0/1:  RESERVED queue 266.  Weight: 18, BW 16 kbps
  Conversation supports 1 reservations [0x600040B]
  Data given reserved service: 17449 packets (1659816 bytes)
  Data given best-effort service: 12071 packets (1156100 bytes)
  Reserved traffic classified for 847 seconds
  Long-term average bitrate (bits/sec): 15677 reserved, 10919 best-
effort
  Policy: INSTALL. Policy source(s): Default

<snip>
```

Copyright © 2010 Internetwork Expert

Note that metering is high from being accurate: RSVP considers just 11Kbps as exceeding, instead of 16Kbps. The reason is that we send traffic in bursts of 4Kbyte while we configured the RSVP policer for 2Kbyte bursts. This results in policer metering the rate below its actual value. Adjust the reservation burst sizes and see what happens:

```
R5:
no ip rsvp sender-host 150.1.6.6 150.1.5.5 UDP 16384 16384 16 2
ip rsvp sender-host 150.1.6.6 150.1.5.5 UDP 16384 16384 16 4

R6:
no ip rsvp reservation-host 150.1.6.6 150.1.5.5 UDP 16384 16384 FF RATE 16 2
ip rsvp reservation-host 150.1.6.6 150.1.5.5 UDP 16384 16384 FF RATE 16 4

Rack1R5#show ip rsvp installed detail

RSVP: Serial0/1/0 has the following installed reservations
<snip>

RSVP Reservation. Destination is 150.1.6.6. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16384, Source port is 16384
  Traffic Control ID handle: F200040D
  Created: 01:06:10 UTC Fri Mar 1 2002
  Admitted flowspec:
    Reserved bandwidth: 16K bits/sec, Maximum burst: 4K bytes, Peak rate: 16K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow:
    WFQ on hw idb Se0/1:  RESERVED queue 266.  Weight: 18, BW 16 kbps
  Conversation supports 1 reservations [0x700040E]
  Data given reserved service: 584 packets (56064 bytes)
  Data given best-effort service: 456 packets (43776 bytes)
  Reserved traffic classified for 25 seconds
  Long-term average bitrate (bits/sec): 17407 reserved, 13591 best-effort
  Policy: INSTALL. Policy source(s): Default

<snip>
```

Metering is much more accurate now, when the policer bursts size equals to traffic bursts.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.58 RSVP and SBM

- Configure R1 as the DSBM candidate to coordinate the bandwidth usage on the segment shared by R1, R4, and R6.
- Allocate just 1024Kbps to RSVP flows on this segment
- Configure R1 to inform other routers on the segment that they may not send above 50Kbyte/sec of non-RSVP traffic into the segment. Limit the burst size to 5Kbyte.

Configuration

```
R1:
interface FastEthernet 0/0
 ip rsvp dsbm candidate
 ip rsvp bandwidth 1024
 ip rsvp dsbm non-resv-send-limit rate 50
 ip rsvp dsbm non-resv-send-limit burst 5
```

Verification

Note

RSVP works nicely when you need to reserve bandwidth on the link between two routers. However, on shared segments like Ethernet, one router is not aware of reservations made through another router. This may lead to oversubscription, where the total amount of reserved bandwidth on all routers exceeds the capacity of the shared network. The problem described is only important for topologies that use shared media, such as half-duplex Ethernet or Token-Ring. However, one may imagine a similar situation in a ring-like Ethernet topology built using switches.

To resolve the above problem, one may introduce a special centralized bandwidth management agent on the segment. This agent is RSVP Subnetwork Bandwidth Manager and we call the segment with DSBM (designated SBM) as “managed”. DSBM is one of the many candidate routers on the segment, elected as the SBM. All other RSVP-capable routers on the segment will discover the DSBM and will use it as a “proxy” for RSVP PATH and RESV messages. This way the DSBM will know about all reservations and may perform centralized admission control. The traffic flows will take the direct path of course, and RSVP will install proper bandwidth reservations, if allowed by DSBM.

Copyright © 2010 Internetwork Expert

There is one another set of parameters, distributed by DSBM to all routers on the segment. It is the amount of non-reservable traffic allowed in the segment. The DSBM specifies average/peak rate for non-RSVP traffic as well as other parameters such as burst size. It's up to the routers on the segment to provide proper treatment and implement this policy. The DSBM's main function is performing admission control, not to implement any data plane policing.

So far, since you may rarely see Ethernet segments using WFQ (though properly configured CBWFQ is a viable queuing solution), RSVP/DSBM performs mainly an admission control function on Ethernet segments. In addition, Cisco IOS does not seem to implement any policing for non-RSVP traffic flowing into the shared segment.

You may configure multiple SBM candidates on a segment, and the one with numerically highest priority wins the election and becomes Designated SBM or DSBM. Note that if a DSBM already exist on a segment, newly configured SBM will not try to preempt it. If two candidate SMBs have the same priority, the one with highest IP address wins.

To verify your configuration, issue the following show commands:

```
Rack1R1#show ip rsvp sbm detail

Interface: FastEthernet0/0
Local Configuration              Current DSBM
  IP Address: 155.1.146.1          IP Address: 155.1.146.1
  DSBM candidate: yes              I Am DSBM: yes
  Priority: 64                     Priority: 64
  Non Resv Send Limit              Non Resv Send Limit
    Rate: 50 Kbytes/sec              Rate: 50 Kbytes/sec
    Burst: 5 Kbytes                  Burst: 5 Kbytes
    Peak: unlimited                  Peak: unlimited
    Min Unit: unlimited              Min Unit: unlimited
    Max Unit: unlimited              Max Unit: unlimited

Rack1R4#show ip rsvp sbm detail

Interface: FastEthernet0/1
Local Configuration              Current DSBM
  IP Address: 155.1.146.4          IP Address: 155.1.146.1
  DSBM candidate: no               I Am DSBM: no
  Priority: 0                      Priority: 64
  Non Resv Send Limit              Non Resv Send Limit
    Rate: unlimited                  Rate: 50 Kbytes/sec
    Burst: unlimited                 Burst: 5 Kbytes
    Peak: unlimited                  Peak: unlimited
    Min Unit: unlimited              Min Unit: unlimited
    Max Unit: unlimited              Max Unit: unlimited
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip rsvp sbm detail

Interface: FastEthernet0/0.146
Local Configuration              Current DSBM
  IP Address: 155.1.146.6          IP Address: 155.1.146.1
  DSBM candidate: no               I Am DSBM: no
  Priority: 64                     Priority: 64
  Non Resv Send Limit              Non Resv Send Limit
    Rate: unlimited                  Rate: 50 Kbytes/sec
    Burst: unlimited                 Burst: 5 Kbytes
    Peak: unlimited                  Peak: unlimited
    Min Unit: unlimited              Min Unit: unlimited
    Max Unit: unlimited              Max Unit: unlimited
```

Verify that R1 is now in the path of all RSVP messages on the shared segment. Note that the reservation originated at R1 itself is not take in account by DSBM:

```
Rack1R1#show ip rsvp installed
RSVP: FastEthernet0/0
BPS     To              From            Protoc DPort  Sport
16K     150.1.6.6       150.1.5.5       UDP    16384  16384
24K     150.1.6.6       150.1.5.5       UDP    16384  16386

Rack1R1#show ip rsvp interface detail fastEthernet 0/0

 Fa0/0:
   Interface State: Up
   Bandwidth:
     Curr allocated: 40K bits/sec
     Max. allowed (total): 1024K bits/sec
     Max. allowed (per flow): 1024K bits/sec
     Max. allowed for LSP tunnels using sub-pools: 0 bits/sec
     Set aside by policy (total): 0 bits/sec
   Admission Control:
     Header Compression methods supported:
       rtp (36 bytes-saved), udp (20 bytes-saved)
   Traffic Control:
     RSVP Data Packet Classification is ON via CEF callbacks
   Signalling:
     DSCP value used in RSVP msgs: 0x3F
     Number of refresh intervals to enforce blockade state: 4
     Number of missed refresh messages: 4
     Refresh interval: 30
   Authentication: disabled
```

Copyright © 2010 Internetwork Expert

10.59 RSVP and CBWFQ

- Configure CBWFQ on the Serial link of R5 to ensure that HTTP server replies have a guarantee of all interface bandwidth not reserved to RSVP flows.
- Allow for 64 dynamic flow queues with CBWFQ.

Configuration

```
R5:
ip access-list extended HTTP
 permit tcp any eq www any
!
class-map HTTP
 match access-group name HTTP
!
policy-map CBWFQ
 class HTTP
  bandwidth remaining percent 100
 class class-default
  fair-queue 64
!
interface Serial 0/1/0
 no fair-queue
 service-policy output CBWFQ
```

Verification

Note

As previously discussed, RSVP treats all flow traffic exceeding its reservation as traffic with an IP precedence of zero. However, that means it may directly compete with all other best effort traffic. In this case, we want to ensure that HTTP traffic may claim all bandwidth not guaranteed to RSVP flows. To accomplish that, we set the bandwidth reservation for HTTP class to 100%. Simply speaking, we reserve all bandwidth not need by higher priority flows to the HTTP class. Technically speaking, CBWFQ implements this by assigning a weight of 64 to the HTTP class, as outlined below.

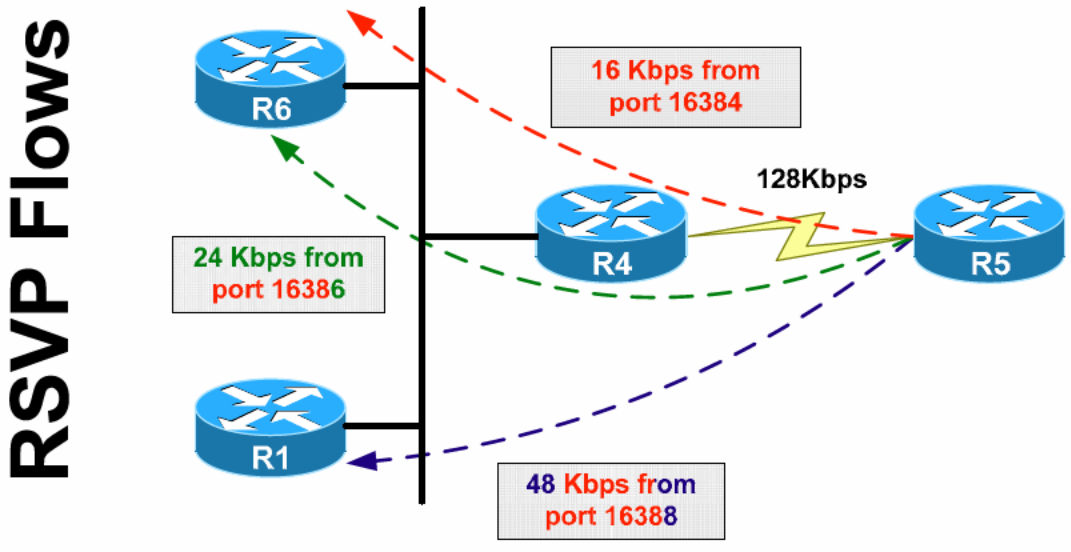
Copyright © 2010 Internetwork Expert

First, we know that CBWFQ is the same as WFQ from the perspective of RSVP. Thus RSVP flows look much like CBWFQ user-defined classes with relatively low weights. However, RSVP flows that exceed their reservation parameters have their treatment changed to "best-effort". This is the key difference between handling of RSVP flows and CBWFQ user-defined class. A user-defined class with bandwidth reservation may grow up to the full interface bandwidth, if other classes permit. CBWFQ never treats it as best-effort traffic with the lowest priority weight. Thus, exceeding traffic from RVSP flows always "loses" against traffic from a user-defined class, since user-defined traffic always have the same weight.

Secondly, if we have a traffic flow matching the RSVP classifier and a user defined class, the RSVP classifier will take precedence. Even though you may see matches in the policy map, the RSVP reservation will provide QoS services.

Let's run a simple simulation to see how CBWFQ divides bandwidth between RSVP and user-defined classes. You may want to refer back to task on CBWFQ bandwidth reservation to recall how the system assigns weights to user-defined classes.

Consider the diagram below. R5 is the sender for three RSVP sessions going down to VLAN146 hosts. R6 reserves bandwidth for two sessions of 16Kbps and 24Kbps, while R1 reserves just one session of 48Kbps. Both sessions going to R6 will ask for guaranteed rate and R1 will ask for controlled load services.



Copyright © 2010 Internetwork Expert

We source the first flow from UDP port 16384, second flow from UDP port of 16386, and the third flow from UDP port of 16388. Note that in the configuration below burst sizes are in kilobytes. The burst size is taken as "Rate*1s" which is 6Kbyte for 48Kbps, 3Kbyte for 24Kbps, and 2Kbyte for 16Kbps.

```
R1:
ip rsvp reservation-host 150.1.1.1 150.1.5.5 UDP 16388 16388 FF LOAD 48 6

R5:
ip rsvp sender-host 150.1.1.1 150.1.5.5 UDP 16388 16388 48 6
ip rsvp sender-host 150.1.6.6 150.1.5.5 UDP 16384 16384 16 2
ip rsvp sender-host 150.1.6.6 150.1.5.5 UDP 16384 16386 24 3

R6:
ip rsvp reservation-host 150.1.6.6 150.1.5.5 UDP 16384 16384 FF RATE 16 2
ip rsvp reservation-host 150.1.6.6 150.1.5.5 UDP 16384 16386 FF RATE 24 3
```

Now configure R5 to sends IP SLA jitter probes. See the task on RSVP/WFQ for more explanation on the burst sizes and number of packets in burst. So far, just remember that three SLA probes send packets in bursts separated by 1-second intervals and have average rates of 16K, 24K, and 48K. Note that we disabled the "control" connection, since it may be jammed on the serial link by bulky HTTP file transfer we are going to start between R5 and R6 as well.

```
R1:
ip sla monitor responder

R6:
ip sla responder

R5:
no ip sla monitor 1
no ip sla monitor 2
no ip sla monitor 3

ip sla 1
 udp-jitter 150.29.6.6 16384 source-ip 150.29.5.5 source-port 16384 num-packets
20 interval 4
 request-data-size 68
 timeout 5000
 threshold 100
 frequency 15
ip sla schedule 1 life forever start-time now

ip sla 2
 udp-jitter 150.29.6.6 16384 source-ip 150.29.5.5 source-port 16384 num-packets
20 interval 5
 request-data-size 68
 timeout 5000
 threshold 100
 frequency 15
ip sla schedule 2 life forever start-time now

ip sla 3
```

Copyright © 2010 Internetwork Expert

```
 udp-jitter 150.29.1.1 16384 source-ip 150.29.5.5 source-port 16384 num-packets
20 interval 6
 request-data-size 68
 timeout 5000
 threshold 100
 frequency 15
ip sla schedule 3 life forever start-time now

R5:
ip http server
ip htt path flash:

Rack1R6#copy http://admin:cisco@150.1.5.5/c1841-adventerprisek9-mz.124-24.T.bin
null:
Loading http://***********@150.1.5.5/c1841-adventerprisek9-mz.124-24.T.bin !!!
```

Now configure a special policy map on R4 to meter incoming traffic and check the resulting statistics:

```
R4:
ip access-list extended FLOW_1
 permit udp host 150.1.5.5 eq 16384 any
!
ip access-list extended FLOW_2
 permit udp host 150.1.5.5 eq 16386 any
!
ip access-list extended FLOW_3
 permit udp host 150.1.5.5 eq 16388 any
!
class-map match-all FLOW_1
 match access-group name FLOW_1
!
class-map match-all FLOW_3
 match access-group name FLOW_3
!
class-map match-all FLOW_2
 match access-group name FLOW_2
!
policy-map METER
 class FLOW_1
 class FLOW_2
 class FLOW_3
!
interface Serial0/1/0
 service-policy input METER
 load-interval 30
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy input: METER

    Class-map: FLOW_1 (match-all)
      1415938 packets, 139293444 bytes
      30 second offered rate 16000 bps
      Match: access-group name FLOW_1

    Class-map: FLOW_2 (match-all)
      1978603 packets, 196699272 bytes
      30 second offered rate 24000 bps
      Match: access-group name FLOW_2

    Class-map: FLOW_3 (match-all)
      3896920 packets, 388329656 bytes
      30 second offered rate 48000 bps
      Match: access-group name FLOW_3

    Class-map: class-default (match-any)
      115838 packets, 61721521 bytes
      30 second offered rate 35000 bps, drop rate 0 bps
      Match: any
```

The above seems to be in accordance with the configuration – all flows get their reserved rates and HTTP gets all the remaining bandwidth which is close to 40Kbps (128-16-24-48), but is definitely less due to various overhead and layer 1 procedures such as HDCL bit-stuffing.

Now let's configure one flow to send above its reserved rate, and see how CBWFQ shares bandwidth in this case. We are going to increase the burst size per second for one flow and change the reservations to reflect that burst size adjustment:

```
R5:
no ip sla 1

ip sla 1
 udp-jitter 150.1.6.6 16384 source-ip 150.1.5.5 source-port 16384 num-packets
20 interval 4
 request-data-size 68
 timeout 5000
 threshold 100
 frequency 15
ip sla schedule 1 life forever start-time now

ip sla 2
 udp-jitter 150.1.6.6 16384 source-ip 150.1.5.5 source-port 16384 num-packets
20 interval 5
 request-data-size 68
 timeout 5000
 threshold 100
 frequency 15
ip sla schedule 2 life forever start-time now
```

Copyright © 2010 Internetwork Expert

```
ip sla 3
 udp-jitter 150.1.1.1 16384 source-ip 150.1.5.5 source-port 16384 num-packets
20 interval 6
 request-data-size 68
 timeout 5000
 threshold 100
 frequency 15
ip sla schedule 3 life forever start-time now

R5:
no ip rsvp sender-host 150.1.6.6 150.1.5.5 UDP 16384 16384 16 2
ip rsvp sender-host 150.1.6.6 150.1.5.5 UDP 16384 16384 16 4

R6:
no ip rsvp reservation-host 150.1.6.6 150.1.5.5 UDP 16384 16384 FF RATE 16 2
ip rsvp reservation-host 150.1.6.6 150.1.5.5 UDP 16384 16384 FF RATE 16 4
```

Copyright © 2010 Internetwork Expert

Now look at the policy-map statistics on R4. Note that even though FLOW_1 now sends twice as much, it still gets 16Kbps from CBWFQ. This is because RSVP tells CBWFQ to treat exceeding traffic as best effort.

```
Rack1R4#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy input: METER

    Class-map: FLOW_1 (match-all)
      1473895 packets, 145089144 bytes
      30 second offered rate 16000 bps
      Match: access-group name FLOW_1

    Class-map: FLOW_2 (match-all)
      2047904 packets, 203629472 bytes
      30 second offered rate 24000 bps
      Match: access-group name FLOW_2

    Class-map: FLOW_3 (match-all)
      4006490 packets, 399286656 bytes
      30 second offered rate 46000 bps
      Match: access-group name FLOW_3

    Class-map: class-default (match-any)
      195068 packets, 107394026 bytes
      30 second offered rate 37000 bps, drop rate 0 bps
      Match: any
```

Look at the CBWFQ queue contents to see the flow weights:

```
Rack1R5#show queueing interface serial 0/1/0
Interface Serial0/1/0 queueing strategy: fair
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 292205
  Queueing strategy: Class-based queueing
  Output queue: 142/1000/64/292205 (size/max total/threshold/drops)
     Conversations  8/9/256 (active/max active/max total)
     Reserved Conversations 4/4 (allocated/max allocated)
     Available Bandwidth 8 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 3/12/0/0/0
  Conversation 267, linktype: ip, length: 100
  source: 150.1.5.5, destination: 150.1.6.6, id: 0x001B, ttl: 255,
  TOS: 0 prot: 17, source port 16386, destination port 16384

  (depth/weight/total drops/no-buffer drops/interleaves) 36/6/0/0/0
  Conversation 268, linktype: ip, length: 100
  source: 150.1.5.5, destination: 150.1.1.1, id: 0x0018, ttl: 255,
  TOS: 0 prot: 17, source port 16388, destination port 16388

  (depth/weight/total drops/no-buffer drops/interleaves) 22/18/0/0/0
  Conversation 266, linktype: ip, length: 100
  source: 150.1.5.5, destination: 150.1.6.6, id: 0x0012, ttl: 255,
  TOS: 0 prot: 17, source port 16384, destination port 16384
```

Copyright © 2010 Internetwork Expert

```
(depth/weight/total drops/no-buffer drops/interleaves) 7/16/0/0/0
Conversation 265, linktype: ip, length: 580
source: 150.1.5.5, destination: 155.1.146.6, id: 0x0D42, ttl: 255,
TOS: 0 prot: 6, source port 80, destination port 35205

(depth/weight/total drops/no-buffer drops/interleaves) 1/1024/0/0/0
Conversation 263, linktype: ip, length: 216
source: 155.1.45.5, destination: 224.0.0.9, id: 0x0000, ttl: 2,
TOS: 192 prot: 17, source port 520, destination port 520

(depth/weight/total drops/no-buffer drops/interleaves) 8/32384/0/0/0
Conversation 257, linktype: cdp, length: 328

(depth/weight/total drops/no-buffer drops/interleaves) 44/32384/15805/0/0
Conversation 213, linktype: ip, length: 100
source: 150.1.5.5, destination: 150.1.6.6, id: 0x000D, ttl: 255,
TOS: 0 prot: 17, source port 16384, destination port 16384

(depth/weight/total drops/no-buffer drops/interleaves) 20/32384/6485/0/0
Conversation 211, linktype: ip, length: 100
source: 150.1.5.5, destination: 150.1.1.1, id: 0x0008, ttl: 255,
TOS: 0 prot: 17, source port 16388, destination port 16388
```

The three highlighted flows are the conforming FLOW_1, HTTP traffic flow (which has CBWFQ weight maximum for user-defined classes, since it has 100% of bandwidth), and lastly the exceeding part of FLOW_1. Since HTTP has a better weight than the exceeding part of FLOW_1, it claims all the available bandwidth.

Copyright © 2010 Internetwork Expert

Now stop the HTTP file transfer, and check flow rates once again:

```
Rack1R4#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy input: METER

    Class-map: FLOW_1 (match-all)
      1495428 packets, 147242444 bytes
      30 second offered rate 32000 bps
      Match: access-group name FLOW_1

    Class-map: FLOW_2 (match-all)
      2077330 packets, 206572072 bytes
      30 second offered rate 24000 bps
      Match: access-group name FLOW_2

    Class-map: FLOW_3 (match-all)
      4065348 packets, 405172456 bytes
      30 second offered rate 48000 bps
      Match: access-group name FLOW_3

    Class-map: class-default (match-any)
      202138 packets, 111410986 bytes
      30 second offered rate 3000 bps, drop rate 0 bps
      Match: any
```

This time, FLOW_1 takes all the bandwidth it needs, even though it exceeds the reserved bandwidth.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.60 RSVP and LLQ

- Configure R5 to classify RSVP flows with a PQ-profile that ignores peak-rate, and classifies all flows below 2000 byte/s with the burst sizes below or equal to 2000 bytes as eligible for LLQ.

Configuration

```
R5:
ip rsvp pq-profile 2000 2000 ignore-peak-value
```

Verification

Note

The RSVP PQ profile defines parameters for RSVP flows that should use the CBWFQ priority queue (LLQ). The parameters are maximum rate in bytes per second, maximum burst size in bytes, and peak-to-average rate ratio in percents:

ip rsvp pq-profile <*max-rate*> <*max-burst*> <*peak-to-avg-ratio*>

If you configure **ip rsvp pq-profile voice-like** then the values are 12288 byte/s, burst 592 bytes, and 110%, which are the defaults.

RSVP if often used with VoIP applications to reserve resources for voice bearer flows. However, the WFQ services do not provide priority treatment for RSVP flows and thus do not guarantee low jitter values. This may result in lower voice quality, compared to the use of LLQ.

The RSVP PQ-profile enhancement allows the RSVP classifier to direct certain flows into the CBWFQ priority queue (LLQ). RSVP makes decisions based on the 'flowspec' structure of the reservation. If the parameters meet certain requirements specified in RSVP PQ-profile then the respective flow's traffic uses LLQ. However, unlike the LLQ policer, RSVP does not drop the exceeding packets, but rather marks them for best-effort services. Thus you don't need to configure any class in the CBWFQ policy with "priority" keyword – RSVP will classify flows automatically and will use their 'flowspecs' to define the policers.

Copyright © 2010 Internetwork Expert

When a router receives an RSVP reservation flowspec, it compares the parameters of the flow with the configured PQ-profile. If the parameters (avg. rate, peak rate, burst size) are less than or equal to the values specified in the global PQ profile, the RSVP scheduler considers the flow to be voice-like and directs it to LLQ for service.

To verify the feature, configure R5 to reserve bandwidth for three traffic flows, just as we did in previous tasks. Note that only the flow sourced off the port 16384 of R5 satisfies the PQ-profile requirements. After that, check the installed RSVP reservations on R5:

```
R1:
ip rsvp reservation-host 150.1.1.1 150.1.5.5 UDP 16388 16388 FF LOAD 48 6

R5:
ip rsvp sender-host 150.1.1.1 150.1.5.5 UDP 16388 16388 48 6
ip rsvp sender-host 150.1.6.6 150.1.5.5 UDP 16384 16384 16 2
ip rsvp sender-host 150.1.6.6 150.1.5.5 UDP 16384 16386 24 3

R6:
ip rsvp reservation-host 150.1.6.6 150.1.5.5 UDP 16384 16384 FF RATE 16 2
ip rsvp reservation-host 150.1.6.6 150.1.5.5 UDP 16384 16386 FF RATE 24 3

Rack1R5#show ip rsvp installed detail
<snip>

RSVP Reservation. Destination is 150.1.6.6. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16384, Source port is 16384
  Traffic Control ID handle: 0900040B
  Created: 02:09:17 UTC Fri Mar 1 2002
  Admitted flowspec:
    Reserved bandwidth: 16K bits/sec, Maximum burst: 2K bytes, Peak rate: 16K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow:
    WFQ on hw idb Se0/1/0:  PRIORITY queue 72.  Weight: 0, BW 16 kbps
  Conversation supports 1 reservations [0x1800040C]
<snip>
```

We see that the flow uses the priority queue inside CBWFQ. Now configure R5 to sends IP SLA jitter probes. See the task on RSVP/WFQ for more explanation on the burst sizes and number of packets in burst. So far, just remember that three SLA probes send packets in bursts separated by 1-second intervals and have average rates of 32K, 24K, and 48K. Note that the first source (source port 16384), which corresponds to voice-like RSVP reservation, is deliberately set to source twice as much packet as the reservation expects.

Copyright © 2010 Internetwork Expert

Also note that we disabled the "control" connection, since it may be jammed on the serial link by the bulky HTTP file transfer we are going to start between R5 and R6 as well.

```
R1:
ip sla monitor responder

R6:
ip sla responder

R5:
no ip sla monitor 1
no ip sla monitor 2
no ip sla monitor 3

ip sla 1
 udp-jitter 150.1.6.6 16384 source-ip 150.1.5.5 source-port 16384 num-packets
20 interval 4
 request-data-size 68
 timeout 5000
 threshold 100
 frequency 15
ip sla schedule 1 life forever start-time now

ip sla 2
 udp-jitter 150.1.6.6 16384 source-ip 150.1.5.5 source-port 16384 num-packets
20 interval 5
 request-data-size 68
 timeout 5000
 threshold 100
 frequency 15
ip sla schedule 2 life forever start-time now

ip sla 3
 udp-jitter 150.1.1.1 16384 source-ip 150.1.5.5 source-port 16384 num-packets
20 interval 6
 request-data-size 68
 timeout 5000
 threshold 100
 frequency 15
ip sla schedule 3 life forever start-time now

R5:
ip http server
ip htt path flash:

Rack1R6#copy http://admin:cisco@150.1.5.5/c1841-adventerprisek9-mz.124-24.T.bin
null:
Loading http://***********@150.1.5.5/c1841-adventerprisek9-mz.124-24.T.bin !!!
```

Now configure a special policy map on R4 to meter incoming traffic and check the resulting statistics:

Copyright © 2010 Internetwork Expert

```
R4:
ip access-list extended FLOW_1
 permit udp host 150.1.5.5 eq 16384 any
!
ip access-list extended FLOW_2
 permit udp host 150.1.5.5 eq 16386 any
!
ip access-list extended FLOW_3
 permit udp host 150.1.5.5 eq 16388 any
!
class-map match-all FLOW_1
 match access-group name FLOW_1
!
class-map match-all FLOW_3
 match access-group name FLOW_3
!
class-map match-all FLOW_2
 match access-group name FLOW_2
!
policy-map METER
 class FLOW_1
 class FLOW_2
 class FLOW_3
!
interface Serial0/1/0
 service-policy input METER
 load-interval 30

Rack1R4#show policy-map interface serial 0/1/0
 Serial0/1/0

  Service-policy input: METER

    Class-map: FLOW_1 (match-all)
      1636262 packets, 161325844 bytes
      30 second offered rate 16000 bps
      Match: access-group name FLOW_1

    Class-map: FLOW_2 (match-all)
      2231819 packets, 222020972 bytes
      30 second offered rate 24000 bps
     Match: access-group name FLOW_2

    Class-map: FLOW_3 (match-all)
      4373557 packets, 435993356 bytes
      30 second offered rate 48000 bps
      Match: access-group name FLOW_3

    Class-map: class-default (match-any)
      228882 packets, 126489279 bytes
      30 second offered rate 34000 bps, drop rate 0 bps
      Match: any
```

Even though FLOW_1 uses priority queue on R5, it does not exceed the 16Kbps limit. This is, of course, is thanks to the RSVP policer. Now let's check the CBWFQ queue contents on R5 to confirm that the "voice-like" flow uses LLQ instead of a regular conversation:

```
Rack1R5#show queueing interface serial 0/1/0
Interface Serial0/1/0 queueing strategy: fair
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 374535
  Queueing strategy: Class-based queueing
  Output queue: 105/1000/64/374535 (size/max total/threshold/drops)
     Conversations  5/7/64 (active/max active/max total)
     Reserved Conversations 3/3 (allocated/max allocated)
     Available Bandwidth 8 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 12/0/0/0/0
  Conversation 72, linktype: ip, length: 100
  source: 150.1.5.5, destination: 150.1.6.6, id: 0x0009, ttl: 255,
  TOS: 0 prot: 17, source port 16384, destination port 16384

  (depth/weight/total drops/no-buffer drops/interleaves) 20/12/0/0/0
  Conversation 75, linktype: ip, length: 100
  source: 150.1.5.5, destination: 150.1.6.6, id: 0x0000, ttl: 255,
  TOS: 0 prot: 17, source port 16386, destination port 16384

  (depth/weight/total drops/no-buffer drops/interleaves) 7/57/0/0/0
  Conversation 73, linktype: ip, length: 580
  source: 150.1.5.5, destination: 155.1.146.6, id: 0x0FDE, ttl: 255,
  TOS: 0 prot: 6, source port 80, destination port 11251

  (depth/weight/total drops/no-buffer drops/interleaves) 64/32384/29363/0/0
  Conversation 21, linktype: ip, length: 100
  source: 150.1.5.5, destination: 150.1.6.6, id: 0x001D, ttl: 255,
  TOS: 0 prot: 17, source port 16384, destination port 16384

  (depth/weight/total drops/no-buffer drops/interleaves) 2/32384/0/0/0
  Conversation 65, linktype: cdp, length: 328
```

From this output we see that some packets of the flow use the priority queue (weight 0) while the exceeding packets have a WFQ weight of 32384 and have the lowest scheduling priority.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.61 RSVP and Per-VC WFQ

- Configure the link between R4 and R5 for Frame-Relay encapsulation using DLCI 100.
- Enable legacy Frame-Relay traffic shaping on R5 and shape DLCI 100 down to 96Kbps, with the burst size corresponding to time-interval of 10ms.
- Allow RSVP to use up to 96Kbps of interface bandwidth with 64Kbps per-flow.
- Enable WFQ as the VC queue and configure per-VC WFQ as the RSVP resource provider.

Configuration

```
R4:
interface Serial0/1/0
 bandwidth 128
 encapsulation frame-relay
 load-interval 30
 no keepalive
 fair-queue 64 256 3
 frame-relay map ip 155.1.45.5 100 broadcast
 ip rsvp bandwidth 96 64

R5:
!
! Map class contains shaping parameters
! and defines WFQ as per-VC queue
!
map-class frame-relay DLCI_100
 frame-relay cir 96000
 frame-relay bc 9600
 frame-relay be 0
 frame-relay mincir 96000
 frame-relay fair-queue
!
interface Serial0/1/0
 bandwidth 128
 encapsulation frame-relay
 no keepalive
 no fair-queue
 clock rate 128000
 frame-relay traffic-shaping
 frame-relay map ip 155.1.45.4 100 broadcast
 frame-relay interface-dlci 100
  class DLCI_100
 ip rsvp bandwidth 96 64
 ip rsvp resource-provider wfq pvc
```

Copyright © 2010 Internetwork Expert

Verification

Note

RSVP uses a "resource provider" to implement the QoS policy for flows. The Resource Provider is either a per-interface WFQ or per-VC WFQ, when working with technologies like Frame-Relay. In order to engage per-VC WFQ or CBWFQ you need to configure legacy Frame-Relay traffic shaping (RSVP does not work any other form of traffic shaping) and enable WFQ as per-VC queue. Additionally, you need to set the resource provider to Per-VC WFQ, using the special interface-level command.

Remember that RSVP bandwidth configuration takes place an interface level. Thus, if you want to use a per-VC RSVP bandwidth admission granularity, create a separate sub-interface for each VC. If you have a hub site terminating many spoke VCs that all use RSVP, then consider using the lowest PVC CIR to deduce the RSVP bandwidth value. Otherwise some RSVP flows may reserve more bandwidth than low-speed PVCs may supply. (This process is similar to the bandwidth calculations you do when configuring EIGRP bandwidth pacing).

To verify your configuration, configure three bandwidth reservations as you did in the previous tasks. Keep in mind that we still have the PQ profile configured, and this will affect flow weights:

```
R1:
ip rsvp reservation-host 150.1.1.1 150.1.5.5 UDP 16388 16388 FF LOAD 48 6

R5:
ip rsvp sender-host 150.1.1.1 150.1.5.5 UDP 16388 16388 48 6
ip rsvp sender-host 150.1.6.6 150.1.5.5 UDP 16384 16384 16 2
ip rsvp sender-host 150.1.6.6 150.1.5.5 UDP 16384 16386 24 3

R6:
ip rsvp reservation-host 150.1.6.6 150.1.5.5 UDP 16384 16384 FF RATE 16 2
ip rsvp reservation-host 150.1.6.6 150.1.5.5 UDP 16384 16386 FF RATE 24 3
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip rsvp installed detail

RSVP: Serial0/1/0 has the following installed reservations
RSVP Reservation. Destination is 150.1.1.1. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16388, Source port is 16388
  Traffic Control ID handle: 4500040D
  Created: 03:06:15 UTC Fri Mar 1 2002
  Admitted flowspec:
    Reserved bandwidth: 48K bits/sec, Maximum burst: 6K bytes, Peak rate: 48K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow:
    WFQ on FR PVC dlci 100 on Se0/1:  RESERVED queue 25.  Weight: 6, BW 48 kbps
  Conversation supports 1 reservations [0xE700040A]
  Data given reserved service: 67920 packets (6520320 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 1131 seconds
  Long-term average bitrate (bits/sec): 46081 reserved, 0 best-effort
  Policy: INSTALL. Policy source(s): Default

RSVP Reservation. Destination is 150.1.6.6. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16384, Source port is 16384
  Traffic Control ID handle: A700040F
  Created: 03:06:15 UTC Fri Mar 1 2002
  Admitted flowspec:
    Reserved bandwidth: 16K bits/sec, Maximum burst: 2K bytes, Peak rate: 16K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow:
    WFQ on FR PVC dlci 100 on Se0/1:  PRIORITY queue 24.  Weight: 0, BW 16 kbps
  Conversation supports 1 reservations [0x2200040C]
  Data given reserved service: 23646 packets (2270016 bytes)
  Data given best-effort service: 21714 packets (2084544 bytes)
  Reserved traffic classified for 1134 seconds
  Long-term average bitrate (bits/sec): 16012 reserved, 14703 best-effort
  Policy: INSTALL. Policy source(s): Default

RSVP Reservation. Destination is 150.1.6.6. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16384, Source port is 16386
  Traffic Control ID handle: 5C000409
  Created: 03:06:15 UTC Fri Mar 1 2002
  Admitted flowspec:
    Reserved bandwidth: 24K bits/sec, Maximum burst: 3K bytes, Peak rate: 24K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow:
    WFQ on FR PVC dlci 100 on Se0/1:  RESERVED queue 26.  Weight: 12, BW 24
kbps
  Conversation supports 1 reservations [0xB00040E]
  Data given reserved service: 34080 packets (3271680 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 1136 seconds
  Long-term average bitrate (bits/sec): 23034 reserved, 0 best-effort
  Policy: INSTALL. Policy source(s): Default
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show traffic-shape queue
Traffic queued in shaping queue on Serial0/1/0 dlci 100
  Queueing strategy: weighted fair
  Queueing Stats: 94/600/64/14893 (size/max total/threshold/drops)
     Conversations  3/6/16 (active/max active/max total)
     Reserved Conversations 2/0 (allocated/max allocated)
     Available Bandwidth 8 kilobits/se
```

We see that flows use WFQ conversations on the PVC corresponding to DLCI 100. The number of reserved conversation is two, since one flow uses priority queue. Note the reserved conversation numbers, which correspond to the number of WFQ flows equal to 16. The IOS picked up this number automatically, based on PVC CIR, when we configured **fair-queue** on the PVC.

You may replace the per-VC WFQ with CBWFQ, since RSVP also interoperates with the latter one. For example, configure the following:

```
R5:
ip access-list extended HTTP
 permit tcp any eq www any
!
policy-map CBWFQ
 class HTTP
  bandwidth remaining percent 100
 class class-default
  fair-queue 64
!
map-class frame-relay DLCI_100
 no fair-queue
 service-policy output CBWFQ
```

Verify the installed RSVP flows once again. Note the new conversation numbers, based on the new number of dynamic WFQ flows (64):

```
Rack1R5#show ip rsvp installed detail

RSVP: Serial0/1/0 has the following installed reservations
RSVP Reservation. Destination is 150.1.1.1. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16388, Source port is 16388
  Traffic Control ID handle: 4700040D
  Created: 03:31:20 UTC Fri Mar 1 2002
  Admitted flowspec:
    Reserved bandwidth: 48K bits/sec, Maximum burst: 6K bytes, Peak rate: 48K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow:
    WFQ on FR PVC dlci 100 on Se0/1/0:  RESERVED queue 74.  Weight: 6, BW 48
kbps
  Conversation supports 1 reservations [0x2300040C]
  Data given reserved service: 12960 packets (1244160 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 216 seconds
  Long-term average bitrate (bits/sec): 45987 reserved, 0 best-effort
  Policy: INSTALL. Policy source(s): Default
```

Copyright © 2010 Internetwork Expert

```
RSVP Reservation. Destination is 150.1.6.6. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16384, Source port is 16384
  Traffic Control ID handle: A900040F
  Created: 03:31:17 UTC Fri Mar 1 2002
  Admitted flowspec:
    Reserved bandwidth: 16K bits/sec, Maximum burst: 2K bytes, Peak rate: 16K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow:
    WFQ on FR PVC dlci 100 on Se0//01:  PRIORITY queue 72.  Weight: 0, BW 16
kbps
  Conversation supports 1 reservations [0xC00040E]
  Data given reserved service: 4626 packets (444096 bytes)
  Data given best-effort service: 4214 packets (404544 bytes)
  Reserved traffic classified for 220 seconds
  Long-term average bitrate (bits/sec): 16101 reserved, 14667 best-effort
  Policy: INSTALL. Policy source(s): Default

RSVP Reservation. Destination is 150.1.6.6. Source is 150.1.5.5,
  Protocol is UDP, Destination port is 16384, Source port is 16386
  Traffic Control ID handle: 34000401
  Created: 03:31:25 UTC Fri Mar 1 2002
  Admitted flowspec:
    Reserved bandwidth: 24K bits/sec, Maximum burst: 3K bytes, Peak rate: 24K
bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow:
    WFQ on FR PVC dlci 100 on Se0/1/0:  RESERVED queue 75.  Weight: 12, BW 24
kbps
  Conversation supports 1 reservations [0xE800040A]
  Data given reserved service: 6450 packets (619200 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 214 seconds
  Long-term average bitrate (bits/sec): 23082 reserved, 0 best-effort
  Policy: INSTALL. Policy source(s): Default
```

Now you see how RSVP may apply to per-VC queues when using WFQ or CBWFQ as the queuing discipline.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.62 Catalyst QoS Port-Based Classification

- Enable MLS QoS in all switches.
- Configure CoS to DSCP maps on SW1 and SW2 so that CoS 3 maps to DSCP 26 and CoS 5 maps to DSCP EF.
- Ensure CoS 4 maps to DSCP 44 on SW1.
- Configure IP Precedence to DSCP map in SW4 to map IP Precedence values 3 and 5 into DSCP 26 and DSCP EF.
- Trust DSCP values in packets coming from R1's interface connected to SW1.
- Trust IP Precedence on SW4 port connected to R4 and set the default CoS to 2.
- Override CoS to a value of four on the port connected to R5's VLAN58 interface.
- Use 802.1p bits in the Ethernet headers for classification of packets coming from R6; packets on native VLAN should use a CoS value of one.
- Use interface-level commands only to accomplish the task.

Configuration

```
SW1:
mls qos
!
! This map is indexed by CoS values. That is, for CoS 3 take the value
! with offset 3 in this list. Indexes start at zero and end at 7.
! AF31 is DSCP value of 26 and DSCP EFhas DSCP value of 46.
!
! Note that CoS 4 maps to DSCP 44
!
mls qos map cos-dscp 0 8 16 26 44 46 48 56
!
interface FastEthernet 0/1
 mls qos trust dscp
!
! Force CoS value of 4 on all packets. The switch will modify the DSCP
! value according to the CoS to DSCP table and set it to 44
!
interface FastEthernet 0/5
 mls qos cos 4
 mls qos cos override
```

Copyright © 2010 Internetwork Expert

```
SW2:
mls qos
!
mls qos map cos-dscp 0 8 16 26 32 46 48 56
!
! Trust CoS bits in 802.1p headers. For packets on native VLAN
! (e.g. CDP) apply the CoS value of 1, since they don't have 802.1p
! bits of their own
!
interface FastEthernet 0/6
 mls qos trust cos
 mls qos cos 1

SW3:
mls qos

SW4:
mls qos
!
! IP precedence to DSCP map is similar to CoS to DSCP
!
mls qos map ip-prec-dscp 0 8 16 26 32 46 48 56
!
interface FastEthernet 0/4
 mls qos trust ip-precedence
 mls qos cos 2
```

Verification

Note

Before you can configure any QoS settings in the 3550 or 3560 Catalyst series switches you need to issue the global **`mls qos`** command. Remember that with a clean QoS configuration the default is not to trust any marking, but reset it to zero. In this task, we implement the simplest form of classification based on port-level trusting.

Note the use of QoS tables to attain the desired markings. The trick is that you can infer DSCP/IP Precedence for untagged packets based on the default interface value, or using the CoS override functionality. The mapping tables maps the L2 to L3 mappings thus achieving the desired goal assigning the DSCP 44 marking.

Copyright © 2010 Internetwork Expert

The Catalyst QoS implementation uses the Differentiated Services model with the following design goals:

1) Edge network nodes classify input packets using locally configured policy and QoS marking found in packets.
2) Edge nodes encode traffic class using a special field in a packet to let the "core" nodes know of the packet class.
3) Core and edge nodes allocate resources and provide services based on the packet class.

A peculiar feature is that each node acts on its own, according to the configured policy. In order for the QoS policy to be consistent end-to-end, a network administrator must ensure that all nodes use the same classification and resource allocation policy. Another feature is that Differentiated Services model scales well, because only edge nodes classify traffic.

The following are options for encoding marking in IP/Ethernet networks:

1) ToS byte in IP packet or Traffic Class byte in IPv6 packet. The switch may interpret this field in two different ways:
1.1) Interpret the topmost six bits of the byte as DSCP code point. This is in full compliance with Diff-Serv model.
1.2) Interpret the topmost three bits of the byte as IP Precedence value. This complies with the old, "military-style" QoS model, where higher precedence traffic preempts the lower precedences. Note that the IP Precedence numbers form a subspace of DSCP numbers.

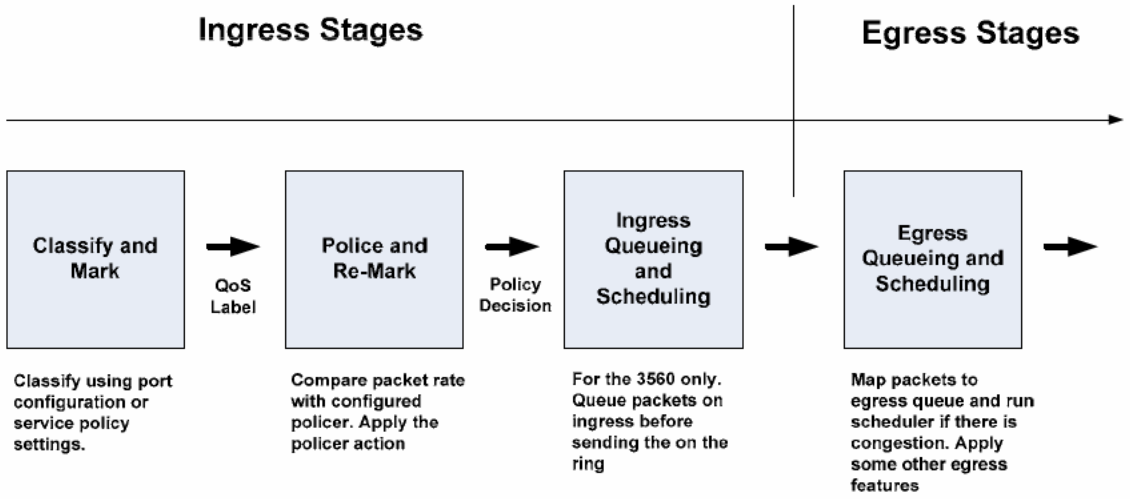
2) The CoS bits (three bits) in 802.1q/ISL header of a tagged Ethernet frame. Those bits are also known as 802.1p priority bits, and should be interpreted under the respective QoS model of relative traffic priorities.

The first two types of marking are Layer 3, and the CoS based marking is Layer 2. The Layer 2 marking is only possible on trunk links. The Catalyst switches have no special intelligence to implement any of the QoS models by themselves. It's up to you to define the policy and encode classes using any marking you find more flexible. Due to the large number of available markings, the Catalyst switches assign a special QoS label to packets and use this label for internal QoS processing. Both switches represent this label as an "internal DSCP" value, associated with a packet.

Copyright © 2010 Internetwork Expert

Keep in mind the distinction that the Catalyst Switches make between IP and non-IP packets. As we know, layer 3 switches are hardware optimized to route IP or IPv6 packets using their ASICs. This results in differences of handling the IP and non-IP packets. Specifically, you may use MAC-based ACLs (MAC addresses, Ether-types, LSAPs) only with non-IP packets (ARP, DECNet, IPX). The MAC ACLs will not match any of IP packets, even if you specify the correct MAC addresses. Also, note that the 3560 models treats IPv6 as "IP" traffic, while the 3550 treats the IPv6 packets as "non-IP" and matches them with MAC ACLs.

The following diagram displays the stage of QoS processing on a typical Catalyst switch.



We are now considering the **Classification and Marking** stage. The switch uses the local policy configuration to classify input packets. The local policy may be as simple as port QoS settings or as complex as policy-maps using QoS ACLs. The result of the classification stage is that the internal QoS label (e.g. internal DSCP value with the 3550) is created. At this stage, the switch extensively uses special globally configurable QoS mapping tables. The tables map one QoS marking to another, e.g. CoS values to DSCP, or DSCP to CoS. The switch uses those tables for "synchronization" of different types of QoS markings. For example, if you instruct the switch to set the CoS field to "3", the switch will look up the CoS-to-DSCP mapping table and set the DSCP value in the packet header respectively. In addition, the switch uses those tables to translate input marking (e.g. CoS, or IP Precedence) into the Internal DSCP label and vice versa: on egress, the switch maps internal DSCP value to CoS marking if needed.

Copyright © 2010 Internetwork Expert

You may classify using either interface level settings or by applying a pre-configured policy-map. Those two options are mutually exclusive, i.e. if you configure interface level classification settings and later apply a service-policy, then the latter removes any interface-level QoS classification settings (there are some exceptions though).

The following is the list of the classification options:

1) Trust one of the marking types already found in the packet (**`mls qos trust {dscp|ip-precedence|cos}`**). For IP packets, it is possible to trust either IP Precedence or DSCP value, and trusting Precedence or DSCP makes sense only for IP packets. If the switch trusts IP markings and the incoming packet is non-IP, the switch will attempt to classify based on the CoS value in the Ethernet header. If there is a CoS value in the packet (i.e. the port is trunk) the switch uses this value to build the QoS label. However, if there is not a CoS field in the packet, the switch will look at the default CoS value configured on the interface (**`mls qos cos x`**). The switch builds the corresponding DSCP value for the label using the CoS to DSCP table. When you trust IP precedence, the switch builds the DSCP value using the IP-Precedence to DSCP mapping table and the CoS value using the DSCP to CoS mapping table.

2) Trusting CoS (**`mls qos trust cos`**) is a little bit different. First, it works for both IP and non-IP packets, since they *both* may carry CoS bits in the Ethernet header. Thus, when the switch trusts CoS on interface, it attempts to build the QoS label based on the CoS bits. If there is no 802.1q/ISL header, the default CoS value from the interface settings is used instead (not the IP/DSCP value from the packet header!). This procedure works for both IP and a non-IP packets, as the switch does not take into account the IP Precedence/DSCP values. The corresponding mapping table allows the switch to compute the internal DSCP value and rewrite it in the IP packet header.

3) You may explicitly assign a specific CoS value to all packet entering the interface using the **`mls qos cos override`** command. This command will enforce the use of CoS value specified via **`mls qos cos x`** command for all IP and non-IP packets. Note that this only works with CoS marking, and the switch deduces DSCP marking using the CoS to DSCP mapping table. Also any trust configuration on the interface conflicts with the override settings and thus the switch removes it.

4) The most flexible option is to use QoS ACLs to perform classification. These are be covered in a separate task.

Copyright © 2010 Internetwork Expert

A special type of trust is conditional trust. That means trusting QoS marking only if the switch detects certain device connected to the port. For example, if the switch sees a Cisco IP Phone via CDP or a Cisco SoftPhone sending CDP packets from the attached PC. The command for conditional trust is **`mls qos trust device <device-name>`** and it instructs the switch to trust the packet marking only if the certain device reports itself via CDP.

To verify your configuration, first try using the switch-specific show commands. Start with R6's port, which uses CoS-based classification. The IOS code will by default map DSCP/IP Precedence values to CoS bits using a simple mapping table – highest 3 bits of the ToS byte maps to the CoS value.

```
Rack1SW2#show mls qos interface fastEthernet 0/6 statistics
FastEthernet0/6

  dscp: incoming
-------------------------------

  0 -  4 :           0           0           0           0           0
  5 -  9 :           0           0           0           0           0
 10 - 14 :           0           0           0           0           0
 15 - 19 :           0           0           0           0           0
 20 - 24 :           0           0           0           0           0
 25 - 29 :           0           0           0           0           0
 30 - 34 :           0           0           0           0           0
 35 - 39 :           0           0           0           0           0
 40 - 44 :           0           0           0           0           0
 45 - 49 :           0           0           0          11           0
 50 - 54 :           0           0           0           0           0
 55 - 59 :           0           0           0           0           0
 60 - 64 :           0           0           0           0
  dscp: outgoing
-------------------------------

  0 -  4 :          16           0           0           0           0
  5 -  9 :           0           0           0           0           0
 10 - 14 :           0           0           0           0           0
 15 - 19 :           0           0           0           0           0
 20 - 24 :           0           0           0           0           0
 25 - 29 :           0           0           0           0           0
 30 - 34 :           0           0           0           0           0
 35 - 39 :           0           0           0           0           0
 40 - 44 :           0           0           0           0           0
 45 - 49 :           0           0           0           0           0
 50 - 54 :           0           0           0           0           0
 55 - 59 :           0           0           0           0           0
 60 - 64 :           0           0           0           0
  cos: incoming
-------------------------------

  0 -  4 :          11           5           0           0           0
  5 -  7 :           0           0           0
```

```
  cos: outgoing
-------------------------------

  0 -  4 :          16            0            0            0            0
  5 -  7 :           0            0            0
Policer: Inprofile:            0 OutofProfile:            0
```

The output consists of two sections: DSCP and CoS values. Those are the values observed in the packets entering leaving the switch interface (i.e. incoming traffic – coming from the router, outgoing traffic – coming to the router). To find statistics for a particular DSCP or CoS value, locate the row corresponding to the value. For example, DSCP 3 maps to the fourth column of the first row:

```
  dscp: incoming
-------------------------------

  0 -  4 :          21            0            0          <0>            0
```

And the CoS value of 2 maps to the third column of the correspond row:

```
  cos: outgoing
-------------------------------

  0 -  4 :        2288            0            0            0            0
```

From the output above, you may note the incoming values corresponding to DSCP 48 (this is the RIP updates, since routing traffic uses IP Precedence of six, which is DSCP 48). Also note, that all incoming routing updates bear CoS value of zero, since IOS does not set the CoS field by default.

Since the switch is set to classify based on CoS it will reset DSCP 48 value in the header to DSCP value of zero. However, the output above does show the DSCP values before the rewrite based on the mapping tables. Note the count of packets with CoS value of "1" – five packets. Those are the Layer 2 keepalives and CDP packets. Since they travel on native VLAN, they have no CoS fields of their own and thus the switch assigns the interface default value to them.

Now configure R6 to mark all outgoing tagged packets with the CoS value of 4, and stream some packets between R6 and R1. To ensure that other switches do not erase our marking, disable MLS QoS on SW1, SW3, and SW4 temporarily. Configure R1 to track packets with DSCP values of CS4 (corresponds to CoS 4 in our CoS to DSCP mapping table):

Copyright © 2010 Internetwork Expert

```
R6:
policy-map SET_COS
 class class-default
  set cos 4
!
interface FastEthernet 0/0
 service-policy output SET_COS

SW1, SW3, & SW4:
no mls qos

R1:
ip access-list extended TEST
 permit icmp any any dscp cs4
 permit ip any any
!
interface FastEthernet 0/0
 ip access-group TEST in

Rack1R6#ping 155.1.146.1 repeat 100 size 36

Type escape sequence to abort.
Sending 100, 36-byte ICMP Echos to 155.1.146.1, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 1/2/4
ms

Rack1R1#sh ip access-lists TEST
Extended IP access list TEST
    10 permit icmp any any dscp cs4 (100 matches)
    20 permit ip any any (927 matches)
```

This test demonstrates that the Catalyst switch actually classifies the IP packets based on the CoS value in Ethernet header. Re-enable MLS QoS on SW1 and see how we can verify QoS marking on SW4, which is a 3550.

The 3550 switch supports a special feature known as the DSCP monitor. The DSCP values monitored are the internal DSCP values (QoS labels) assigned by the switch for incoming packets and seen by the switch in the outgoing packets. You can monitor either packet count or bytes count. You set the monitor mode on special “master” interfaces (e.g. interface Fa 0/1 controls mode for interfaces 1-12 inclusive and interface Fa 0/13 controls mode for interfaces 13-24). You may specify up to 8 DSCP values to monitor in both directions.

Copyright © 2010 Internetwork Expert

```
SW4:
interface FastEthernet 0/1
 mls qos monitor packets
!
interface FastEthernet 0/4
 mls qos monitor dscp 0 16 26 46 48
```

```
Rack1SW4#clear mls qos interface fastEthernet 0/4 statistics
```

Let's see some preliminary statistics. So far, you can see Ingress packets marked with DSCP 16 and DSCP 48. They correspond to the CDP/L2 Keepalives (CoS 2 maps to DSCP 16) and RIP routing updates (IP Precedence of 6 maps to DSCP of 48). Note that all packets are listed as "no_change" since they have been classified on trusted values. Also, note that "Egress" direction does not show any counters for classified packets, since classification is only possible ingress.

```
Rack1SW4#show mls qos interface fastEthernet 0/4 statistics
FastEthernet0/4
Ingress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 0          0          0          0          0
    16: 7          7          0          0          0
    26: 0          0          0          0          0
    46: 0          0          0          0          0
    48: 7          7          0          0          0
Others: 0          0          0          0          0
Egress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 14            n/a        n/a      0          0
    16: 0             n/a        n/a      0          0
    26: 0             n/a        n/a      0          0
    46: 0             n/a        n/a      0          0
    48: 0             n/a        n/a      0          0
Others: 37            n/a        n/a      0          0
```

Now generate some packets with IP precedence of 3 (ToS byte 96) and see how the switch re-marks them:

Copyright © 2010 Internetwork Expert

```
Rack1R4#ping
Protocol [ip]:
Target IP address: 155.1.146.1
Repeat count [5]: 100
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface:
Type of service [0]: 96
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 155.1.146.1, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 1/3/5
ms

Rack1SW4#show mls qos interface fastEthernet 0/4 statistics
FastEthernet0/4
Ingress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 0          0          0          0          0
    16: 60         60         0          0          0
    26: 0          0          100        0          0
    46: 0          0          0          0          0
    48: 55         55         0          0          0
Others: 100        0          0          0          0
Egress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 211           n/a        n/a      0          0
    16: 0             n/a        n/a      0          0
    26: 0             n/a        n/a      0          0
    46: 0             n/a        n/a      0          0
    48: 0             n/a        n/a      0          0
Others: 315           n/a        n/a      0          0
```

Copyright © 2010 Internetwork Expert

This time 100 packets have been classified to the DSCP value of 26. This is because the switch used its mapping table to deduce the internal DSCP value. Of course, the incoming packets had a DSCP value of 96 and you can see them under the "Others" row of the table.

Now, verify the CoS override feature on the port connected to R5. Configure SW2 to detect packets marked with DSCP value of 44 and disable MLS QoS on all switches with except to SW1:

```
SW3, SW4:
no mls qos

SW2:
no mls qos
!
ip access-list extended TEST
 permit ip any any dscp 44
 permit ip any any
!
interface Vlan 58
 ip access-group TEST in

Rack1R5#ping 155.1.58.8 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 155.1.58.8, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 1/2/4
ms
Rack1R5#

Rack1SW2#show ip access-list TEST
Extended IP access list TEST
    10 permit ip any any dscp 44 (101 matches)
    20 permit ip any any (735 matches)
```

The verification of DSCP trust is the simplest of them. Source packets off R1 marked with DSCP value of EF (ToS bytes 184, DSCP numerical value of 46).

```
Rack1R1#ping
Protocol [ip]:
Target IP address: 155.1.146.4
Repeat count [5]: 100
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface:
Type of service [0]: 184
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 1/3/12
ms
```

Next, verify packets matching on SW1 side. Note the 100 packets matched for DSCP value of 46. Note that CDP packets and keepalives increment counters for the CoS value of 1.

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show mls qos interface fastEthernet 0/1 statistics
FastEthernet0/1

  dscp: incoming
-------------------------------

  0 -  4 :           0           0           0           0           0
  5 -  9 :           0           0           0           0           0
 10 - 14 :           0           0           0           0           0
 15 - 19 :           0           0           0           0           0
 20 - 24 :           0           0           0           0           0
 25 - 29 :           0           0           0           0           0
 30 - 34 :           0           0           0           0           0
 35 - 39 :           0           0           0           0           0
 40 - 44 :           0           0           0           0           0
 45 - 49 :           0         100           0          13           0
 50 - 54 :           0           0           0           0           0
 55 - 59 :           0           0           0           0           0
 60 - 64 :           0           0           0           0
  dscp: outgoing
-------------------------------

  0 -  4 :         126           0           0           0           0
  5 -  9 :           0           0           0           0           0
 10 - 14 :           0           0           0           0           0
 15 - 19 :           0           0           0           0           0
 20 - 24 :           0           0           0           0           0
 25 - 29 :           0           0           0           0           0
 30 - 34 :           0           0           0           0           0
 35 - 39 :           0           0           0           0           0
 40 - 44 :           0           0           0           0           0
 45 - 49 :           0           0           0           0           0
 50 - 54 :           0           0           0           0           0
 55 - 59 :           0           0           0           0           0
 60 - 64 :           0           0           0           0
  cos: incoming
-------------------------------

  0 -  4 :           0         127           0           0           0
  5 -  7 :           0           0           0
  cos: outgoing
-------------------------------

  0 -  4 :         126           0           0           0           0
  5 -  7 :           0           0           0
Policer: Inprofile:            0 OutofProfile:             0
```

Now confirm that the switch actually sends DSCP 46 packets out to R4. To do that, you need to find the exit trunk for the MAC address of R4. Find out the MAC address of R4 and look up the exit port. After that, check the MLS QoS statistics for the output port.

```
Rack1R4#show interfaces fastEthernet 0/1
FastEthernet0/1 is up, line protocol is up
  Hardware is AmdFE, address is 0012.43c1.6f01 (bia 0012.43c1.6f01)
<snip>

Rack1SW1#show mac-address-table address 0012.43c1.6f01
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
 146    0012.43c1.6f01    DYNAMIC     Fa0/16
Total Mac Addresses for this criterion: 1

Rack1SW1#show mls qos interface fastEthernet 0/16 statistics
FastEthernet0/16

  dscp: incoming
-------------------------------

  0 -  4 :        8660            0            0            0            0
  5 -  9 :           0            0            0            0            0
 10 - 14 :           0            0            0            0            0
 15 - 19 :           0            0            0            0            0
 20 - 24 :           0            0            0            0            0
 25 - 29 :           0          100            0            0            0
 30 - 34 :           0            0         1159            0            0
 35 - 39 :           0            0            0            0            0
 40 - 44 :           0            0            0            0          100
 45 - 49 :           0          100            0        59898            0
 50 - 54 :           0            0            0            0            0
 55 - 59 :           0         9491            0            0            0
 60 - 64 :           0            0            0            0
  dscp: outgoing
-------------------------------

  0 -  4 :        1629            0            0            0            0
  5 -  9 :           0            0            0            0            0
 10 - 14 :           0            0            0            0            0
 15 - 19 :           0            0            0            0            0
 20 - 24 :           0            0            0            0            0
 25 - 29 :           0          100            0            0            0
 30 - 34 :           0            0          700            0            0
 35 - 39 :           0            0            0            0            0
 40 - 44 :           0            0            0            0          213
 45 - 49 :           0          100            0        24082            0
 50 - 54 :           0            0            0            0            0
 55 - 59 :           0            0            0            0            0
 60 - 64 :           0            0            0            0
```

Copyright © 2010 Internetwork Expert

```
  cos: incoming
------------------------------

  0 -  4 :       69688            0          342          100         2160
  5 -  7 :         100        28168       208834
  cos: outgoing
------------------------------

  0 -  4 :       10796           11            0            0          709
  5 -  7 :         100        15335            4
Policer: Inprofile:            0 OutofProfile:            0
```

Note the number of DSCP EF packets as well as the count of CoS 5 packets that were switched out of the port. Note that may need to clear statistics on this link as well before sending the sample packets sif you want to get more accurate statistics.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.63 Catalyst QoS Marking Pass-Through

- Ensure SW2 classifies packets coming from R6's VLANs 67 and VLAN 146 based on CoS values, but does not change the DSCP value in IP packets.
- Additionally ensure that IP-precedence based classification on the SW4's connection to R4 does not change the CoS value in Ethernet headers.

Configuration

```
SW2:
no mls qos rewrite ip dscp

SW4:
interface FastEthernet 0/4
 mls qos trust dscp pass-through cos
```

Verification

Note

The automatic rewrite feature ensures the uniform marking - that is it takes care of synchronizing L2 and L3 code points. Is it possible to trust DSCP and built the internal QoS label based on its value, but retain the CoS bits in the packet? Alternatively – trust CoS bits and retain the DSCP values? You may need this capability occasionally, if you want to "tunnel" one type of QoS marking through your network, while using the other type for your needs. One possible example could be a Metro Ethernet provider that uses CoS bits for QoS marking in core, but preserves customer DSCP marking.

In the Catalyst 3550, you may set one type of marking as "pass-through". For example, when you trust CoS you may enable DSCP pass-through with the command **`mls qos trust cos pass-through dscp`**. The command with reversed logic is obviously **`mls qos trust dscp pass-through cos`**.

On the 3560 switches, you may only disable DSCP rewrite in IP headers, when you trust the CoS values, using the global command no **`mls qos rewrite ip dscp`**.

Copyright © 2010 Internetwork Expert

To verify QoS "tunneling" on the 3560, perform the following steps. First, temporarily disable MLS QoS in all switches except for SW2 (which connects to R6) to ensure that other switches do not change our QoS marking. Next, configure R6 to mark all outgoing packets with a CoS value of four. After that, configure R1 to "detect" ICMP packets with a DSCP value of 24 using an extended ACL. Lastly, source ICMP packets with a ToS byte of 96 (DSCP 24) out of R6 to R1 and check the QoS statistics.

```
R6:
policy-map SET_COS
 class class-default
  set cos 4
!
interface FastEthernet 0/0
 service-policy output SET_COS

SW1,SW3,SW4:
no mls qos

R1:
no ip access-list extended TEST
ip access-list extended TEST
 permit icmp any any dscp cs3
 permit ip any any
!
interface FastEthernet 0/0
 ip access-group TEST in

Rack1SW2#clear mls qos interface fastEthernet 0/6 statistics

Rack1R6#ping
Protocol [ip]:
Target IP address: 155.1.146.1
Repeat count [5]: 100
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface:
Type of service [0]: 96
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 155.1.146.1, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 1/2/4
ms
Rack1R6#
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip access-list TEST
Extended IP access list TEST
    10 permit icmp any any dscp cs3 (100 matches)
    20 permit ip any any (114 matches)

Rack1SW2#show mls qos interface fastEthernet 0/6 statistics
FastEthernet0/6

  dscp: incoming
-------------------------------

  0 -  4 :           0            0            0            0            0
  5 -  9 :           0            0            0            0            0
 10 - 14 :           0            0            0            0            0
 15 - 19 :           0            0            0            0            0
 20 - 24 :           0            0            0            0          100
 25 - 29 :           0            0            0            0            0
 30 - 34 :           0            0            0            0            0
 35 - 39 :           0            0            0            0            0
 40 - 44 :           0            0            0            0            0
 45 - 49 :           0            0            0           16            0
 50 - 54 :           0            0            0            0            0
 55 - 59 :           0            0            0            0            0
 60 - 64 :           0            0            0            0
  dscp: outgoing
-------------------------------

  0 -  4 :           0            0            0            0            0
  5 -  9 :           0            0            0            0            0
 10 - 14 :           0            0            0            0            0
 15 - 19 :           0            0            0            0            0
 20 - 24 :           0            0            0            0          100
 25 - 29 :           0            0            0            0            0
 30 - 34 :           0            0            0            0            0
 35 - 39 :           0            0            0            0            0
 40 - 44 :           0            0            0            0            0
 45 - 49 :           0            0            0           26            0
 50 - 54 :           0            0            0            0            0
 55 - 59 :           0            0            0            0            0
 60 - 64 :           0            0            0            0
  cos: incoming
-------------------------------

  0 -  4 :           0           10            0            0          116
  5 -  7 :           0            0            0
  cos: outgoing
-------------------------------

  0 -  4 :         126            0            0            0            0
  5 -  7 :           0            0            0
Policer: Inprofile:            0 OutofProfile:            0
```

From the output above you can see that SW2 accepts packets with DSCP 24 and CoS 4 from R6. In accordance with CoS to DSCP mapping table, DSCP should have changed to 32. However, as we see R1 still receives packets marked with DSCP 24 (CS3).

Copyright © 2010 Internetwork Expert

To verify CoS pass-through, we temporarily need to convert the port connected to R4 VLAN146 interface into a trunk and configure a policy-map that sets some CoS value, e.g. 3, in outgoing packets:

```
R4:
policy-map SET_COS
 class class-default
  set cos 3
!
interface FastEthernet 0/1
 no ip address
 service-policy output SET_COS
!
interface FastEthernet 0/1.146
 encapsulation dot1q 146
 ip address 155.1.146.4 255.255.255.0

SW4:
interface FastEthernet0/4
 switchport trunk encapsulation dot1q
 switchport mode trunk
 mls qos trust dscp pass-through cos
```

Configure R6 to "detect" frames marked with CoS 3 and IP packets marked with DSCP EF. After that, source IP packets marked with DSCP EF from R4 to R6. Be sure to disable MLS QoS temporarily on SW1, SW2, and SW3:

```
SW1, SW2, SW3:
no mls qos

R6:
ip access-list extended TEST
 permit icmp any any dscp ef
!
interface FastEthernet 0/0.146
 ip access-group TEST in

class-map COS_3
 match cos 3
!
policy-map MATCH_COS
 class COS_3
!
interface FastEthernet 0/0
 service-policy input MATCH_COS
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#ping
Protocol [ip]:
Target IP address: 155.1.146.6
Repeat count [5]: 100
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface:
Type of service [0]: 184
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 155.1.146.6, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 1/3/4
ms
Rack1R4#
```

Observe the resulting statistics on R6. Note that incoming packets have DSCP value of EF and CoS marking of 3. That means SW4 did not change the CoS field in accordance with DSCP.

```
Rack1R6#show ip access-lists TEST
Extended IP access list TEST
    10 permit icmp any any dscp ef (100 matches)
    20 permit ip any any (339 matches)

Rack1R6#show policy-map int fastEthernet 0/0
 FastEthernet0/0

  Service-policy input: MATCH_COS

    Class-map: COS_3 (match-all)
      104 packets, 13120 bytes
      5 minute offered rate 2000 bps
      Match: cos  3

    Class-map: class-default (match-any)
      8 packets, 3280 bytes
      5 minute offered rate 1000 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

10.64 Catalyst QoS ACL Based Classification & Marking

- Create a policy-map on SW2 that performs the following on traffic received from R6.
  - Set a DSCP value of EF on IPX packets (EtherType value 0x8137)
  - Set a DSCP value of CS3 on ICMP packets
  - Set an IP Precedence of 2 on TELNET packets
  - Trust CoS under the default class
- Configure SW4 and R4 to change the port connected to R4 into a trunk.
- Configure SW4 to trust IP DSCP values in packets coming from R4, and set the CoS to a value of 2.

Configuration

```
SW2:
mls qos rewrite ip dscp
!
mac access-list extended IPX
 permit any any 0x8137 0x0
!
ip access-list extended ICMP
 permit icmp any any
!
ip access-list extended TELNET
 permit tcp any any eq telnet
 permit tcp any eq telnet any
!
class-map match-all TELNET
  match access-group name TELNET
!
class-map match-all ICMP
  match access-group name ICMP
!
class-map match-all IPX
  match access-group name IPX
!
! Note that we set DSCP under non-IP traffic. The switch computes
! the respective CoS value using DSCP-to-CoS table.
!
policy-map CLASSIFY
  class IPX
   set dscp ef
  class ICMP
   set dscp cs3
  class TELNET
   set precedence 2
  class class-default
    trust cos
```

Copyright © 2010 Internetwork Expert

```
!
interface FastEthernet 0/6
 mls qos cos 1
 service-policy input CLASSIFY

SW4:
mls qos cos policy-map
!
ip access-list standard IP_ANY
 permit any
!
class-map IP_ANY
 match access-group name IP_ANY
!
! Note that we can't set in class-default with the 3550
! you need a special class that matches all IP traffic
! in order to implement the policy
!
policy-map CLASSIFY
 class IP_ANY
  trust dscp
  set cos 2
!
interface FastEthernet 0/4
 switchport trunk encapsulation dot1q
 switchport mode trunk
 service-policy input CLASSIFY

R4:
interface FastEthernet 0/1
 no ip address
!
interface FastEthernet 0/1.146
 encapsulation dot1q 146
 ip address 155.1.146.4 255.255.255.0
```

Verification

Note

The most flexible classification option for the Catalyst is using QoS ACLs. Even though they are called QoS ACLs (the term has been borrowed from CatOS) you actually apply them using MQC syntax using class-maps and policy-map. You create MQC classes by matching one of the following:

1) MAC or IP/IPv6 access-list. The MAC access-list matches non-IP packets and IP/IPv6 match IP or IPv6 packets respectively. You can only match IPv6 packets on the 3560. You cannot use MAC ACLs to match IP traffic (though you can use MAC ACLs on the 3550 to match IPv6 traffic).

Copyright © 2010 Internetwork Expert

2) DSCP and IP Precedence values. Note that you cannot match CoS values in Ethernet headers.

3) Additional platform-specific matches like **`match input-interface`** and **`match vlan`**. We will consider them in separate tasks.

Note that the classification criteria is very limited compared to IOS routers. You cannot match size or packet contents. Additionally, you cannot do hierarchical matching, with the exception of the per-VLAN classification feature discussed in a separate task. The limitations are the result of hardware optimization in the Layer 3 switches.

Pay attention to the behavior of the class "class-default" with the Catalyst QoS. This class works fine in the 3560 switches, matching both IP and non-IP traffic. However, it seems to work inconsistently or does not work at all with the 3550 switches. In the latter model, as a workaround, create a special class to match either all IP or all non-IP traffic using ACLs.

Under a class in the policy-map you can either trust (CoS, DSCP, IP Precedence) or set (DSCP, IP Precedence) the respective QoS marking. Note that you cannot set CoS value directly in the 3560 switches, but you can set DSCP for non-IP packets. The switch translates DSCP into CoS using the DSCP-to-CoS mapping table. The same holds true for the 3550 switches – you can set DSCP on the non-IP packets and it is translated to the CoS.

You cat set CoS directly in the 3550 switches in one special case. First, you need the global command **`mls qos cos policy-map`**. Using this feature, you *must* set the DSCP marking *and* set Layer 2 marking using the **`set cos`** command. This simulates the behavior of "CoS pass-through" feature available with interface-level settings. The set cos feature only works when you trust DSCP. Furthermore, the 3550 performs all QoS processing and deduces internal DSCP based on the *trusted* DSCP value, not the CoS value set.

On both switch models, you can either configure **`set dscp`** or **`set ip precedence`** but not both – the one configured later erases the previous one. Also, if a class contains "trust" and "set" statements for the same type of marking (e.g. L3 or L2) the trust statement takes precedence over explicit set.

Aside from that, the trust feature works the same way as it works on the interface, just it has the scope limited to a class defined by ACL.

Copyright © 2010 Internetwork Expert

Remember that applying a service-policy will remove any interface-level QoS settings on the interface, with the exception of the default CoS (which is used by the policy map when you trust CoS inside a class). If the input packet does not match any class in the service-policy, the switch will set all markings to zero. If you trust CoS for IP or non-IP packets and there is no 802.1q/ISL header, the switch will take the default CoS value from the interface settings or use the zero markings.

The verification procedure is somewhat complicated and we will try to do it in systematic manner. First, we configure IPX on both R4 and R6, using IPX SNAP encapsulation. This way we can generate non-IP traffic, understood both by the 3550 and 3560 switches.

```
R4:
ipx routing
!
interface FastEthernet 0/1.146
 ipx network 146 encapsulation snap

R6:
ipx routing
!
interface FastEthernet 0/0.146
 ipx network 146 encapsulation snap
```

The first thing we want to test is to see how the 3560 assign QoS marking. We expect the IPX packets to bear CoS value of 5 as they leave the switch. In addition, we expect the switch to enforce marking for telnet and ICMP packets.

First, prepare SW4 to count CoS 5 packets coming to R4. You will need to trust CoS values on all trunk links and disable QoS on SW3 and SW1 (so that they don't interfere):

```
SW1, SW3:
no mls qos

SW4:
mls qos
!
interface FastEthernet 0/1
 mls qos monitor packets
!
interface FastEthernet 0/4
 mls qos monitor dscp 0 40 46
!
interface range FastEthernet 0/13 - 21
 mls qos trust cos
```

Copyright © 2010 Internetwork Expert

Now send IPX ping packets from R6 to R4. To do that, you need to learn the IPX address of R4 first. After that, check the count of DSCP 40 packets sent to R4. Remember that DSCP values are not the actual, but internal switch DSCP (QoS labels). Thus, we ensure that set DSCP operation works for non-IP packets by the virtue of the mapping tables.

```
Rack1R4#show ipx interface fastEthernet 0/1.146
FastEthernet0/1.146 is up, line protocol is up
  IPX address is 146.000e.8475.1341, 802.1Q vLAN (SNAP) [up]
  Delay of this IPX network, in ticks is 1

Rack1SW4#clear mls qos interface fastEthernet 0/4 statistics

Rack1R6#ping 146.000e.8475.1341

Translating "146.000e.8475.1341"

Type escape sequence to abort.
Sending 5, 100-byte IPX Novell Echoes to 146.000e.8475.1341, timeout is
2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
Rack1R6#

Rack1SW4#show mls qos interface fastEthernet 0/4 statistics
FastEthernet0/4
Ingress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 3          3          7          0          0
    40: 0          0          0          0          0
    46: 0          0          0          0          0
Others: 7          0          0          0          0
Egress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 15            n/a        n/a      0          0
    40: 5             n/a        n/a      0          0
    46: 0             n/a        n/a      0          0
Others: 137           n/a        n/a      0          0
```

Now we need to ensure marking for telnet and ICMP packets going to R1.

```
R1:
ip access-list extended TEST
 permit tcp any any eq telnet precedence immediate
 permit icmp any any dscp cs3
 permit ip any any
!
interface FastEthernet 0/0
 ip access-group TEST in
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#telnet 155.1.146.1
Trying 155.1.146.1 ... Open


User Access Verification

Password: cisco
Rack1R1>en
Password: cisco

Rack1R1#show ip access-list TEST
Extended IP access list TEST
    10 permit tcp any any eq telnet precedence immediate (108 matches)
    20 permit icmp any any dscp cs3
    30 permit ip any any (18 matches)

Rack1R6#ping 155.1.146.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
Rack1R6#

Rack1R1#show ip access-lists TEST
Extended IP access list TEST
    10 permit tcp any any eq telnet precedence immediate (207 matches)
    20 permit icmp any any dscp cs3 (5 matches)
    30 permit ip any any (216 matches)
```

Copyright © 2010 Internetwork Expert

Ensure that packets not matching any of the user-defined classes match class-default and have their CoS value trusted or assigned from the interface default setting. The output shows CoS value of 1, assigned to non-tagged packets such as CDP or L2 keepalives received from R6.

```
Rack1SW2#show mls qos interface fastEthernet 0/6 statistics
FastEthernet0/6

  dscp: incoming
-------------------------------

  0 -  4 :          110            0            0            0            0
  5 -  9 :            0            0            0            0            0
 10 - 14 :            0            0            0            0            0
 15 - 19 :            0            0            0            0            0
 20 - 24 :            0            0            0            0          100
 25 - 29 :            0            0            0            0            0
 30 - 34 :            0            0            0            0            0
 35 - 39 :            0            0            0            0            0
 40 - 44 :            0            0            0            0            0
 45 - 49 :            0          200            0         4761            0
 50 - 54 :            0            0            0            0            0
 55 - 59 :            0            0            0            0            0
 60 - 64 :            0            0            0            0
  dscp: outgoing
-------------------------------

  0 -  4 :         2094            0            0            0            0
  5 -  9 :            0            0            0            0            0
 10 - 14 :            0            0            0            0            0
 15 - 19 :            0            0            0            0            0
 20 - 24 :            0            0            0            0          100
 25 - 29 :            0            0            0            0            0
 30 - 34 :            0            0            0            0            0
 35 - 39 :            0            0            0            0            0
 40 - 44 :            0            0            0            0            0
 45 - 49 :            0          200            0         4988            0
 50 - 54 :            0            0            0            0            0
 55 - 59 :            0            0            0            0            0
 60 - 64 :            0            0            0            0
  cos: incoming
-------------------------------

  0 -  4 :         1205         1401            0            0         5184
  5 -  7 :            0            0            0
  cos: outgoing
-------------------------------

  0 -  4 :         5089            0            0         1313            0
  5 -  7 :            0         1064            0
Policer: Inprofile:            0 OutofProfile:             0
```

Now configure R6 to mark all outgoing *tagged* packets (the router may only set CoS in trunk tagged packets) with a CoS value of 4. After that, send HTTP traffic to R1 from R6 and ensure R1 sees it marked with DSCP value of CS4, which corresponds to automatic rewrite from CoS value of 4. This procedure verifies that the 3560 actually trusts CoS values in class-default packets even for IP packets.

```
R1:
no ip access-list extended TEST
ip access-list extended TEST
 permit tcp any any eq 80 dscp cs4
 permit ip any any
!
interface FastEthernet 0/0
 ip access-group TEST in
!
ip http server

R6:
policy-map SET_COS
 class class-default
   set cos 4
!
interface FastEthernet 0/0
 service-policy output SET_COS

Rack1R6#telnet 155.1.146.1 80
Trying 155.1.146.1, 80 ... Open
GET /

WWW-Authenticate: Basic realm="level_15_access"

401 Unauthorized

[Connection to 155.1.146.1 closed by foreign host]
Rack1R6#

Rack1R1#show ip access-list TEST
Extended IP access list TEST
    10 permit tcp any any eq www dscp cs4 (5 matches)
    20 permit ip any any (10 matches)
```

Copyright © 2010 Internetwork Expert

The last thing to verify is the simultaneous DSCP trusting and CoS setting in the 3550. We need to make sure that an IP packet sent from R4 reaches R6 with non-synchronized Layer 3 and Layer 2 markings. To verify that, shut down the VLAN 67 interface of R6 and VLAN 146 interface of R1. This is to ensure that R6 only receives routing updates from R4.

Now R4 generate routing updates with an IP precedence of six. The switch trusts this value but sets CoS to "2". Therefore, we should expect packets with DSCP value of CS6 and CoS 2 on R6.

Configure R6 to "detect" both types of marking, and disable MLS QoS in SW1, SW2 and SW3:

```
SW1, SW2, SW3:
no mls qos

R6:
class-map COS_2
 match cos 2
class-map COS_6
 match cos 6
!
policy-map MATCH_COS
 class COS_2
 class COS_6
!
ip access-list extended TEST
 permit udp any any eq rip dscp 48
 permit ip any any
!
interface FastEthernet 0/0
 service-policy input MATCH_COS
 ip access-group TEST in

Rack1R6#clear access-list counters

Rack1R6#clear counters fastEthernet 0/0
Clear "show interface" counters on this interface [confirm]

Rack1R6#show ip access-list TEST
Extended IP access list TEST
    10 permit udp any any eq rip dscp cs6 (3 matches)
    20 permit ip any any
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#show policy-map int fa 0/0
 FastEthernet0/0

  Service-policy input: MATCH_COS

    Class-map: COS_2 (match-all)
      3 packets, 1650 bytes
      5 minute offered rate 0 bps
      Match: cos  2

    Class-map: COS_6 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps
      Match: cos  6

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

  Service-policy output: SET_COS

    Class-map: class-default (match-any)
      6 packets, 450 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      QoS Set
        cos 4
          Packets marked 3
```

This example demonstrates that it is possible in the 3550 to set CoS and Trust DSCP independently for the same packet.

Copyright © 2010 Internetwork Expert

10.65 Catalyst 3550 Per-Port Per-VLAN Classification

- Configure SW4 so that ICMP traffic coming in on the port connected to R4 has the DSCP value set to CS3.
- For IPX traffic with EtherType 0x8137 set the CoS field to 2.
- Ensure your classification only affects traffic on VLAN 146 but not any VLANs trunked to R4 in the future.

Configuration

```
SW4:
mls qos
!
ip access-list extended ICMP
 permit icmp any any
!
mac access-list extended IPX
 permit any any 0x8137 0x0
!
! Second-level class-maps
!
class-map ICMP
 match access-group name ICMP
!
class-map IPX
 match access-group name IPX
!
! First-level class-maps
!
class-map VLAN_146_ICMP
 match vlan 146
 match class-map ICMP
!
class-map VLAN_146_IPX
 match vlan 146
 match class-map IPX
!
! Note that we set DSCP for IPX packets
! In fact, this is the internal DSCP value, not the IP DSCP.
! You cannot set CoS directly - only when trusting DSCP.
! DSCP 16 translates to CoS 2 by the virtue of the default
! DSCP-to-CoS table.
!
policy-map PER_PORT_PER_VLAN
 class VLAN_146_ICMP
   set dscp CS3
 class VLAN_146_IPX
   set dscp 16
!
interface FastEthernet 0/4
  no service-policy input CLASSIFY
  service-policy input PER_PORT_PER_VLAN
```

Copyright © 2010 Internetwork Expert

Verification

Note

Per-Port Per-VLAN classification adds granularity to the classification process based on ACLs. You may create a policy-map that matches specific traffic classes insides specific VLANs. The only limitation is that the policy-map is applied at the port level, not globally to the VLAN.

Usually you apply a per-port per-VLAN (PPPV) policy-map to a trunk port. However, it is possible to apply the PPPV classification to an access-port, if the access-vlan matches any of the VLANs configured in the policy-map.

Remember to observe the following strict syntax rules when configuring PPPV classification in the 3550 model:

1) You must use two-levels of class-maps. The first level class-map must have the following format:

```
class-map L1_CLASS
  match vlan <VLAN ID>
  match class-map L2_CLASS
```

The **match vlan** statement must always be the first statement in the class-map, and the only allowed second statement is match class-map. In addition, both first level and second-level class-maps must be of type **match-any**.

2) All first-level class-maps must have the **match vlan** statement, and you cannot mix them with regular class-maps.

3) The second level class map is only allowed a *single* match statement, and it cannot be **match class-map**. For example,

```
class-map L2_CLASS
 match access-group name <NAME>
```

4) The policy-map lists the above-defined first-level classes with respective actions (e.g. set/trust statements):

```
policy-map PPPV
 class L1_CLASS
  set dscp ef
```

Copyright © 2010 Internetwork Expert

To start verification, enable IPX on R4 and R6. We will need that to generate non-IP packets. After that, configure SW4 to monitor DSCP values of interest in packets entering the switchports.

```
R4:
ipx routing
!
interface FastEthernet 0/1.146
 ipx network 146 encapsulation snap

R6:
ipx routing
!
interface FastEthernet 0/0.146
 ipx network 146 encapsulation snap

SW4:
interface FastEthernet 0/1
 mls qos monitor packets
!
interface FastEthernet 0/4
 mls qos monitor dscp 0 16 24 48
```

Now send ICMP packets to R1 and IPX packets to R6. Before you start, learn the IPX addres of R6.

```
Rack1R6#show ipx interface fastEthernet 0/0.146
FastEthernet0/0.146 is up, line protocol is up
  IPX address is 146.0011.9221.da80, 802.1Q vLAN (SNAP) [up]
<snip>

Rack1SW4#clear mls qos interface fastEthernet 0/4
statistics
Rack1SW4#

Rack1R4#ping 155.1.146.1 repeat 50

Type escape sequence to abort.
Sending 50, 100-byte ICMP Echos to 155.1.146.1, timeout is 2
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (50/50), round-trip min/avg/max
= 1/2/4 ms

Rack1R4#ping 146.0011.9221.da80

Translating "146.0011.9221.da80"
```

```
Type escape sequence to abort.
Sending 5, 100-byte IPX Novell Echoes to 146.0011.9221.da80,
timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max =
1/2/4 ms
```

Copyright © 2010 Internetwork Expert

Now verify the statistics on SW4 port connected to R4:

```
Rack1SW4#show mls qos interface fastEthernet 0/4 statistics
FastEthernet0/4
Ingress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 72         17         16         0          0
    16: 0          0          5          0          0
    24: 0          0          50         0          0
    48: 16         0          0          0          0
Others: 0          0          0          0          0
Egress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 510           n/a        n/a      0          0
    16: 0             n/a        n/a      0          0
    24: 0             n/a        n/a      0          0
    48: 32            n/a        n/a      0          0
Others: 971           n/a        n/a      0          0
```

Note the packet counters in “classified” column. The column list packets that have their DSCP value changed from their original value. Remember that from the internal DSCP standpoint both IP and non-IP packets have a DSCP value associated with them.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.66 Catalyst 3560 Per-VLAN Classification

- Enable VLAN-based QoS on the trunk ports of SW1 and on the port connected to R1.
- Configure SW1 to mark all TCP traffic on VLAN 146 with a DSCP value of EF.
- IPX SNAP-encapsulated traffic should be marked with a CoS value of 4.

Configuration

```
SW1:
mls qos
!
interface FastEthernet 0/1
 mls qos vlan-based
!
! Enable VLAN-based QoS on the trunks
!
interface range FastEthernet 0/13 - 21
 mls qos vlan-based
!
ip access-list extended TCP
 permit tcp any any
!
mac access-list extended IPX
 permit any any 0x8137 0x0
!
! First-level class-maps
!
class-map TCP
 match access-group name TCP
!
class-map IPX
 match access-group name IPX
!
! VLAN-based policy-map
!
policy-map PER_VLAN
 class TCP
  set dscp ef
 class IPX
  set dscp 32
!
! Apply the policy-map to SVI
!
interface VLAN 146
 service-policy input PER_VLAN
```

Copyright © 2010 Internetwork Expert

Verification

Note

The 3560 model configuration differs significantly from the 3550 with respect to per-VLAN classification. You cannot use the **match vlan** statement anymore. Rather, you may enable certain ports for per-VLAN QoS services using the command **mls qos vlan-based**. The mentioned ports will inherit all QoS configurations from the respective SVIs. This approach allows for true switch-wide VLAN QoS. With the VLAN-based QoS model you apply the service policy to an SVI, and all ports enabled for VLAN QoS that have this VLAN active (trunks or access ports) will use the service policy on the respective VLAN.

The major difference from the 3550 model is that you apply classification to all ports enabled for VLAN QoS. You may selectively disable VLAN-wide classification on specific ports by using **no mls qos vlan-based** command.

Verifications of this configuration is tricky due to the limited set of show commands. We need to verify the following:

1) Traffic coming from R1 (IP and non-IP) receives proper marking in SW1
2) VLAN146 traffic coming across SW1 on the trunk ports also receives proper marking

Let's start with R1's connection. Enable IPX on R1, R4, and R6 to generate non-IP traffic. At the same time, Configure SW4 to monitor DSCP values 0, 32, 46, and 48 on the connection to R4's port. Note that DSCP values of 32 and 46 correspond to IPX and TCP traffic. Lastly, configure SW4 to trust DSCP markings on all trunk connections - this will ensure it does not erase the marking in IP/non-IP packets. (Remember when you trust DSCP, the switch classifies non-IP packets trusting their CoS).

```
R1:
ipx routing
!
interface FastEthernet 0/0
 ipx network 146 encapsulation snap

R4:
ipx routing
!
interface FastEthernet 0/1.146
 ipx network 146 encapsulation snap
```

Copyright © 2010 Internetwork Expert

```
R6:
ipx routing
!
interface FastEthernet 0/0.146
 ipx network 146 encapsulation snap

SW4:
interface FastEthernet 0/1
 mls qos monitor packets
!
interface FastEthernet 0/4
 mls qos monitor dscp 0 32 46 48
!
interface range FastEthernet 0/13 - 21
 mls qos trust dscp
```

At this point, clear the MLS QoS statistics on the connection to R4. After that, find the IPX address of R4 in VLAN146 and then generate IPX and TCP traffic from R1 to R4. Also, make sure you disabled MLS QoS in all non-participating switches, i.e. SW2 and SW3:

```
SW2, SW3:
no mls qos

Rack1SW4#clear mls qos interface fastEthernet 0/4 statistics

Rack1R4#show ipx interface fastEthernet 0/1.146
FastEthernet0/1.146 is up, line protocol is up
  IPX address is 146.000f.8f0f.6101, 802.1Q vLAN (SNAP) [up]
<snip>

Rack1R1#ping 146.000f.8f0f.6101

Translating "146.000f.8f0f.6101"

Type escape sequence to abort.
Sending 5, 100-byte IPX Novell Echoes to 146.000f.8f0f.6101, timeout is
2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1R1#telnet 155.1.146.4
Trying 155.1.146.4 ... Open


User Access Verification

Password: cisco
Rack1R4>exit

[Connection to 155.1.146.4 closed by foreign host]
```

```
Rack1SW4#show mls qos interface fastEthernet 0/4 statistics
FastEthernet0/4
Ingress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 26         21         4          0          0
    32: 0          0          0          0          0
    46: 1          0          0          0          0
    48: 3          0          0          0          0
Others: 0          0          5          0          0
Egress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 62            n/a        n/a      0          0
    32: 5             n/a        n/a      0          0
    46: 21            n/a        n/a      0          0
    48: 77            n/a        n/a      0          0
Others: 221           n/a        n/a      0          0
```

Note that R4 received 5 packets marked with internal DSCP of 32 which corresponds to the input CoS of 4:

```
Rack1SW4#show mls qos maps cos-dscp
   Cos-dscp map:
        cos:   0  1  2  3  4  5  6  7
      --------------------------------
       dscp:   0  8 16 24 32 40 48 56
```

There are also 21 packets from the TCP sessions initiated off R1 and marked with DSCP value of EF.

Now we are going to send IPX packets from R6 to R4 across SW1. To enforce this path, we shutdown all SW2 links connected to other switches in the topology. After that, send IPX packets from R6 to R4.

```
SW2:
interface range FastEthernet 0/16 - 21
 shutdown
```

```
Rack1R6#ping 146.000f.8f0f.6101

Translating "146.000f.8f0f.6101"

Type escape sequence to abort.
Sending 5, 100-byte IPX Novell Echoes to 146.000f.8f0f.6101, timeout is
2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
Rack1R6#
```

Copyright © 2010 Internetwork Expert

```
Rack1SW4#show mls qos interface fastEthernet 0/4 statistics
FastEthernet0/4
Ingress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 200        190        158        0          0
    32: 0          0          0          0          0
    46: 1          0          0          0          0
    48: 157        0          0          0          0
Others: 0          0          10         0          0
Egress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 1521          n/a        n/a      0          0
    32: 10            n/a        n/a      0          0
    46: 21            n/a        n/a      0          0
    48: 3291          n/a        n/a      0          0
Others: 9683          n/a        n/a      0          0
```

As you can see from the statistics output on SW4, now we got 5 more packets with internal DSCP of 32. Those are the packets sent from R6 across SW1. Since they traversed trunks that have VLAN-based QoS enabled, the switch classified them according to the VLAN policy.

Copyright © 2010 Internetwork Expert

10.67 Catalyst QoS Port-Based Policing and Marking

- Configure SW4's port connected to R4 as follows:
  - Limit the input rate of ICMP packets to 64Kbps. Meter using a burst size value of 16Kbytes. Trust DSCP values in conforming packets and re-mark exceeding traffic down to CS1.
  - Irrespective of the DSCP value, use a CoS value of two for ICMP packets entering the switch.
  - Mark traffic from WWW servers with DSCP CS2. If the traffic exceeds 256 Kbps, re-mark it down to CS1. Assume the burst size of 32000 bytes.
  - Meter the rate of non-IP traffic at 128Kbps using the minimum burst size. Set a CoS value of 0 for conforming traffic, and re-mark exceeding packets with a CoS value of 1.
  - Assume that incoming ICMP packets are pre-colored either with DSCP AF31 or DSCP CS3.
- Configure SW1 to limit the total rate of traffic coming from R1 to 128Kbps. Use a large enough burst size to accommodate 10ms of traffic generated by R1 transmitting at the full 100Mbps interface rate.

Configuration

```
SW1:
mls qos
!
policy-map POLICE
 class class-default
  police 128000 125000
!
interface FastEthernet 0/1
 service-policy input POLICE
```

Copyright © 2010 Internetwork Expert

```
SW4:
mls qos
!
! Allow setting CoS in policy-map
!
mls qos cos policy-map
!
ip access-list extended ICMP
 permit icmp any any
!
ip access-list extended HTTP
 permit tcp any eq 80 any
!
mac access-list extended ALL_NON_IP
 permit any any 0x0 0xFFFF
!
class-map ICMP
 match access-group name ICMP
!
class-map HTTP
 match access-group name HTTP
!
class-map ALL_NON_IP
 match access-group name ALL_NON_IP
!
policy-map POLICE_INBOUND
 class ICMP
  trust dscp
  set cos 2
  police 64000 16000 exceed-action policed-dscp-transmit
 class HTTP
  set dscp cs2
  police 256000 32000 exceed-action policed-dscp-transmit
 class ALL_NON_IP
  set dscp 0
  police 128000 8000 exceed-action policed-dscp-transmit
!
! Mapping original DSCP values to new DSCP values
! We mention all DSCP values that could be potentially
! assigned to inbound traffic.
!
mls qos map policed-dscp 0 16 24 26 to 8
!
interface FastEthernet 0/4
 no service-policy input PER_PORT_PER_VLAN
 service-policy input POLICE_INBOUND
```

Copyright © 2010 Internetwork Expert

Verification

Note

Along with the classification settings in policy-maps, you may also set policer parameters to control the rates of input traffic. The policer applies only to the traffic matching the respective class. The only two metering parameters are average rate and burst size. The switch always accepts conforming traffic, and for exceeding traffic, you have only two options: drop or set policed DSCP and transmit.

The switch determines the markdown DSCP based on the global mapping table. This table specifies original DSCP and the DSCP applied when policer implements markdown actions. As usual, if the packet carries Layer 2 and Layer 3 markings, the switch uses DSCP to CoS mapping table to find the new CoS value.

If you apply a per-port policy to a trunk port, it applies to all VLANs traffic at the same time (e.g. IP traffic from VLAN1, VLAN2 etc).

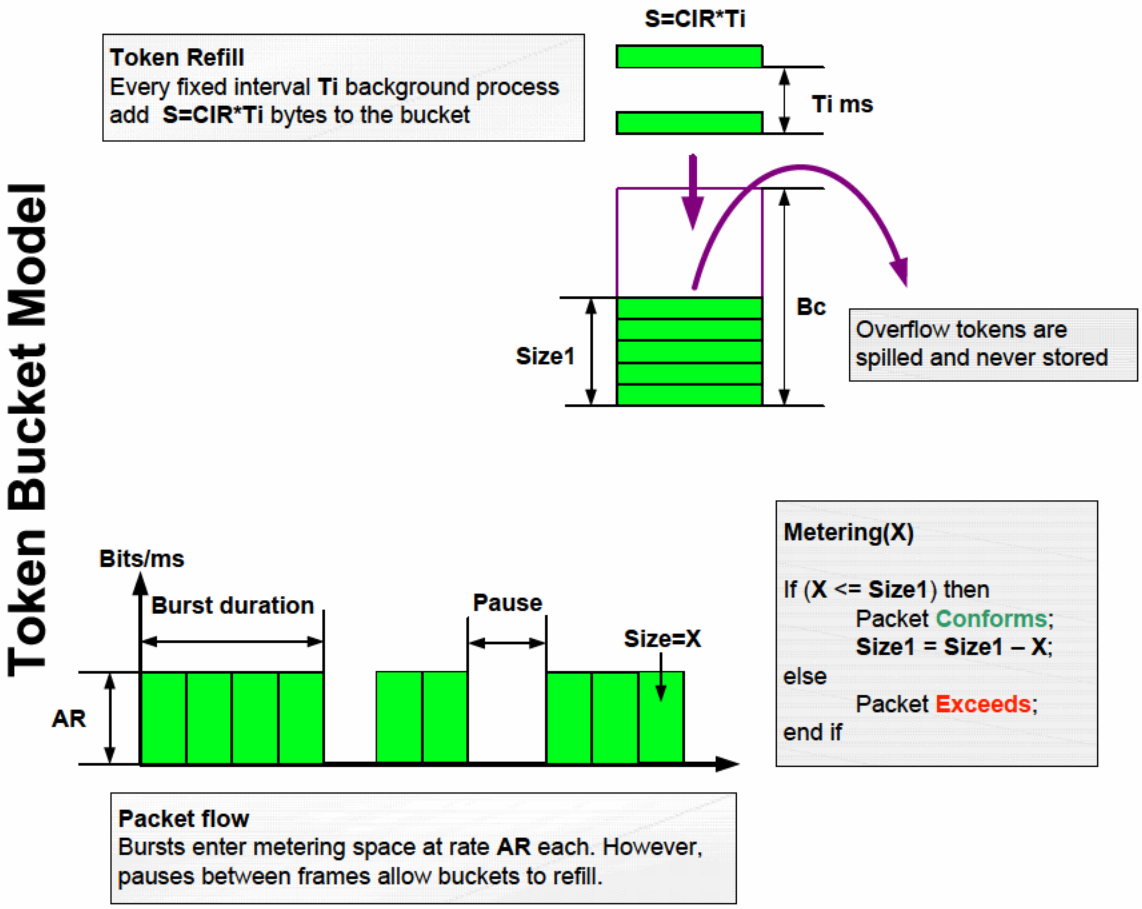
In our task, the burst size for R1 is 100Mbps*10ms/8 = 125000. Note that the switch will round up the burst size to the closest value acceptable for hardware optimization.

Another special thing is handling ICMP traffic. Per the requirements, we trust the DSCP value and remark it down to CS1 if the traffic exceeds. The problem is in general we don't know the DSCP values ahead of time, unless someone else pre-colors them. In this particular case, we know that traffic may carry either CS3 or AF31 DSCP code-points and we can configure policed DSCP mappings accordingly (24 and 26 map to 8). In real life, it's common to first set a fixed DSCP value and mark it down for exceeding traffic, like with do with HTTP class (16 maps to 8).

Finally, we set the CoS value in parallel with trusting DSCP value for the ICMP traffic. This results in Layer 2 marking calculated independently of DSCP. Effectively, not even the markdown action affects the CoS value – it remains the same irrespective of the policer action and DSCP changes.

As for the actual implementation of policing on the Catalyst switches, ASICs allow policing at high rates. Such optimization results in certain limitations imposed on rates and bursts sizes. The Catalyst switches use the familiar token bucket algorithm, which we discussed in the section dedicated to CAR, as follows.

Copyright © 2010 Internetwork Expert



Here **Ti** value is fixed at 8000 times per second (1/8000, or 0,125ms). From the equation S=CIR*Ti and based on the fact that Bc>=S we can see that Bc should be not less than CIR/(8*8000). For example with CIR=100Mbps this value is around 1600 bytes. However, the minimum allowed burst size in the 3550/3560 models is 8000 bytes so you don't have to worry about those details.

The switch's capability of remarking traffic allows changing DSCP values only. You cannot customize markdown settings per policy-map but only set the markdown settings globally using policed-dscp mapping table. This is the limitation of the hardware-optimized platform. The table maps the original DSCP values to their markdown equivalents. Note that the "original" DSCP value is the value taken from the QoS label, not the packet itself. Thus, if you have either a `trust` or `set` commands under a class map, the switch will take DSCP values for markdown lookup based on the "set " command value or on the trusted element value.

Copyright © 2010 Internetwork Expert

Note one important limitation – policing only applies to hardware switched traffic. If you have traffic destined to the switch's SVI and it ends up on switch's CPU, the policing configuration will not affect such traffic.

For this task, we split verification procedure into three steps. First, we verify the ICMP packets markdown. Per our configuration, the switch should trust the DSCP value, and remark it in the case it exceeds the configured rate.

```
SW1, SW2, SW3:
no mls qos

R1:
interface FastEthernet 0/0
 shutdown

R6:
class-map COS_2
 match cos 2
!
class-map COS_1
 match cos 1
!
! Policy map to count CoS values
!
policy-map MATCH_COS
 class COS_1
 class COS_2
!
! Policy map to detect marked ICMP packets
!
ip access-list extended DETECT
 permit icmp any any dscp af31
 permit icmp any any dscp cs1
 permit ip any any
!
interface FastEthernet 0/0
 service-policy input MATCH_COS
 load-interval 30
!
interface FastEthernet 0/0.146
 ip access-group DETECT in
!
interface FastEthernet 0/0.67
 shutdown
```

Copyright © 2010 Internetwork Expert

Now configure R4 to source a stream of packets marked with DSCP AF31 and shaped down to 128Kbps. The switch should remark approximately half of the packets coming into the port down to CS1. However, the packets should remain marked with CoS value of two, no matter what.

```
R4:
interface FastEthernet 0/1.146
 traffic-shape rate 128000 7936 7936 1000

Rack1R6#clear access-list counters

Rack1R6#clear counters fastEthernet 0/0
Clear "show interface" counters on this interface [confirm]

Rack1R4#ping
Protocol [ip]:
Target IP address: 155.1.146.6
Repeat count [5]: 10000000
Datagram size [100]: 1500
Timeout in seconds [2]: 0
Extended commands [n]: y
Source address or interface:
Type of service [0]: 104
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 10000000, 1500-byte ICMP Echos to 155.1.146.6, timeout is 0
seconds:
 ............................................................
 ............................................................
 ............................................................
 ............................................................
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#show policy-map interface fastEthernet 0/0
 FastEthernet0/0

  Service-policy input: MATCH_COS

    Class-map: COS_1 (match-all)
      0 packets, 0 bytes
      30 second offered rate 0 bps
      Match: cos  1

    Class-map: COS_2 (match-all)
      892 packets, 1354056 bytes
      30 second offered rate 124000 bps
      Match: cos  2

    Class-map: class-default (match-any)
      9 packets, 4410 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any

Rack1R6#show ip access-lists
Extended IP access list DETECT
    10 permit icmp any any dscp af31 (437 matches)
    20 permit icmp any any dscp cs1 (435 matches)
    30 permit ip any any (27 matches)
```

From the statistics, it is clear that half of the packets exceed and SW4 remarks them down to DSCP value of CS1. At the same time, we don't see any packet with a CoS value of 1, which means switch does not change the CoS value.

You can verify the HTTP re-marking in the same way. However, this time the switch will modify CoS values according to the policed DSCP value. We will demonstrate a simplified verification. First, set R4 as an HTTP server and start downloading a file from R4 down to R6. Before that, clear MLS QoS statistics counters on the port of SW4 connected to R4 and enable monitoring of DSCP values "0 CS1 and CS2". Ensure that no other switches with except for SW4 have MLS QoS enabled. Don't forget to remove traffic-shaping configuration on R4's VLAN146 interface for this verification.

```
SW1, SW2, SW3:
no mls qos

SW4:
mls qos
!
interface FastEthernet 0/1
 mls qos monitor packets
!
interface FastEthernet 0/4
 mls qos monitor dscp 0 8 16
```

Copyright © 2010 Internetwork Expert

```
R4:
ip http server
ip http path flash:

Rack1SW4#clear mls qos interface fastEthernet 0/4 statistics

Rack1R6#copy http://admin:cisco@155.1.146.4/c1841-adventerprisek9-
mz.124-24.T.bin null:
Loading http://***********@155.1.146.4/c1841-adventerprisek9-mz.124-
24.T.bin
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

Rack1SW4#show mls qos interface fastEthernet 0/4 statistics
FastEthernet0/4
Ingress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 1142       1           1           0          0
    8 : 0          0           0           0          0
    16: 0          0           1141        731        0
Others: 1          0           0           0          0
Egress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 1746          n/a        n/a       0          0
    8 : 0             n/a        n/a       0          0
    16: 0             n/a        n/a       0          0
Others: 84            n/a        n/a       0          0
```

From this output, we can see that all incoming packets are DSCP 0, but the switch classifies them as CS2. Some packets are policed down, though we can't see the new DSCP value.

Now check the CoS monitoring setup at R6 to ensure it receives packets marked with a CoS value of 1 (marked down packets):

```
Rack1R6#show policy-map interface fastEthernet 0/0
 FastEthernet0/0

  Service-policy input: MATCH_COS

    Class-map: COS_1 (match-all)
      32758 packets, 47265183 bytes
      30 second offered rate 1134000 bps
      Match: cos  1
<snip>
```

Copyright © 2010 Internetwork Expert

The last thing to verify now is markdown of non-IP packets. Generate a barrage of IPX packets from R4 to R6 and ensure that you can see CoS 1 packets (exceeding) going down from SW2 on its connection to R6:

```
Rack1R6#show ipx interface fastEthernet 0/0.146
FastEthernet0/0.146 is up, line protocol is up
  IPX address is 146.0011.9221.da80, 802.1Q vLAN (SNAP) [up]

Rack1R4#ping
Protocol [ip]: ipx
Target IPX address: 146.0011.9221.da80
Repeat count [5]: 1000000
Datagram size [100]: 1000
Timeout in seconds [2]: 0
Verbose [n]:
Type escape sequence to abort.
Sending 1000000, 1000-byte IPX Novell Echoes to 146.0011.9221.da80, timeout is
0 seconds:
<snip>

Rack1SW2#show mls qos interface fastEthernet 0/6 statistics
FastEthernet0/6

<snip>
-------------------------------

  0 -  4 :        3258            0            0            0            0
  5 -  7 :           0            0            0
  cos: outgoing
-------------------------------

  0 -  4 :          78         3179            0            0            0
  5 -  7 :           0            0            0
Policer: Inprofile:            0 OutofProfile:            0
```

Note that we constantly get CoS 1 marked packets transmitted on the port connected to R6. Those IPX packets exceeded the configured policer rate in SW4.

The last and the simplest thing to verify is the policer configured on the port connected to R1. Simply send stream of ICMP packets to R6, using large packet sizes.

```
Rack1R1#ping 155.1.146.6 size 1500 repeat 10000

Type escape sequence to abort.
Sending 10000, 1500-byte ICMP Echos to 155.1.146.6, timeout is 2
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!.
Success rate is 97 percent (128/131), round-trip min/avg/max = 4/5/9 ms
```

Note that the length of the first “burst” is much larger compared to the length of sequential bursts. Here are the calculations why this occurs:

First, we set a policer burst size to 125000 bytes. Divide this value by 1514 byte sized packets and you get 83 packets in a single burst. That means we can “instantly” send 83 packets in the network. However, the router sends each packet with 5ms interval (average RTT shown in the output). This means, while sending those packets, the switch accumulates the following credit:

CIR*Interval = 128000*5ms*83 = 53120 bits = 6640 which is 4 packets of size 1514 bytes.

This means that initially we can send around 87 packets (which is easy to verify by counting the “!” signs).

Next, the policer drops a packet and the router waits 2 seconds of timeout. During this interval, the switch accumulates

CIR*Interval = 128000*2s = 256000 bits = 32000 bytes or 21 packet of size 1514 bytes.

This means that every next burst would be of size 21 packets or a bit larger, thanks to extra-accumulated credit. This fact is easy to verify counting the “!” signs in the second and third “bursts”.

Copyright © 2010 Internetwork Expert

10.68 Catalyst 3560 Per-Port Per-VLAN Policing

- Remove any previous policing configuration on VLAN146 in SW2.
- Configure SW2 to limit the rate of IP traffic entering VLAN146 on every trunk port to 128Kbps.
- Mark all conforming packets with AF21 and exceeding with CS1.
- Use the burst size of twice the minimal size.
- Restrict the amount of IP traffic entering from R6 on VLAN146 to 256Kbps, and drop any exceeding packets. Use the burst size of 32000 bytes.

Configuration

```
SW2:
mls qos map policed-dscp 18 to 8
!
! For second level policy you can only match input interfaces
!
class-map TRUNKS
 match input-interface FastEthernet 0/13 -  FastEthernet 0/21
!
! The second class-map matches a group of ports or a single port
!
class-map PORT_TO_R6
 match input-interface FastEthernet 0/6
!
! IP traffic: ACL and class-map
!
ip access-list extended IP_ANY
 permit ip any any
!
class-map IP_ANY
 match access-group name IP_ANY
!
! Second-level policy-map may only police, but not mark
!
policy-map INTERFACE_POLICY
 class TRUNKS
  police 128000 16000 exceed policed-dscp-transmit
 class PORT_TO_R6
  police 256000 32000
!
! First level policy-map may only mark, not police
! VLAN aggregate policing is not possible in the 3560
!
policy-map VLAN_POLICY
 class IP_ANY
  set dscp af21
  service-policy INTERFACE_POLICY
!
interface Vlan 146
 service-policy input VLAN_POLICY
```

Copyright © 2010 Internetwork Expert

```
!
! Enable VLAN-based QoS on the ports
!
interface FastEthernet 0/6
 mls qos vlan-based
!
! Some of these are port-channel members, but it's OK
!
interface range FastEthernet 0/13 - 21
 mls qos vlan-based
```

Verification

Note

The first thing to remember about VLAN-based QoS in the 3560 is that you can use either single-level or two-level policy-maps. In the case of single-level policy map, you match classes of traffic globally, on all ports with the VLAN active and enabled for VLAN QoS. You cannot police traffic using single-level policy map, that is, VLAN-wide aggregate policing is not possible in the 3560. However, it is allowed to restrict the amount of traffic entering the VLAN on every specific port.

Think of a VLAN as cloud, with switch ports connected to this cloud. The first level policy applies to VLAN traffic globally, where the second-level policy may only match port ranges and applies policing to individual ports. It is not possible to apply aggregate policing among different ports, even if they are under the same range of a class map.

The syntax is pretty much MQC based, and the policed DSCP configuration is the same as with the port-level policy maps. You can match up to six ports in a class-map assigned to 2nd level policy-map. Separate different ports with spaces and separate ranges by a hyphen. A port range counts as two entries in match clause.

However, there is one limitation specific to dual-level policy-maps. You cannot apply the 2nd level policy map under the class-default, only under user defined classes. Even though in such configuration the class-default will match and mark any traffic, the second-level policy map will not be active.

Copyright © 2010 Internetwork Expert

The verification consists of two parts. First, we need to verify the limit imposed on traffic entering from a VLAN 146 port. We should verify that it applies only to VLAN 146 traffic and does not affect VLAN 67. To accomplish this, send some traffic destined to R1. Disable MLS QoS in SW1, SW3, and SW4 to avoid any side effects from existing QoS configuration.

```
SW1, SW3, SW4:
no mls qos

Rack1R6#ping 155.1.146.1 repeat 100000 size 1500 timeout 1

Type escape sequence to abort.
Sending 10000, 1500-byte ICMP Echos to 155.1.146.1, timeout is 2
seconds:
!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!
!.!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!.
Success rate is 95 percent (115/120), round-trip min/avg/max = 4/5/9 ms

Rack1R6#ping 155.1.67.7 repeat 10000 size 1500 timeout 1

Type escape sequence to abort.
Sending 10000, 1500-byte ICMP Echos to 155.1.67.7, timeout is 1
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
<snip>
Success rate is 100 percent (725/725), round-trip min/avg/max = 1/3/8
ms
```

Next, configure R1 to generate a packet flow of rate 256Kbps. Use traffic shaping and ping flooding to accomplish this. After that, observe the statistics for packets leaving towards R6:

```
R1:
interface FastEthernet 0/0
 traffic-shape rate 256000

Rack1R1#ping 155.1.146.6 repeat 100000000 size 1000 timeout 0

Type escape sequence to abort.
Sending 100000000, 1000-byte ICMP Echos to 155.1.146.6, timeout is 0 seconds:
 ..............................................................
 ..............................................................
 ..............................................................
 ..............................................................
 ..............................................................
<snip>
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show mls qos interface fastEthernet 0/6 statistics
FastEthernet0/6

  dscp: incoming
-------------------------------

<snip>
  dscp: outgoing
-------------------------------

  0 -  4 :           4            0            0            0            0
  5 -  9 :           0            0            0          577            0
 10 - 14 :           0            0            0            0            0
 15 - 19 :           0            0            0          535            0
 20 - 24 :           0            0            0            0            0
 25 - 29 :           0            0            0            0            0
 30 - 34 :           0            0            0            0            0
 35 - 39 :           0            0            0            0            0
 40 - 44 :           0            0            0            0            0
 45 - 49 :           0            0            0            0            0
 50 - 54 :           0            0            0            0            0
 55 - 59 :           0            0            0            0            0
 60 - 64 :           0            0            0            0
  cos: incoming
-------------------------------

  0 -  4 :        1115            0            0            0            0
  5 -  7 :           0            0            0
  cos: outgoing
-------------------------------

  0 -  4 :           5          658          612            0            0
  5 -  7 :           0            0            0
Policer: Inprofile:              0 OutofProfile:              0
```

Note that the number of CS1 and AF21 packet going to R6 is approximately the same. This is because the switch marks approximately half of the packets as exceeding and changes their DSCP values according to the mapping table.

Copyright © 2010 Internetwork Expert

10.69 Catalyst 3550 Per-Port Per-VLAN Policing

- Remove any previously attached service-policy on the port connected to the VLAN 146 interface of SW4
- Limit the rate of IP traffic from R4 on VLAN 146 to 128Kbps. Mark conforming packets as AF11 and exceeding as CS1
- Limit the rate of non-IP traffic from R4 on VLAN 146 to 256Kbps and drop the exceeding packets. Mark conforming packets with a CoS value of 3
- The configuration should not affect any future VLANs trunked to R4.

Configuration

```
SW4:
mls qos
mls qos map policed-dscp 10 to 8
!
! Acess-Lists for IP and non-IP traffic
!
ip access-list extended IP_ANY
 permit IP any any
!
mac access-list extended NON_IP_ANY
 permit any any 0x0 0xFFFF
!
! Second-level class-maps
!
class-map IP_ANY
 match access-group name IP_ANY
!
class-map NON_IP_ANY
 match access-group name NON_IP_ANY
!
! First-level class-maps
!
class-map VLAN_146_IP
 match vlan 146
 match class-map IP_ANY
!
class-map VLAN_146_NON_IP
 match vlan 146
 match class-map NON_IP_ANY
!
! Since we are not limited, we use any burst values
! Policy map matches first-level class-maps
!
policy-map POLICE_PER_PORT_PER_VLAN
 class VLAN_146_IP
   set dscp af11
   police 128000 16000 exceed-action policed-dscp-transmit
 class VLAN_146_NON_IP
   set dscp cs3
   police 256000 32000
```

Copyright © 2010 Internetwork Expert

```
!
! Apply the policy-map
!
interface FastEthernet 0/4
  no service-policy input POLICE_INBOUND
  service-policy input POLICE_PER_PORT_PER_VLAN
```

Verification

> **Note**
>
> Per-Port per-VLAN policing on the 3550 is very similar to the per-port per-VLAN classification model. You simply add policing commands under the 1st level class-maps in the policy map. The restrictions for per-VLAN classification syntax remain the same.
>
> For the purpose of verification, we will temporarily trunk another VLAN between R4 and SW1.

```
SW1, SW2, SW3, SW4:
vlan 47

SW1:
interface Vlan 47
 ip address 155.1.47.7 255.255.255.0

SW4:
interface FastEthernet 0/4
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport trunk allowed vlan 47,146

R4:
interface FastEthernet 0/1
 no ip address
!
interface FastEthernet 0/1.146
 encapsulation dot1q 146
 ip address 155.1.146.4 255.255.255.0
!
interface FastEthernet 0/1.47
 encapsulation dot1q 47
 ip address 155.1.47.4 255.255.255.0
```

Copyright © 2010 Internetwork Expert

We need to verify three things:

1) IP traffic on VLAN146 from R4 is marked with AF11 and remarked with CS1 when it exceeds the configured rate
2) non-IP traffic is marked as CS3 and dropped when it exceeds configured rate
3) Traffic on VLAN14 is not affected by the policing and classification process

To test the non-IP traffic we need to enable IPX on the interfaces of R4 and R6.

```
R4:
ipx routing
!
interface FastEthernet 0/1.146
 ipx network 146 encapsulation snap

R6:
ipx routing
!
interface FastEthernet 0/0.146
 ipx network 146 encapsulation snap
```

We start by verifying traffic on VLAN 47:

```
Rack1R4#ping 155.1.47.7 repeat 1000 size 1500

Type escape sequence to abort.
Sending 1000, 1500-byte ICMP Echos to 155.1.47.7, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Now configure R4 to source ICMP packets at rate of 256Kbps to R6. Also, enable DSCP monitoring in SW4 to watch packets marked with internal DSCP values of 10 (AF11) and other DSCP values used in the task:

```
SW4:
interface FastEthernet 0/1
 mls qos monitor packets
!
interface FastEthernet 0/4
 mls qos monitor dscp 0 8 10 24

R4:
interface FastEthernet 0/1.146
 traffic-shape rate 256000

Rack1SW4#clear mls qos interface fastEthernet 0/4 statistics

Rack1R4#ping 155.1.146.6 repeat 10000 size 1500 timeout 0

Type escape sequence to abort.
Sending 10000, 1500-byte ICMP Echos to 155.1.146.6, timeout is 0 seconds:
 ..........................................................................
 ..........................................................................
 ..........................................................................
 ..........................................................................
 ..........................................................................
<snip>

Rack1SW4#show mls qos interface fastEthernet 0/4 statistics
FastEthernet0/4
Ingress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 332        6          18         0          0
    8 : 0          0          0          0          0
    10: 0          0          331        153        0
    24: 0          0          0          0          0
Others: 23         0          0          0          0
Egress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 28           n/a        n/a      0          0
    8 : 0            n/a        n/a      0          0
    10: 0            n/a        n/a      0          0
    24: 0            n/a        n/a      0          0
Others: 57           n/a        n/a      0          0
```

Note that count of the “policed” packets is close to the half of “classified” packets. The latter are the packets we assigned DSCP to, and the former are the packets that have their DSCP marking changed by the policer.

Copyright © 2010 Internetwork Expert

The last thing to verify is checking the limit imposed on the IPX packets rate. Ensure you removed the policer on the VLAN146 interface of R4 before running the test:

```
Rack1R6#show ipx interface fastEthernet 0/0.146
FastEthernet0/0.146 is up, line protocol is up
  IPX address is 146.0012.d9ed.8c80, 802.1Q vLAN (SNAP) [up]
<snip>

R4:
interface FastEthernet 0/1.146
 no traffic-shape rate

Rack1R4#ping
Protocol [ip]: ipx
Target IPX address: 146.0012.d9ed.8c80
Repeat count [5]: 10000
Datagram size [100]: 1200
Timeout in seconds [2]: 1
Verbose [n]:
Type escape sequence to abort.
Sending 10000, 1200-byte IPX Novell Echoes to 146.0012.d9ed.8c80, timeout is 1
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!
!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!
!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.
Success rate is 96 percent (318/329), round-trip min/avg/max = 1/3/5 ms

Rack1SW4#show mls qos interface fastEthernet 0/4 statistics
FastEthernet0/4
Ingress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 2183       211         581        0          0
    8 : 0          0           0          0          0
   10 : 0          0           519        153        0
   24 : 0          0           329        0          11
Others: 774        0           0          0          0
Egress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 2215          n/a        n/a      0          0
    8 : 0             n/a        n/a      0          0
   10 : 0             n/a        n/a      0          0
   24 : 0             n/a        n/a      0          0
Others: 2047          n/a        n/a      0          0
```

Note the number of packets classified with internal DSCP value of 24 (CS3) and the number of dropped packets.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.70 Catalyst QoS Aggregate Policers

- Limit the aggregate rate of ICMP and IPX traffic coming from R1 on SW1 to 128Kbps, and drop exceeding packets.
- Trunk a new VLAN 47 to the port of R4 connected to SW4.
- Limit the aggregate rate of IP traffic to 256Kbps in both VLAN 146 and the new VLAN 47. Mark conforming packets with DSCP AF11 and remark packets to CS1 if they exceed.

Configuration

```
SW1:
mls qos
!
! Access-lists for ICMP pakets
!
ip access-list extended ICMP
 permit icmp any any
!
! Access-list for IPX packets
!
mac access-list extended IPX
 permit any any 0x8137 0x0
!
! Class-maps for traffic flows
!
class-map ICMP
 match access-group name ICMP
!
class-map IPX
 match access-group name IPX
!
! Aggregate policer
!
mls qos aggregate-policer AGG128 128000 16000 exceed-action drop
!
! Policy-map that uses the aggregate policer
!
policy-map POLICE_AGGREGATE
 class ICMP
  police aggregate AGG128
 class IPX
  police aggregate AGG128
 class class-default
  set dscp cs1
!
interface FastEthernet 0/1
 service-policy input POLICE_AGGREGATE
```

Copyright © 2010 Internetwork Expert

```
SW4:
!
! IP traffic on the new VLAN (first level class-map)
! Access-lists and second level class-maps
! have been already created
!
class-map VLAN_47_IP
 match vlan 47
 match class-map IP_ANY
!
mls qos map policed-dscp 10 to 8
!
mls qos aggregate-policer AGG256 256000 32000 exceed-action policed-
dscp-transmit
!
! Since we are not limited, we may use any burst values
! Policy map matches first-level class-maps as usual
!
policy-map POLICE_PER_PORT_PER_VLAN
 class VLAN_146_IP
   set dscp af11
   no police 128000 16000 exceed-action policed-dscp-transmit
   police aggregate AGG256
 class VLAN_47_IP
   set dscp af11
   police aggregate AGG256
!
! Apply the policy-map
!
interface FastEthernet 0/4
  switchport trunk allowed vlan add 47
  no service-policy input POLICE_PER_PORT_PER_VLAN
  service-policy input POLICE_PER_PORT_PER_VLAN
```

Verification

Note

An aggregate policer applies to multiple traffic classes on the same interface. You define the aggregate policer rate, burst, and action globally using a unique name. Later, you apply this policer under the respective classes of a policy-map. All classes configured with the same aggregate policer share the configured rate.

The restrictions are that you cannot aggregate metered rate across different physical ports. In the 3560 model it is impossible to insert an aggregate policer inside a second-level policy map. Thus, you may *only* aggregate the bitrate between VLANs on the same interface if you are using the 3550 model.

Copyright © 2010 Internetwork Expert

For the purpose of verification, we will configure IP addressing in VLAN 47 on SW1 and R4:

```
SW1, SW2, SW3, SW4:
vlan 47

SW1:
interface Vlan 47
 ip address 155.1.47.7 255.255.255.0

SW4:
interface FastEthernet 0/4
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport trunk allowed vlan 47,146

R4:
interface FastEthernet 0/1
 no ip address
!
interface FastEthernet 0/1.146
 encapsulation dot1q 146
 ip address 155.1.146.4 255.255.255.0
!
interface FastEthernet 0/1.47
 encapsulation dot1q 47
 ip address 155.1.47.4 255.255.255.0
```

Now start transferring a large file from R4 across both VLANs 146 and 47.

```
R4:
ip http server
ip http path flash:

Rack1R1#$copy http:/admin:cisco@155.1.146.4/c1841-adventerprisek9-
mz.124-24.T.bin null:
Loading http://***********@155.1.146.4/c1841-adventerprisek9-mz.124-
24.T.bin
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!

Rack1SW1#copy http:/admin:cisco@155.1.47.4/c1841-adventerprisek9-
mz.124-24.T.bin null:
Loading http://admin:cisco@155.1.47.4/c1841-adventerprisek9-mz.124-
24.T.bin !!!
```

Copyright © 2010 Internetwork Expert

Wait a few minutes for transfer rates to stabilize. Now collect input and output rate statistics as shown below (note that all load-intervals have value of 30 seconds):

```
Rack1R1#show interfaces fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
<snip>
  30 second input rate 125000 bits/sec, 16 packets/sec
  30 second output rate 11000 bits/sec, 22 packets/sec

Rack1SW1#show interfaces vlan47
Vlan47 is up, line protocol is up
<snip>
  30 second input rate 142000 bits/sec, 18 packets/sec
  30 second output rate 14000 bits/sec, 27 packets/sec
<snip>

Rack1R4#show interface fastEthernet 0/1
FastEthernet0/1 is up, line protocol is up
<snip>
  30 second input rate 25000 bits/sec, 48 packets/sec
  30 second output rate 340000 bits/sec, 38 packets/sec
<snip>

Rack1SW4#show mls qos interface fastEthernet 0/4 statistics
FastEthernet0/4
Ingress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 8645       27         50         0          0
    8 : 0          0          0          0          0
    10: 0          0          8667       0          1693
    24: 0          0          0          0          0
Others: 99         0          0          0          0
Egress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    0 : 10647        n/a        n/a      0          0
    8 : 0            n/a        n/a      0          0
    10: 0            n/a        n/a      0          0
    24: 0            n/a        n/a      0          0
Others: 269          n/a        n/a      0          0
```

Copyright © 2010 Internetwork Expert

Calculate the following:

R1_Input_Rate + SW1_Input_Rate = 142000+125000 = 267000
R4_Output_Rate = 340000
Packet_Drop_Rate=1693/8667*100%=20%

If you multiply *R4_Output_Rate* by *Packet_Drop_Rate* this value will be close to *R4_OutputRate – R1_Input_Rate – SW1_Input_Rate*. The conclusion is that switch limits aggregate rate for traffic from both VLANs.

Now we need to ensure the traffic rate limit imposed on IP and non-IP packets from R1. To verify that, send a flood of IPX and ICMP packets from R1 towards R6. Configure R1 and R6 for IPX packet exchange first:

```
R1:
ipx routing
!
interface FastEthernet 0/0
 ipx network 146 encapsulation snap

R6:
ipx routing
!
interface FastEthernet 0/0.146
 ipx network 146 encapsulation snap
```

Now generate two traffic flows from R1: one with ICMP packets and the other one with IPX packets.

```
Rack1R1#ping 155.1.146.6 repeat 1000000 size 1000 timeout 0

Type escape sequence to abort.
Sending 1000000, 1000-byte ICMP Echos to 155.1.146.6, timeout is 0 seconds:
 ............................................................
 ............................................................
 ............................................................
 ............................................................
 ............................................................

Rack1R6#show ipx interface fastEthernet 0/0.146
FastEthernet0/0.146 is up, line protocol is up
  IPX address is 146.0011.2042.55e0, 802.1Q vLAN (SNAP) [up]
```

```
Rack1R1#ping
Protocol [ip]: ipx
Target IPX address: 146.0011.2042.55e0
Repeat count [5]: 10000000
Datagram size [100]: 1000
Timeout in seconds [2]: 0
Verbose [n]:
Type escape sequence to abort.
Sending 10000000, 1000-byte IPX Novell Echoes to 146.0011.2042.55e0, timeout is
0 seconds:
 ....................
 ....................
 ....................
 ....................
 ....................
 ....................
 ....................
 ....................
```

Check the output packet rate on the connection to R6 and the policer statistics on connection to R1:

```
Rack1SW2#show interfaces fastEthernet 0/6
FastEthernet0/6 is up, line protocol is up (connected)
  Hardware is Fast Ethernet, address is 001b.d4df.c308 (bia 001b.d4df.c308)
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex, 100Mb/s, media type is 10/100BaseTX
  input flow-control is off, output flow-control is unsupported
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:42, output 00:00:01, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  30 second input rate 124000 bits/sec, 15 packets/sec
  30 second output rate 125000 bits/sec, 16 packets/sec
<snip>
```

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show mls qos interface fastEthernet 0/1 statistics
FastEthernet0/1

<snip>

  cos: incoming
-------------------------------

  0 -  4 :      429084            0            0            0            0
  5 -  7 :           0            0            0
  cos: outgoing
-------------------------------

  0 -  4 :       16289            0            0            0            0
  5 -  7 :           0            1            1
Policer: Inprofile:        12527 OutofProfile:       372720
```

We can see that SW1 limits the aggregate input rate to a value close to the target 128Kbps by dropping the exceeding packets in both classes.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

> **Note**
>
> Load the *QoS* initial configurations prior to starting.

10.71 Catalyst 3560 Ingress Queueing

- Enable MLS QoS on SW1 and trust IP precedence on the port connected to R1.
- Configure ingress queue 1 as priority queue with 15% of ring bandwidth.
- Map all packets marked with CoS values of 5 to the priority queue.
- Ensure all other packets map to the non-priority queue.
- The bandwidth remaining after the priority queue should be shared between ingress queues in proportions 1:3 for priority and non-priority queues respectively.

Configuration

```
SW1:
mls qos
!
! Set queue 1 as priority queue
! Set the bandwidth limit to 15%
!
mls qos srr-queue input priority-queue 1 bandwidth 15
!
! SRR weights for input queue 1 and 2
!
mls qos srr-queue input bandwidth 10 30
!
! CoS to Queue-ID mapping. Queue ID is followed by CoS list
!
mls qos srr-queue input cos-map queue 1 5
mls qos srr-queue input cos-map queue 2 1 2 3 4 6 7
!
! Trust precedence on the port connected to R1
!
interface FastEthernet 0/1
 mls qos trust ip-precedence
```

Copyright © 2010 Internetwork Expert

Verification

Note

Ingress queues are unique to the 3560 switches due to their internal ring architecture. There are two ingress queues acting as a "shim' layer between all ports and the internal ring. The switch queues input traffic after the classification and marking stage and uses the QoS label assigned to a packet to map it to the respective input queue, if the internal backplane experiences congestion. There are exist two separate mapping tables: CoS to Queue-ID and DSCP to Queue-ID. Depending on the value you trusted or set on ingress (DSCP, IP Precedence or CoS) the switch selects the respective mapping table. You may learn the existing mappings using the commands **`show mls qos maps dscp-input-q`** or **`show mls qos maps cos-input-q`**.

All ingress queue settings are global. In this task, we configure mappings for ingress queues as well as queue weights. The queue weights define the logic of input packet scheduler. Input packet scheduler is a variation of Round Robin (RR) procedure called Shared Round Robin or SRR. It has some additional modifications over classic Round Robin to increase accuracy and swiching performance. The documentation is really scarce on this topic, but the logic is as follows:

1) You may configure one of the queues (Queue 1 or Queue 2) as priority.
2) If queue is a priority queue, you assign it a bandwith threshold value expressed in percents ranging from 0% to 40%. Those percents define the amount of *internal ring bandwidth* available to priority queue.
3) Both Queue 1 and Queue 2 has additional SRR weights, used by the scheduler.

The scheduling process works as follows, every time quantum:

a) If a queue is the priority queue, service it in exclusive mode up to the amount of packets enough not to fill the intenal ring over the configured threshold.
b) If the scheduler reaches the threshold for the priority queue, it stops servicing it in exclusive mode.
c) At that point the scheduler uses the weighted round robin procedure between Queue 1 and Queue 2, sharing the remaining bandwidth between queues proportional to their SRR weights.
d) If there is no priority queue defined, service both queues using SRR weighted round robin logic.

Copyright © 2010 Internetwork Expert

The general logic with priority queue mode is to service the priority queue up to the maximum rate but then schedule all exceeding packets using "fair" round robin scheduling. This is similar to the process used by RSVP policer, which does not drop exceeding flow traffic but rather services it as best-effort using lowest WFQ weight.

The actual implementation of SRR scheduler is not important to us. Rather, the general idea of SRR is that "n" number of queues of weights W(1)…W(n) will have shares of bandwidth proportional to

Share(i)=W(i)/(W(1)+...W(n)) for queue "i".

Therefore, the absolute values of SRR weight do no matter, but only their proportions (relative values). The actual implemtnation may use those weights as extra credits added to deficit counter of every queue.

Consider the example where Queue 1 and Queue 2 are set with SRR weights "10" and "20" while Queue 2 is in priority mode with bandwidth limit of 20%. Assuming that internal ring bandwidth is 10Gbps we get the following:

1) Queue1 may send traffic up to 20%*10Gbps=2Gbps.
2) The remaining bandwidth (8Gbps) is shared in proportions 10:20 between the exceeding packets from Queue 1 and all packets from Queue 2.

Thus in result Queue 1 has guaranteed 8Gbps*10/(10+20)=2.6Gbps of ring bandwidth and Queue 2 has guranteed 2Gbps+8Gbps*20(10+20)=7.3Gbps of ring bandwidth. At the same time, if Queue 2 traffic does not exceed 2Gbps the scheduler treats it as priority.

As another example, consider both queues configured in regular mode with SRR weights of 33 and 66. Assuming that internal ring bandwidth is 5Gbps, we conclude that Queue 1 guaranteed bandwidth is 1.65Gbps and Queue 2 guaranteed bandwidth is 3.3Gbps.

As usual with max-min sharing procedure, if any queue does not use its minimum guarantee bandwidth, the other queue may use the remaining bandwidth.

In the configuration above, we trust IP precedence in the incoming packets. This will automatically enable trusting CoS in non-IP packets. In both cases, the switch deduces the equivalent CoS value using the DSCP to CoS mapping table. Thus we only map CoS values to the respective queues, performing "aggregate".

Copyright © 2010 Internetwork Expert

There is no easy way to verify the actual working of ingress queues unless you really congest the switch. Therefore, we will limit verifications just to some show commands.

```
Rack1SW1#show mls qos input-queue
Queue     :       1       2
-------------------------------------------
buffers   :      90      10
bandwidth :      10      30
priority  :      15       0
threshold1:     100     100
threshold2:     100     100

Rack1SW2#show mls qos maps dscp-input-q
   Dscp-inputq-threshold map:
     d1 :d2    0     1     2     3     4     5     6     7     8     9
     ------------------------------------------------------------------
      0 :    02-01 01-01 01-01 01-01 01-01 01-01 01-01 01-01 02-01 01-01
      1 :    01-01 01-01 01-01 01-01 01-01 01-01 02-01 01-01 01-01 01-01
      2 :    01-01 01-01 01-01 01-01 02-01 01-01 01-01 01-01 01-01 01-01
      3 :    01-01 01-01 02-01 01-01 01-01 01-01 01-01 01-01 01-01 01-01
      4 :    01-01 02-01 02-01 02-01 02-01 02-01 02-01 02-01 02-01 01-01
      5 :    01-01 01-01 01-01 01-01 01-01 01-01 02-01 01-01 01-01 01-01
      6 :    01-01 01-01 01-01 01-01

Rack1SW2#show mls qos maps cos-input-q
   Cos-inputq-threshold map:
              cos:  0   1   2   3   4   5   6   7
              ------------------------------------
  queue-threshold: 1-1 2-1 2-1 2-1 2-1 1-1 2-1 1-1
```

As you can see, the SRR weights are 10 and 30, while priority bandwidth for queue 1 is set to 15% of the internal ring bandwidth.

The mapping tables are for DSCP and CoS values. In the DSCP to queue-threshold mapping table you see DSCP values split in rows and columns. Rows are for the first digits of DSCP codepoints and columns are for the second digits. There are 64 DSCP codepoints and the table is partitioned accordingly. In the cells, you can see “queue_id-theshold_id” pairs, which specify the queue number and the threshold the value maps to. Note that DSCPs corresponding to IP Precedence 6 and 7 map to queue 2, not the default queue 1. We will discuss thresholds in more details in the next task.

Copyright © 2010 Internetwork Expert

10.72 Catalyst 3560 Ingress Queue Tuning

- Configure drop thresholds on SW1 for the non-priority ingress queue to 80% and 100%.
- Ensure that routing and network control traffic mapped to queue 2 is less likely to be dropped than regular traffic.
- Ensure that the queue 2 buffer space is 3 times larger than the priority queue's size.

Configuration

```
SW1:
mls qos
!
! Change the threholds for Queue 2. Thresholds are in percents
! Queue ID is folloed by two threshold values
!
mls qos srr-queue input threshold 2 80 100
!
! Map CoS corresponding to network control and IP routing
! traffic to Threshold 2
!
mls qos srr-queue input cos-map threshold 2 6 7
!
! Share buffers in proportions 1:3 between Q1 and Q2
!
mls qos srr-queue input buffers 25 75
```

Verification

Note

Each input queue has the following configurable parameters:

1) Drop thresholds. There are *three* thresholds per queue, and by default, they are set to 100% of the queue size. You can only set thresholds in percents, not the absolute values. Note that the third threshold value is locked to "queue full" state or 100% of queue size and *cannot* be changed. You can still map DSCP and CoS values to the third threshold.

2) Buffer space partitioning. Both queues share the same buffer space, and you can divide it using relative weights expressed in percents. By default, first queue has 90% of buffer space and the second queue has 10% of queue space.

Copyright © 2010 Internetwork Expert

The drop strategy used by SRR is known as WTD – weighted tail drop. That is, every queue has a number of drop thresholds (three in the 3560) and you map codepoints to these values. For example, you may want to map voice or control packets to the maximum drop threshold and map less priority traffic to a lower threshold. Per the default configuration, all codepoints map to the first drop threshold, which is set to 100% of the queue size.

Note that you can map DSCP and CoS values to threshold in two different ways: using commands:

```
mls qos srr-queue input {dscp-map|cos-map} threshold <t-id>
<List>
```

And

```
mls qos srr-queue input {dscp-map|cos-map} queue <q-id>
threshold <t-id> <List>.
```

The first form seems to be deprectated as if you enter the first command, the CLI will automatically convert it to the second format.

Copyright © 2010 Internetwork Expert

Use the following show commands to display your new settings:

```
Rack1SW1#show mls qos input-queue
Queue     :       1       2
-------------------------------------------
buffers   :      25      75
bandwidth :      10      30
priority  :      15       0
threshold1:     100      80
threshold2:     100     100

Rack1SW1#show mls qos maps dscp-input-q
   Dscp-inputq-threshold map:
     d1 :d2    0     1     2     3     4     5     6     7     8     9
     ------------------------------------------------------------
      0 :    02-01 01-01 01-01 01-01 01-01 01-01 01-01 01-01 02-01 01-01
      1 :    01-01 01-01 01-01 01-01 01-01 01-01 02-01 01-01 01-01 01-01
      2 :    01-01 01-01 01-01 01-01 02-01 01-01 01-01 01-01 01-01 01-01
      3 :    01-01 01-01 02-01 01-01 01-01 01-01 01-01 01-01 01-01 01-01
      4 :    01-01 02-01 02-01 02-01 02-01 02-01 02-01 02-01 02-02 01-01
      5 :    01-01 01-01 01-01 01-01 01-01 01-01 02-02 01-01 01-01 01-01
      6 :    01-01 01-01 01-01 01-01

Rack1SW1#show mls qos maps cos-input-q
   Cos-inputq-threshold map:
              cos:  0   1   2   3   4   5   6   7
              ------------------------------------
  queue-threshold: 1-1 2-1 2-1 2-1 2-1 1-1 2-2 1-2
```

You can see the new buffer settings, and the new queue-id and threshold mappings for DSCP and CoS values.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.73 Catalyst 3550 Egress Queueing

- Network traffic consists of the following traffic classes:
  - VoIP bearer packets marked with DSCP EF
  - Interactive video traffic marked with DSCP AF41
  - VoIP signaling traffic marked with DSCP CS3
  - Interactive Citrix traffic marked with DSCP AF21
  - Best-effort traffic marked with DSCP 0
- Implement the following traffic policy on SW4's connection to R4:
  - Provide priority treatment for VoIP bearer packets
  - Provide a separate queue for video traffic and guarantee it 30% of the remaining bandwidth
  - Provide a separate queue for interactive traffic and VoIP signaling with 20% of the bandwidth
  - The remaining bandwidth should be used by best-effort traffic
- Ensure no other traffic but voice can use the priority queue

Configuration

```
SW4:
mls qos
!
interface FastEthernet 0/4
 wrr-queue bandwidth 50 20 30 1
 wrr-queue cos-map 4 5
 wrr-queue cos-map 3 4 6 7
 wrr-queue cos-map 2 2 3
 wrr-queue cos-map 1 0 1
 priority-queue out
```

Copyright © 2010 Internetwork Expert

Verification

Note

The 3550 switches have two types of ports: Gigabit and FastEthernet, and they use different ASICs. As we see later this allows for more features on Gigabit ports. However, both types use the same queuing scheduler known as WRR (Weighted Round Robin). It is a modification of well-known Round Robin procedure, which is similar to SRR used in the 3560 but has different implementation. So far, we only know that SRR is more effective than WRR.

WRR works with four egress queues per port. That means you can define up to four traffic classes and define bandwidth sharing between them. Compare this to dynamic queue capabilites of WFQ, where you can have up to 4096 flows of traffic or CBWFQ where you can define 64 classes. Obviously, this is the limitation of highly optimized hardware-based QoS.

Each of the four egress queues has numerical weight values assigned. The WRR scheduler uses these values to share the bandwidth between classes. The absolute values of the weights are not important, for the bandwidth share of class "j" is

S(j)=W(j)/(W(1)+...W(4)),

where W(1),...,W(4) are the weight values of all four classes. If you are interested in the actualy implementation, it seems that the weight values are used to increment the deficit round robin counter by an amount of tokens proportional to the weight value. Thus, setting larger absolute weight values may slightly increase packet latency, though is hardly possible on high-speed interfaces. You can set the WRR weight for a queue using the interface-level command **`wrr-queue bandwidth <w1> <w2> <w3> <w4>`** where **`<wj>`** is the weight for queue "j". Remember that different absolute weight values may result in the same relative shares of bandwidth.

It is possible to designate queue 4 as priority queue on the 3550 switch. In this case, WRR scheduler always empties this queue first. The danger is that there is no restriction on the priority queue rate, and you must ensure the use of ingress policing for admisstion control on the priority traffic. When you enable priority-queue services using the interface-level command **`priority-queue out`** the weight assigned to queue 4 with the command **`wrr-queue bandwidth`** does not play any role for the WRR scheduler and you may set it to zero when calculating bandwidth share.

Copyright © 2010 Internetwork Expert

For example, consider the following situation. The interface rate is 10Mbps and WRR weights are 10 20 30 40 for queues from 1 to 4. The interface uses a priority queue. Imagine that voice traffic (priority queue) rate is 1Mbps (that's way too much, but it's an imaginary situation). In this case, the minimum bandwidth guarantees are:

Queue 1: 9Mbps*10/(10+20+30+0)=1/6*9Mbs=1.5 Mbps.
Queue 2: 9Mbps*20/(10+20+30+0)=3 Mbps.
Queue 3: 9Mbps*30/(10+20+30+0)=4.5 Mbps.

In case if priority traffic is not present, the extra 1Mbps could be shared in the same proportions between the remaining queues.

The last question is mapping packets to queues. In the 3550 model you can only do that based on the CoS values. However, as we remember, the 3550 assigns internal DSCP value for all classified packets. In order to come up with CoS value for a packet, the switch consults the DSCP to CoS mapping table. It is important to note, that if the switch policed down the internal DSCP, it uses the new, policed value for mapping. The command to map a CoS to Queue-ID is interface level command **`wrr-queue cos-map <q-id> <CoS List>`**. You may find out the current mappings using the command **`show mls qos interface queueing`**.

In our case, we use the default DSCP to CoS mapping table, which results in the following mappings:

DSCP EF -> CoS 5
DSCP AF41 -> CoS 4
DSCP CS3 -> CoS 3
DSCP AF21 -> CoS 2
DSCP 0 - > CoS 0

We configure the fourth queue as the priority queue and map CoS 5 to it. Next we map CoS 4 to queue 3 and both CoS 2 and CoS 3 to queue 2. Finally, CoS 0 is mapped to Queue 1 by default. The bandwidth weight values are easy to find from the bandwidth percentage: 30% for video traffic, 20% for interactive and voice signaling and 50% for best-effort traffic. Assign any WRR weight value to queue 4 since it is not used when the priority queue is engaged.

Copyright © 2010 Internetwork Expert

We also need to deal with the control plane traffic, which usually bears CoS values of 6 or 7. For our scenario we'll just map them to the queue used by video traffic. Use the following show commands to display the new mappings and WRR weights:

```
Rack1SW4#show mls qos interface fastEthernet 0/4 queueing
FastEthernet0/4
Egress expedite queue: ena
wrr bandwidth weights:
qid-weights
 1 - 50
 2 - 20
 3 - 30
 4 - 1    when expedite queue is disabled
Cos-queue map:
cos-qid
 0 - 1
 1 - 1
 2 - 2
 3 - 2
 4 - 3
 5 - 4
 6 - 3
 7 - 3
```

Now let's run a stress test for WRR. We will configure the connection to R4 for a speed of 10Mbps and send a barrage of ICMP packets from R6 and R1 to saturate the link. One of the flows will be marked with DSCP AF21 and the other will use the default DSCP 0. Per the WRR algorithm, the bandwidth should be shared in proportions of 50:20 between DSCP 0 and DSCP AF21 (ToS byte 72) traffic. The resulting bandwidth values should be

50/(20+50)*10Mbps = 7.1 Mbps for DSCP 0 traffic
20/(20+50)*10Mbps = 2.9 Mbps for DSCP AF21 traffic.

SW4 should trust DSCP marking for packets coming from other switches. R4 should block ICMP packets from getting to the CPU and meter the input rate for DSCP AF21 packets.

Copyright © 2010 Internetwork Expert

```
SW4:
interface range FastEthernet 0/13 - 21
 mls qos trust dscp
         !
R4:
class-map DSCP_AF21
 match dscp af21
!
policy-map METER
 class DSCP_AF21
!
ip access-list extended FILTER
 deny icmp any any
 permit ip any any
!
interface FastEthernet 0/1
 ip access-group FILTER in
 service-policy input METER
 load-interval 30

Rack1R6#ping
Protocol [ip]:
Target IP address: 155.1.146.4
Repeat count [5]: 100000000
Datagram size [100]: 1500
Timeout in seconds [2]: 0
Extended commands [n]: y
Source address or interface:
Type of service [0]: 72
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100000000, 1500-byte ICMP Echos to 155.1.146.4, timeout is 0
seconds:
 ............................................................
 ............................................................
 ............................................................

Rack1R1#ping 155.1.146.4 repeat 1000000 size 1500 timeout 0

Type escape sequence to abort.
Sending 1000000, 1500-byte ICMP Echos to 155.1.146.4, timeout is 0
seconds:
 ............................................................
 ............................................................
 ............................................................
 ............................................................
 ............................................................
```

Verify the statistics accumulated at R4 and note how close it is to the values calculated above:

```
Rack1R4#show policy-map interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy input: METER

    Class-map: DSCP_AF21 (match-all)
      41104 packets, 58211273 bytes
      30 second offered rate 2730000 bps
      Match:  dscp af21 (18)

    Class-map: class-default (match-any)
      228668 packets, 346115656 bytes
      30 second offered rate 7104000 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

10.74 Catalyst 3550 Regular Queues Tuning

- Configure SW4 to set the global min-reserve levels 7 and 8 to 170 (max) and 10 (min) buffers respectively.
- Under the interface connected to R4 map global level 7 to queue 1 and global level 8 to queue 4.

Configuration

```
SW4:
mls qos min-reserve 7 170
mls qos min-reserve 8 10
!
interface FastEthernet 0/4
 wrr-queue min-reserve 1 7
 wrr-queue min-reserve 4 8
```

Verification

Note

As mentioned earlier, the 3550 makes a clear distinction between FastEthernet and GigabitEthernet ports. The buffer space allocation for regular FastEthernet ports differs significantly from the buffer allocation scheme for GigabitEthernet Port.

Every queue on a regular interface has its own buffer limit. You cannot change it for every specific queue on any interface. Rather, there are eight global variables knows as “minimal reserve levels”. Each of these values is the number of packet buffers. You can map the global constants to a queue-id on each interface.

By default, queues 1 through 4 on every interface use global minimal reserve levels 1 through 4 as well. Initially, all eight levels have value of 100 buffers. The maximum level size is 170 buffers and the minimum size is 10. The buffer tuning commands for regular interfaces are as follows:

Global command defines values any of 8 levels

```
mls qos min-reserve <level> <size>
```

Interface-level command:

```
wrr-queue min-reserve <q-id> <level>
```

Copyright © 2010 Internetwork Expert

Note that you cannot change the drop policy on regular interface. It's fixed to tail drop, with no advanced features. For this task, we should tune the levels that are not used by default, e.g. levels 7 and 8. Use the following show command to verify your settings:

```
Rack1SW4#show mls qos interface fastEthernet 0/4 buffers
FastEthernet0/4
Minimum reserve buffer size:
 100 100 100 100 100 100 170 10
Minimum reserve buffer level select:
 7 2 3 8
```

Here you can see the minimum reserve levels and their assignment for the given interface.

Copyright © 2010 Internetwork Expert

10.75 Catalyst 3550 Gigabit Interface Queues Tuning

- Configure the first GigbitEthernet interface on SW4 as follows:
  - Set thresholds to 50 and 100 for all four queues
  - Map DSCP values 46, 48, and 56 to threshold 2
  - Set WRR queue weights to 20, 20, 20 and 40 for queues 1 through 4
- Enable WRED on the second GigabitEthernet interface of SW4 and configure as following:
  - Set WRED minimum thresholds to 50 and 80 for all queues
  - Map DSCP values 46, 48 and 56 to WRED threshold 2
  - Set WRR queue weights to 20, 20, 20 and 40 for queues 1 through 4

Configuration

```
SW4:
!
! Use WTD on the first interface
!
interface GigabitEthernet0/1
 wrr-queue threshold 1 50 100
 wrr-queue threshold 2 50 100
 wrr-queue threshold 3 50 100
 wrr-queue threshold 4 50 100
 wrr-queue dscp-map 2 46 48 56
 wrr-queue queue-limit 20 20 20 40
!
! Use WRED on the second interface
!
interface GigabitEthernet0/2
 wrr-queue random-detect max-threshold 1 50 80
 wrr-queue random-detect max-threshold 2 50 80
 wrr-queue random-detect max-threshold 3 50 80
 wrr-queue random-detect max-threshold 4 50 80
 wrr-queue dscp-map 2 46 48 56
 wrr-queue queue-limit 20 20 20 40
```

Copyright © 2010 Internetwork Expert

Verification

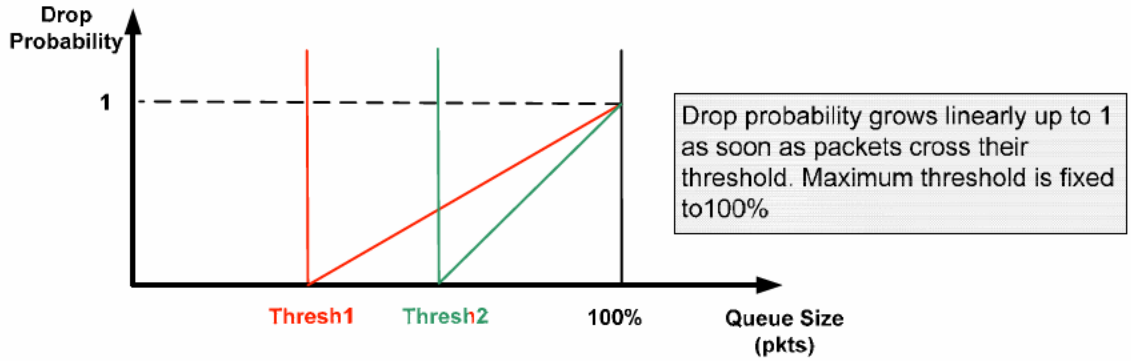
Note

GigabitEthernet ports support two queue management techniques. The first one is similar to the 3560 WTD (Weighted Tail Drop). Each of the four queues has two thresholds, expressed in percents. By default, both thresholds are set to 100%, simulating tail drop. You can tune the thresholds per-queue and map internal DSCP values to any of the thresholds. Thus, packets with different internal DSCP values have different chances of being dropped in the output queue. By default all DSCP values map to the first threshold.

The interface-level commands to set WTD thresholds are as follows:

```
wrr-queue threshold <q-id> <thresh1> <thresh2>
wrr-queue dscp-map <thresh> <DSCP List>
```

The second drop mode for any queue on GigabitEthernet interface is WRED. You can enable WRED per-queue and set two thresholds where the system starts dropping packets. Unlike the classic WRED found in routers, this WRED implementation does not allow you to set the maximum threshold. It is fixed to 100%. However, you can map DSCP values to minimum (starting) thresholds, and thus provide differentiated drop behavior.



On the diagram, you see two different thresholds and the graph of dropping probability for each starting value. There is no Marking Probability like the one you see in IOS WRED, not there is average queue size weighting parameter. By default all DSCP values maps to threshold 1.

Copyright © 2010 Internetwork Expert

The commands to configure WRED in the 3550 are as follows:

```
 wrr-queue random-detect max-threshold <q-id> <thresh1>
<thresh2>
 wrr-queue dscp-map <thresh> <DSCP List>.
```

You can see that the threshold mapping is shared by both queue modes. The default queue mode for gigabit interface is WTD with both thresholds set to 100%.

Lasty, the buffer space allocation for the Gigabit ports is also different. Each port has common buffer space shared between all queues. You can not specify the number of buffers available to each specific queue but you can set the weighted part of shared space allocated to each queue. The command to share the buffers is:

```
wrr-queue queue-limit <w1> <w2> <w3> <w4>
```

Queue "j" will get the share of buffer space equals to

s(j)=w(j)/(w(1)+...w(4).

Try to configure the weights so that they sum to 100, so that you can interpret weights as percents of buffer pool size.
Use the following show command to verify your new settings for both ports.

Copyright © 2010 Internetwork Expert

For the first port, notice how the DSCP values map to the thresholds and new threshold values.

```
Rack1SW4#show mls qos interface gigabitEthernet 0/1 queueing
GigabitEthernet0/1
Egress expedite queue: dis
wrr bandwidth weights:
qid-weights
 1 - 25
 2 - 25
 3 - 25
 4 - 25
Dscp-threshold map:
     d1 :  d2 0  1  2  3  4  5  6  7  8  9
     ----------------------------------------
      0 :    01 01 01 01 01 01 01 01 01 01
      1 :    01 01 01 01 01 01 01 01 01 01
      2 :    01 01 01 01 01 01 01 01 01 01
      3 :    01 01 01 01 01 01 01 01 01 01
      4 :    01 01 01 01 01 01 02 01 02 01
      5 :    01 01 01 01 01 01 02 01 01 01
      6 :    01 01 01 01
Cos-queue map:
cos-qid
 0 - 1
 1 - 1
 2 - 2
 3 - 2
 4 - 3
 5 - 3
 6 - 4
 7 - 4

Rack1SW4#show mls qos interface gigabitEthernet 0/1 buffers
GigabitEthernet0/1
Notify Q depth:
qid-size
 1 - 20
 2 - 20
 3 - 20
 4 - 40
qid WRED thresh1 thresh2
1   dis  50      100
2   dis  50      100
3   dis  50      100
4   dis  50      100
```

Copyright © 2010 Internetwork Expert

For the second port, notice that WRED is the drop mode for all four queues.

```
Rack1SW4#show mls qos interface gigabitEthernet 0/2 queueing
GigabitEthernet0/2
Egress expedite queue: dis
wrr bandwidth weights:
qid-weights
 1 - 25
 2 - 25
 3 - 25
 4 - 25
Dscp-threshold map:
     d1 :  d2 0  1  2  3  4  5  6  7  8  9
     ----------------------------------------
      0 :    01 01 01 01 01 01 01 01 01 01
      1 :    01 01 01 01 01 01 01 01 01 01
      2 :    01 01 01 01 01 01 01 01 01 01
      3 :    01 01 01 01 01 01 01 01 01 01
      4 :    01 01 01 01 01 01 02 01 02 01
      5 :    01 01 01 01 01 01 02 01 01 01
      6 :    01 01 01 01
Cos-queue map:
cos-qid
 0 - 1
 1 - 1
 2 - 2
 3 - 2
 4 - 3
 5 - 3
 6 - 4
 7 - 4

Rack1SW4#show mls qos interface gigabitEthernet 0/2 buffers
GigabitEthernet0/2
Notify Q depth:
qid-size
 1 - 20
 2 - 20
 3 - 20
 4 - 40
qid WRED thresh1 thresh2
1   ena  50      80
2   ena  50      80
3   ena  50      80
4   ena  50      80
```

Copyright © 2010 Internetwork Expert

10.76 Catalyst 3550 Egress Policing

- Enable the expedite queue on SW4's connection to R4.
- Ensure only CoS 5 maps to this queue.
- Limit the rate of the expedite queue to 128Kbps with the burst size of 8000 bytes.
- Limit the rate of traffic marked with CS0 to 384Kbps using a burst size of 32000 bytes for metering.

Configuration

```
SW4:
mls qos
!
class-map DSCP_EF
 match ip dscp ef
!
class-map DSCP_CS0
 match ip dscp 0
!
policy-map EGRESS_POLICING
 class DSCP_EF
  police 128000 8000
 class DSCP_CS0
  police 384000 32000
!
interface FastEthernet 0/4
 priority-queue out
!
! Ensure that CoS 6 and 7 map to non-priority queue
!
 wrr-queue cos-map 3 6 7
 service-policy output EGRESS_POLICING
```

Verification

Note

The egress policing feature is unique to the 3550. You may configure up to eight egress policers on any port. However, the limitation is that you can only match DSCP values as the policer classification criterion. The DSCP values are the internal DSCP and thus apply both to IP and non-IP traffic.

Copyright © 2010 Internetwork Expert

For verification, configure SW4 to trust DSCP values on all trunk ports and send two ping floods from R6 towards R4.

```
SW4:
interface range FastEthernet 0/13 - 21
 mls qos trust dscp

Rack1R6#ping
Protocol [ip]:
Target IP address: 155.1.146.4
Repeat count [5]: 100000
Datagram size [100]: 1500
Timeout in seconds [2]: 1
Extended commands [n]: y
Source address or interface:
Type of service [0]: 184
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100000, 1500-byte ICMP Echos to 155.1.146.4, timeout is 1
seconds:
!!!!!!!!!!!!.!!!!!!!!!!!.!!!!!!!!!!!.!!!!!!!!!!!.!!!!!!!!!!!.!!!!!!!!!
!!!.!!!!!!!!!!!.!!!!!!!!!!!.!!!!!!!!!!!.!!!!!!!!!!!.!!!!!!!!!!!!.!!!!!
!!!!!!.!!!!!!!!!!!.!!!!!!!!!!!.!!!!!!!!!!!.!!!!!!!!!!!!.!!!!!!!!!!!.!!
Success rate is 91 percent (639/696), round-trip min/avg/max = 4/4/21
ms
```

Note the count of "!" signs in the selected sample which is 11. Multiply this number by the packet size, which is 1514 on Ethernet.

Bits = 11*1514*8=133232. It took 11*4ms to send those packets, plus one second of pause waiting for the dropped packet. Effectively, we send all those bits in the time interval of approximately 1 second. This is close to the desired value of 128Kilobits per second.

```
Rack1R6#ping 155.1.146.4 repeat 10000 size 1500 timeout 1
Type escape sequence to abort.
Sending 10000, 1500-byte ICMP Echos to 155.1.146.4, timeout is 1
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!
!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!
!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!
!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!
!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!
.
Success rate is 96 percent (338/351), round-trip min/avg/max = 4/4/12
ms
```

Take the sample burst of 26 packets. The size of 26 packets in bits is

26*1514*8=314912.

This is far below the target value of 384 Kbps. This is probably due to the fact that now the token bucket refills much faster, and our procedure is not accurate enough to meter precisely the average rate.

Now verify that policing applies to IPX traffic as well. In this case, we source IPX packets that are going to have CoS value of 0 in SW4. This value maps to DSCP 0.

```
R4:
ipx routing
!
interface FastEthernet 0/1
 ipx network 146 encapsulation snap

R6:
ipx routing
!
interface FastEthernet 0/0.146
 ipx network 146 encapsulation snap

Rack1R4#show ipx interface fastEthernet 0/1
FastEthernet0/1 is up, line protocol is up
  IPX address is 146.0011.2031.0901, SNAP [up]

Rack1R6#ping
Protocol [ip]: ipx
Target IPX address: 146.0011.2031.0901
Repeat count [5]: 100000
Datagram size [100]: 1000
Timeout in seconds [2]: 1
Verbose [n]:
Type escape sequence to abort.
Sending 100000, 1000-byte IPX Novell Echoes to 146.0011.2031.0901,
timeout is 1 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!.
Success rate is 97 percent (222/228), round-trip min/avg/max = 1/3/5 ms
```

Copyright © 2010 Internetwork Expert

10.77 Catalyst 3560 SRR Shared Mode

- Configure SW1 to set the speed of the port connected to R1 to 10 Mbps, and limit the bandwidth to 20%.
- Preserve the default DSCP to Queue ID mapping.
- Set all SRR shaped weights to zero.
- Set the SRR shared weights to 1, 10, 20, and 30 for queues 1 through 4.
- Enable expedite queue on the interface.

Configuration

```
SW1:
mls qos
!
! Those mappings are the default but we still present
! them for the sake of clarity
!
mls qos srr-queue output dscp-map queue 1 46
mls qos srr-queue output dscp-map queue 2 0
mls qos srr-queue output dscp-map queue 3 18
mls qos srr-queue output dscp-map queue 4 34
!
! Apply SRR weights
!
interface FastEthernet 0/1
 speed 10
 srr-queue bandwidth limit 20
 srr-queue bandwidth shape  0  0  0  0
 srr-queue bandwidth share 1 10 20 30
 priority-queue out
```

Verification

Note

Like the 3550 model, the 3560 has four egress queues per interface. However, the 3560 does not make any distinction between GigibitEthernet or FastEthernet ports. All of them have uniform scheduler, called SRR (which we mentioned earlier). Depending on the modes of operation, SRR stands for either Shared Round Robin (we discussed the use of SRR for ingress queues) or Shaped Round Robin.

Copyright © 2010 Internetwork Expert

You can configure each of the four egress queues in either Shaped or Shared mode. Here is how the scheduler works with each of them:

1) All queues configured in shaped mode have an absolute weight assigned. The interface-level command to assign the weights is

```
srr-queue bandwidth shape <w1> <w2> <w3> <w4>
```

If some of the weights are zero, then the respective queues operate in *shared* mode. However, for non-zero weights the scheduler places the respective queue in *shaped* mode.

In shaped mode, the system limits the queue sending rate to **1/weight*interface-speed**. The speed here is the actual physical rate set with the **speed** interface-level command. At the same time, the SRR scheduler *guarantees* this rate to the queue. For example if the interface speed is 100Mbps and the shaped weight is 20, then the respective queue is shaped to 5Mbps. The switch meters output traffic and delays the exceeding traffic to conform to the configure rate. However, the details of the shaping procedure is not published.

2) All queues *not* configured in *shaped* mode operate in *shared* mode. The scheduler obtains weights for shared mode queues from the command

```
srr-queue bandwidth share <w1> <w2> <w3> <w4>
```

These weights are always non-zero. If it happens so that the corresponding *shaped* weight is non-zero also, then the scheduler *ignores* the shared weight and considers it zero in bandwidth share computations.

All shared queues share the bandwidth *remaining* after the shaped queues. In fact, the SRR scheduler "prefers" shaped queues to shared queues and ensures the former have the bandwidth they need up to their limit. However, you need to make sure that absolute bandwidth values assigned to shaped queues do not *exceed* the interface rate (or speed limit, which we discuss later). Otherwise, unexpected behavior may result.

Shared queues share the remaining bandwidth *proportional* to their configured weights. If there are no shaped queues on the interface, four queues simply divide the interface bandwidth in proportions:

<w1>:<w2>:<w3>:<w4>

Copyright © 2010 Internetwork Expert

That means bandwidth shares are

s(i)=w(i)/(w(1)+...w(4)) for queue number "i".

If any of the queues operate in shaped mode, consider the respective *w(i)* to be zero, irrespective of the actual weight setting.

Consider the following example. A one hundred megabit interface has queues 1 and 2 in shaped mode and queues 3 and 4 in shared mode. Shaped weights are 10 and 20 (Qeueues 1 and 2), and shared weights are 30 and 70 (Queues 3 and 4):

Example:

```
interface FastEthernet 0/1
 speed 100
 srr-queue bandwidth shape 10 20 0 0
 srr-queue bandwidth share  1 1 30 70
```

In case of congestions the scheduler divides bandwidth as follows:

Queue 1: 10 Mbps.
Queue 2: 5 Mbps.
Queue 3: 30/(30+70)*(100-10-5)=85*0,3=25,5 Mbps.
Queue 4: 70/(30+70)*(100-10-5)=85,*0,7=59,5 Mbps.

Another feature supported by SRR is *priority* (or expedite) queue. Just as with the 3550, you can enable one queue for expedite services. However, with the 3560 this is queue number 1, not queue number 4. When queue 1 works as priority, it has unlimited bandwidth and may starve any other queue (either shared or shaped). In addition, shared and shaped weights set for the priority queue has no effect and the SRR scheduler does not take them in account.

A feature unique to the 3560 is limiting the egress port sending rate. You can use special interface level shaper to restrict the port sending rate to a value between 10% and 100% of its physical rate. The interface-level command is

srr-queue bandwidth limit <percent>

Note that the <percent> value ranges from 10% to 100%. Thus, the lowest sending rate for an interface is 1Mbps, achieved with physical speed 10Mbps and 10% limit. Note that shaped queue weights still apply to the physical speed, not the bandwidth limit, when calculating shaped queue rates.

Copyright © 2010 Internetwork Expert

The default values for SRR shaped and shared weights are:

Queue 1: Shaped 25, Shared 25
Queue 2: Shaped 0, Shared 25
Queue 3: Shaped 0, Shared 25
Queue 3: Shaped 0, Shared 25

Note that Queue 1 uses shaped mode, while all other queues share the remaining bandwidth in equal proportions.

The next thing is mapping of DSCP and CoS values to Queue IDs. Recall that in the 3550 you can only map CoS values (DSCPs are mapped implicitly) and you have flexibility to do that at the interface level. However, in the 3560 all mappings are global and apply to all interfaces at the same time. Usually that makes sense in enterprise switches, where the users in the same QoS domain share one device.

```
mls qos srr-queue output dscp-map queue <q-id> <DSCP List>
mls qos srr-queue output cos-map queue <q-id> <CoS List>
```

Note that you can map DSCP and CoS values separately. The switch chooses the value you trusted or set at the classification stage to select the proper table. The default mappings exist and you can check them using the **show mls qos map** commands.

In this particular task, we use the default DSCP to Queue ID mappings, where the default DSCP maps to Queue 2, AF21 maps to Queue 3, and AF41 maps to Queue 4. Note that DSCP EF by default maps to Queue 1. In the example below, we enable queue 1 as a priority queue and configure shared weights to attain the required proportions.

We set the port speed to 10Mbps and bandwidth limit to 20% in order to achieve 2Mbps output rate. For verification purpose, we configure the following devices as traffic sources: R4, R6, SW3, and SW4. The switch connected to R1 (SW1) should trust DSCP on all it trunk ports.

R4 will send DSCP EF traffic (TOS 184);
R6 will source DSCP AF21 traffic (TOS 72).
R3 and R5 will source DSCP AF41 (TOS 136) and DSCP 0 packets respectively. Note that you need to reconfigure R3 and R5 temporarily to accomplish this.

Copyright © 2010 Internetwork Expert

After that, we limit all sources sending rate to 1Mbps using traffic shaping on the routers. However, R4's sending rate will be limited down to 256Kbps, since it sources priority traffic (simulates a VoIP packet flow). This configuration will definitely oversubscribe the interface limited to 2Mbps.

Note that MLS QoS is disabled on all switches with except to SW1.

```
R3:
interface FastEthernet 0/1
 ip address 155.1.146.3 255.255.255.0
 no shutdown
 traffic-shape rate 1000000

R4:
interface FastEthernet 0/1
 traffic-shape rate 256000

R5:
interface FastEthernet 0/1
 ip address 155.1.146.5 255.255.255.0
 traffic-shape rate 1000000

R6:
interface FastEthernet 0/1.146
 traffic-shape rate 1000000

SW1:
interface range FastEthernet 0/13 - 21
 mls qos trust dscp

SW2:
no mls qos

SW3:
no mls qos
!
interface FastEthernet 0/3
 switchport access vlan 146
!
interface FastEthernet 0/5
 switchport access vlan 146

SW4:
no mls qos
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#ping
Protocol [ip]:
Target IP address: 155.1.146.1
Repeat count [5]: 100000000
Datagram size [100]: 1500
Timeout in seconds [2]: 0
Extended commands [n]: y
Source address or interface:
Type of service [0]: 184
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100000000, 1500-byte ICMP Echos to 155.1.146.1, timeout is 0
seconds:
 ..............................................................
 ..............................................................
 ..............................................................

Rack1R6#ping
Protocol [ip]:
Target IP address: 155.1.146.1
Repeat count [5]: 100000000
Datagram size [100]: 1500
Timeout in seconds [2]: 0
Extended commands [n]: y
Source address or interface:
Type of service [0]: 72
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100000000, 1500-byte ICMP Echos to 155.1.146.1, timeout is 0
seconds:
 ..............................................................
 ..............................................................
 ..............................................................
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#ping
Protocol [ip]:
Target IP address: 155.1.146.1
Repeat count [5]: 100000000
Datagram size [100]: 1500
Timeout in seconds [2]: 0
Extended commands [n]: y
Source address or interface:
Type of service [0]: 136
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100000000, 1500-byte ICMP Echos to 155.1.146.1, timeout is 0
seconds:
 ..............................................................
 ..............................................................
 ..............................................................

Rack1R5#ping 155.1.146.1 repeat 10000000 size 1500 timeout 0

Type escape sequence to abort.
Sending 10000000, 1500-byte ICMP Echos to 155.1.146.1, timeout is 0
seconds:
 ..............................................................
 ..............................................................
 ..............................................................
 ..............................................................
```

Configure R1 for detection of input traffic marked with the DSCP values mentioned in the task.

```
R1:
class-map DSCP_EF
 match dscp ef
!
class-map DSCP_AF21
 match dscp af21
!
class-map DSCP_0
 match dscp 0
!
class-map DSCP_AF41
 match dscp af41
!
policy-map DETECT
 class DSCP_EF
 class DSCP_0
 class DSCP_AF21
 class DSCP_AF41
!
interface FastEthernet 0/0
 load-interval 30
 service-policy input DETECT
```

Copyright © 2010 Internetwork Expert

Now observe the results on R1:

```
Rack1R1#show policy-map interface fastEthernet 0/0
 FastEthernet0/0

  Service-policy input: DETECT

    Class-map: DSCP_EF (match-all)
      12065 packets, 18266410 bytes
      30 second offered rate 255000 bps
      Match:  dscp ef (46)

    Class-map: DSCP_0 (match-all)
      22841 packets, 34581274 bytes
      30 second offered rate 284000 bps
      Match:  dscp default (0)

    Class-map: DSCP_AF21 (match-all)
      45835 packets, 69394190 bytes
      30 second offered rate 572000 bps
      Match:  dscp af21 (18)

    Class-map: DSCP_AF41 (match-all)
      68628 packets, 103902792 bytes
      30 second offered rate 860000 bps
      Match:  dscp af41 (34)

    Class-map: class-default (match-any)
      102 packets, 14972 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

Note that the proportion 1:2:3 holds true for DSCP 0, AF21, and AF41 traffic. The total aggregate input bandwidth is close to 2 Mbps.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.78 Catalyst 3560 SRR Shaped Mode

- Change the shaped weight for queue 4 to 1Mbps.
- Set the shared weights to 1, 10, 20, and 1 for queues 1 through 4 respectively.

Configuration

```
SW1:
!
! Apply SRR weights
!
interface FastEthernet 0/1
 speed 10
 srr-queue bandwidth limit 20
 srr-queue bandwidth shape  0  0  0  10
 srr-queue bandwidth share 1 10 20 1
 priority-queue out
```

Verification

Note

This task requires limiting one of the queues to 1Mbps. Since the interface rate is 10Mbps, the shaped weight would be 10. The other two classes get the weight values of 20 and 10. Those classes will share the bandwidth remaining after the priority queue and shaped queue in proportions specified.

For verification purpose, we configure the following devices as traffic sources: R4, R6, SW3, and SW4. The switch connected to R1 (SW1) should trust DSCP on all it trunk ports.

R4 will send DSCP EF traffic (TOS 184);
R6 will source DSCP AF21 traffic (TOS 72).
R3 and R5 will source DSCP AF41 (TOS 136) and DSCP 0 packets respectively. Note that you need to reconfigure R3 and R5 temporarily to accomplish this.

After that, we limit all sources sending rate to 1Mbps using traffic shaping in routers. However, R4 sending rate will be limited down to 256Kbs, since it sources priority traffic (simulates a VoIP packet flow). This configuration will definitely oversubscribe the interface limited to 2Mbps.

Note that MLS QoS is disabled on all switches with except to SW1.

Copyright © 2010 Internetwork Expert

```
R3:
interface FastEthernet 0/1
 ip address 155.1.146.3 255.255.255.0
 no shutdown
 traffic-shape rate 1000000

R4:
interface FastEthernet 0/1
 traffic-shape rate 256000

R5:
interface FastEthernet 0/1
 ip address 155.1.146.5 255.255.255.0
 traffic-shape rate 1000000

R6:
interface FastEthernet 0/1.146
 traffic-shape rate 1000000

SW1:
interface range FastEthernet 0/13 - 21
 mls qos trust dscp

SW2:
no mls qos

SW3:
no mls qos
!
interface FastEthernet 0/3
 switchport access vlan 146
!
interface FastEthernet 0/5
 switchport access vlan 146

SW4:
no mls qos
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#ping
Protocol [ip]:
Target IP address: 155.1.146.1
Repeat count [5]: 100000000
Datagram size [100]: 1500
Timeout in seconds [2]: 0
Extended commands [n]: y
Source address or interface:
Type of service [0]: 184
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100000000, 1500-byte ICMP Echos to 155.1.146.1, timeout is 0
seconds:
 ..............................................................
 ..............................................................
Rack1R6#ping
Protocol [ip]:
Target IP address: 155.1.146.1
Repeat count [5]: 100000000
Datagram size [100]: 1500
Timeout in seconds [2]: 0
Extended commands [n]: y
Source address or interface:
Type of service [0]: 72
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100000000, 1500-byte ICMP Echos to 155.1.146.1, timeout is 0
seconds:
 ..............................................................
 ..............................................................
 ..............................................................
```

Copyright © 2010 Internetwork Expert

```
Rack1R3#ping
Protocol [ip]:
Target IP address: 155.1.146.1
Repeat count [5]: 100000000
Datagram size [100]: 1500
Timeout in seconds [2]: 0
Extended commands [n]: y
Source address or interface:
Type of service [0]: 136
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100000000, 1500-byte ICMP Echos to 155.1.146.1, timeout is 0
seconds:
 ..............................................................
 ..............................................................
 ..............................................................

Rack1R5#ping 155.1.146.1 repeat 10000000 size 1500 timeout 0

Type escape sequence to abort.
Sending 10000000, 1500-byte ICMP Echos to 155.1.146.1, timeout is 0
seconds:
 ..............................................................
 ..............................................................
 ..............................................................
 ..............................................................
```

Copyright © 2010 Internetwork Expert

Configure R1 for detection of input traffic marked with the DSCP values mentioned in the task.

```
R1:
class-map DSCP_EF
 match dscp ef
!
class-map DSCP_AF21
 match dscp af21
!
class-map DSCP_0
 match dscp 0
!
class-map DSCP_AF41
 match dscp af41
!
policy-map DETECT
 class DSCP_EF
 class DSCP_0
 class DSCP_AF21
 class DSCP_AF41
!
interface FastEthernet 0/0
 load-interval 30
 service-policy input DETECT
```

Copyright © 2010 Internetwork Expert

Now observe the results on R1:

```
Rack1R1#show policy-map interface fastEthernet 0/0
 FastEthernet0/0

  Service-policy input: DETECT

    Class-map: DSCP_EF (match-all)
      6146 packets, 9305044 bytes
      30 second offered rate 255000 bps
      Match:  dscp ef (46)

    Class-map: DSCP_0 (match-all)
      9794 packets, 14828116 bytes
      30 second offered rate 408000 bps
      Match:  dscp default (0)

    Class-map: DSCP_AF21 (match-all)
      19671 packets, 29781894 bytes
      30 second offered rate 817000 bps
      Match:  dscp af21 (18)

    Class-map: DSCP_AF41 (match-all)
      11792 packets, 17853088 bytes
      30 second offered rate 491000 bps
      Match:  dscp af41 (34)

    Class-map: class-default (match-any)
      73 packets, 16098 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

Note that bandwidth proportions of DSCP 0 and DSCP AF21 classes is 1:2. The bandwidth guaranteed to AF41 class is close to target 500Kbps.

Copyright © 2010 Internetwork Expert

10.79 Catalyst 3560 Egress Queues Tuning

- Configure SW1 to modify buffer sharing for queues on the interface connected to R1 only.
- Ensure that queue 2 has 70% of interface buffer pools and remaining queues share buffers in equal proportions.
- Reserve only 10% of allocated buffers for queue 2; the remaining queues should reserve all allocated buffers.
- Ensure that traffic marked with AF11, AF21, AF41, and CS1 maps to the second drop threshold in their respective queues.
- Set the first drop threshold to 200% and the second drop threshold to 150% for all queues.
- Ensure that the switch drops voice packets marked with DSCP EF only if the respective queue is exhausted.
- Set the maximum threshold to 250% for all queues

Configuration

```
SW1:
!
! Map DSCP EF to queue-full threshold
! Map other DSCPs to the lower threshold (2)
!
mls qos srr-queue output dscp-map threshold 3 46
mls qos srr-queue output dscp-map threshold 2 8 10 18 34
!
! Configure queue threshold. First two are the drop threshold
! The other two specify how much to reserve and how much a queue
! can borrow from a common pool.
!
mls qos queue-set output 2 threshold 1 200 150 100 250
mls qos queue-set output 2 threshold 2 200 150 10 250
mls qos queue-set output 2 threshold 3 200 150 100 250
mls qos queue-set output 2 threshold 4 200 150 100 250
!
! Partition reserved pool between queues in proportions 1:7:1:1
!
mls qos queue-set output 2 buffers 10 70 10 10
!
! Apply queue-set to the interface. By default, all interfaces
! use queue-set 1 so we modify only the port connected to R1.
!
interface FastEthernet 0/1
 queue-set 2
```

Copyright © 2010 Internetwork Expert

Verification

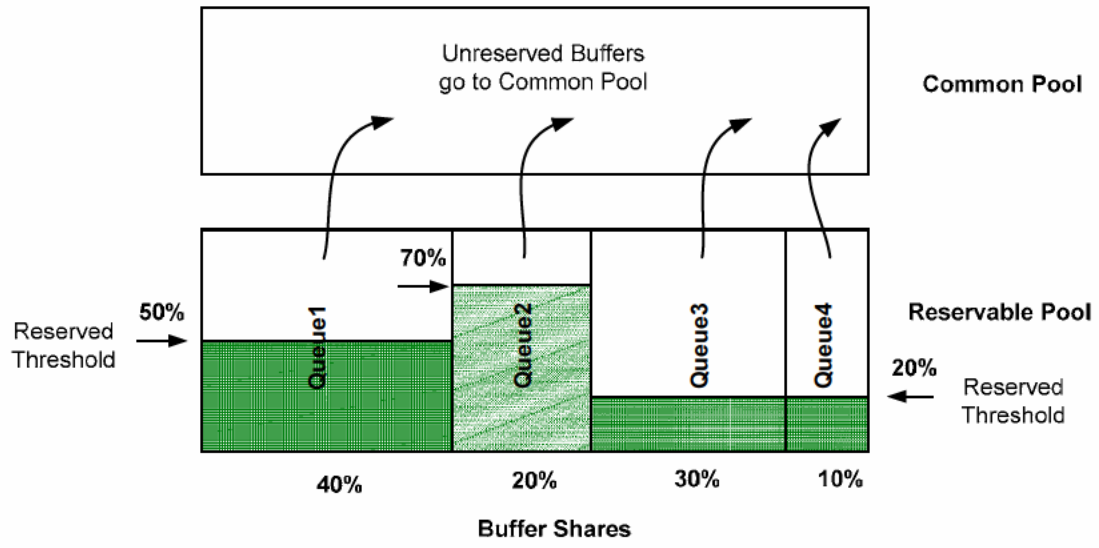
Note

The queue buffer partitioning scheme in the 3560 is somewhat complicated. First of all, you cannot specify buffer allocation per interface. There exist two configurable global templates, which you can apply to interfaces individually. Those templates are called "queue-sets" and by default, all interfaces use queue-set 1. Within a queue-set you can define buffer allocation between interface queues and drop thresholds for each queue. However, remember that a queue set is merely a template that has effect only when applied to physical interface.

Physical interfaces have their own reservable buffers pool. You can partition this pool between four egress queues using the template command:

```
mls qos queue-set <qset-id> output <q-id> <w1> <w2> <w3>
<w4>
```

Where *<w1>...<w4>* are the weights in percents that specify how many buffers in the pool to allocate to the respective queue identified by number *<q-id>*. The sum of all weights should not exceed 100%. On the figure below, you can see those weights as buffer shares on the bottom of the reservable pool. There is also special common pool, shared by all interaces, which we discuss later.



For example if a reservalbe pool is 1000 buffers and you partition it in proportions 40%, 20%, 30%, and 10%, then queues 1 through 4 would get 400, 200, 300 and 100 buffers respectively.

After you allocated shares of reservable space to queues, you can set up queue thresholds. There are two types of threshold per queue, set by a single command.

```
mls qos queue-set output <qset-id> <threshold-id> <q-id>
<drop-threshold1> <drop-threshold2> <reserved-threshold>
<maximum-threshold>
```

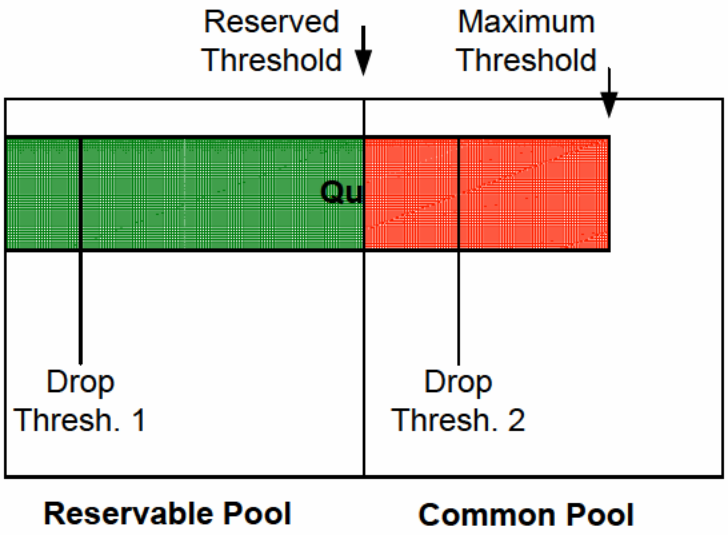
The first type defines the “drop” thresholds and specifies queue depth values where the scheduler drops certain packet types. There are two drop thresholds per queue and you can map DSCP/CoS values to drop thresholds.

The second threshold type defines queue size itself. As we know, each queue gets a part of reservable buffer space. This part may be even zero, as will see later. For every queue, you can define reserved threshold (in percents) and maximum threshold (in percents). The system uses both thresholds to “scale” the number of buffers allocated to a queue.

Let’s look at the reserved threshold first. This one defines the amount of buffer space the system actually reserves for the queue. For example, if the reservable pool is 1000 buffers, and you allocate 40% of the pool to queue 1, then the queue has 400 buffers as its base. Now if if you set the reservable threshold to 50%, then the system guarantees only 200 buffers to the queue. The system takes away and *shares* the remaining buffers in a common pool between other queues. Thus, the reservable threshold may vary from 0% to 100%, with the default value being 50%. Look at the figure above once again for an illustration.

The maximum threshold is also in percent. Like the reservable threshold, it applies to the number of buffers allocated to a queue during the partitioning process. This value ranges from 1% to 400% and specifies the maximum queue size in buffers. You multiply the number of buffers allocated to a queue by this value to get maximum queue size. Note that queue may grow *beyond* its reserved limit if the maximum threshold is above 100%. The queue borrows extra buffers from common buffer pool, if there are any. The queue may grow up to 4 times of its allocated size with this feature.

Copyright © 2010 Internetwork Expert



Now look at the new figure to see the relation between the drop thresholds and the reserved/maximum thresholds. As you can see, reserved threshold simpy defines the cut-off point inside the space allocated for a queue. The remaining space goes to the common buffer pool. Drop thresholds are also in percents, and they may not exceed the maximum threshold. The following is the list of restrictions that apply to the thresholds:

1) Maximum Thresholds <= 400%
2) Reserverd Threshold <= 100%
3) Drop Thresholds <= Maximum Threshold

You can even set the Reserved Threshold to 0%, putting all buffer space allocated to a queue to the common pool. In this condition, the queue will only rely on the common pool for the buffer space.

Here are the default settings for buffers in queue-set 1:

```
Rack1SW1#show mls qos queue-set
Queueset: 1
Queue     :       1       2       3       4
----------------------------------------------
buffers   :      25      25      25      25
threshold1:     100     200     100     100
threshold2:     100     200     100     100
reserved  :      50      50      50      50
maximum   :     400     400     400     400
```

Now the last piece is mapping of the DSCP/CoS value to drop thresholds. You do this using the familiar global mapping commands:

mls qos srr-queue output dscp-map threshold <thresh-id> <DSCP List>

Remember that threshold 3 is a non-configurable "queue-full" state threshold. Generally, you map important traffic to the larger threshold, implementing Weighted Tail Drop (WTD) discipline.

Verification can be performed as follows.

```
Rack1SW1#show mls qos queue-set 2
Queueset: 2
Queue     :       1       2       3       4
----------------------------------------------
buffers   :      10      70      10      10
threshold1:     200     200     200     200
threshold2:     150     150     150     150
reserved  :     100      10     100     100
maximum   :     250     250     250     250

Rack1SW1#show mls qos interface gigabitEthernet 0/1 buffers
GigabitEthernet0/1
The port is mapped to qset : 2
The allocations between the queues are : 10 70 10 10

Rack1SW1#show mls qos maps dscp-output-q
   Dscp-outputq-threshold map:
     d1 :d2    0     1     2     3     4     5     6     7     8     9
     ------------------------------------------------------------
      0 :    02-01 02-01 02-01 02-01 02-01 02-01 02-01 02-01 02-02 02-01
      1 :    02-02 02-01 02-01 02-01 02-01 02-01 03-01 03-01 03-02 03-01
      2 :    03-01 03-01 03-01 03-01 03-01 03-01 03-01 03-01 03-01 03-01
      3 :    03-01 03-01 04-01 04-01 04-02 04-01 04-01 04-01 04-01 04-01
      4 :    01-01 01-01 01-01 01-01 01-01 01-01 01-03 01-01 04-01 04-01
      5 :    04-01 04-01 04-01 04-01 04-01 04-01 04-01 04-01 04-01 04-01
      6 :    04-01 04-01 04-01 04-01
```

Note the DSCP values mapped to the new thresholds in the above output.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.80 Catalyst QoS DSCP Mutation

- Configure SW4 so that R1 sees packets marked at R4 with DSCP values CS0, AF31, and CS5 as DSCP values CS1, CS3, and EF.

Configuration

```
SW4:
mls qos
!
mls qos map dscp-mutation MUTATION 0 to 8
mls qos map dscp-mutation MUTATION 26 to 24
mls qos map dscp-mutation MUTATION 40 to 46
!
interface FastEthernet 0/4
 mls qos trust dscp
 mls qos dscp-mutation MUTATION
```

Verification

Note

A DSCP mutation map is a special function that may come in handy on the border of two QoS domains that use different markings. You may create many maps and apply them per-interface. The syntax for the mapping command is:

mls qos map dscp-mutation <NAME> <From: DSCP List> <To: DSCP Value>
...

Each line in a DSCP mutation map translates a list of up to 8 DSCP values to a new value. One necessary condition to use the mutation maps is trusting DSCP on the ingress interface. To verify this create an access-list in R1 to detect the mentioned DSCP values. Disable MLS QoS in all switches except SW4.

```
R1:
no ip access-list extended DETECT
ip access-list extended DETECT
 permit icmp any any dscp 0
 permit icmp any any dscp cs1
 permit icmp any any dscp 26
 permit icmp any any dscp 24
 permit icmp any any dscp 40
 permit icmp any any dscp 46
 permit ip any any
!
interface FastEthernet 0/0
 ip access-group DETECT in
```

Copyright © 2010 Internetwork Expert

```
SW1, SW2, SW3:
no mls qos
```

Send ICMP packets from R4 marked with DSCP 0, DSCP AF31 (TOS 106), DSCP CS5 (TOS 160).

```
Rack1R4#ping 155.1.146.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms
Rack1R4#

Rack1R4#ping
Protocol [ip]:
Target IP address: 155.1.146.1
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface:
Type of service [0]: 104
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/4 ms

Rack1R4#ping
Protocol [ip]:
Target IP address: 155.1.146.1
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface:
Type of service [0]: 160
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#show ip access-lists
Extended IP access list DETECT
    10 permit icmp any any dscp default
    20 permit icmp any any dscp cs1 (15 matches)
    30 permit icmp any any dscp af31
    40 permit icmp any any dscp cs3 (15 matches)
    50 permit icmp any any dscp cs5
    60 permit icmp any any dscp ef (30 matches)
    70 permit ip any any (240 matches)
```

Note that the only access-list entries matched are the mutated (mapped) values.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

10.81 Advanced HTTP Classification with NBAR

- Configure R6's VLAN146 interface so that HTTP transfers of files with extensions ".bin", ".text" and ".taxt" are limited to 256Kbps.
- Use only one line to match the URL strings.
- Use the method most friendly for TCP connections.
- Do not apply this limit to the transfers of any other file types.

Configuration

```
R6:
class-map match-all EXTENSION
 match protocol http url "*.bin|*.t[ea]xt"
!
!
policy-map SHAPE
 class EXTENSION
  shape average 256000
!
interface FastEthernet 0/0.146
  service-policy output SHAPE
```

Verification

Note

NBAR (Network Based Application Recognition) protocol classification allows deep inspection for some protocol types. One most notable feature is matching various fileds inside HTTP headers. In recent IOS versions, this capability evolved into a full-blown Flexible Packet Matching feature, which allows for comprehensive inspection of IP packets.

Most commonly you will see HTTP inspection matching the URL field against certain strings.

The command to match URL is

match http protocol url <pattern>

Copyright © 2010 Internetwork Expert

Here is the list of the available wildcards used to construct the matching patterns:

"*" - match any pattern e.g. "aaa", "abcd1234". To match the substring "xyz" in the beginning of a string use the pattern "xyz*". To match "xyz" anywhere in the string use the pattern "*xyz*". Use the pattern "*xyz" to match the substring in the end. Note that pattern "xyz" matches only the exact string "xyz". You may also use complicated patterns "ab*cd*ef" at the expense of some CPU penalty probably.

"?" - match any single character. For example pattern "???" matches "xyz", "abc","efg","123". You can mix "*" and "?" like in pattern "tes*.????"

"[]" - match range of characters. E.g. "[abc]" will match any single character "a", "b" or "c".

"|" - alternative. Separate patterns with "|" in order to specify "OR" matching logic. For example "xyz|abc" matches either full "xyz" or "abc" strings. You may mix "|" with other globs like this "*xyz*|*abc*|*pqr*". Note sure about the overall limit of using the "|" symbol but you'd better keep it to minimum, in order to make matching faster.

"()" - grouping. Denotes the boundaries of a sub-pattern. For example instead of "*.txt|*.bin" you can write "*.(txt|bin)".

All matching is case insensitive. Therefore, the pattern "text" matches "TEXT" as well. The engine matches your URL pattern against the directory path and the file name in the URL (NOT the server DNS name!). E.g. If the URL is "http://www.cisco.com/pub/uploads/image.jpeg" the URL matching will only match the "pub/uploads/image.jpeg" part of the URL. As a matter of fact, when you submit request like the above URL, it translates into the following headers (there are actually more, but this is the bare minimum):

```
GET /pub/uploads/image.jpeg HTTP/1.1
Host: www.cisco.com
```

When you apply a policy-map that contains a class with "match protocol" statement, the system starts NBAR classification engine on the interface. Any packet, be it ingress or ingress, passes the NBAR inspection engine provided that it passes the basic filters like matching the port number assigned to the protocol. You can change the port map using the global command **`ip nbar port-map [protocol]`**.

Copyright © 2010 Internetwork Expert

When the engine sees a TCP SYN packet for the matching session, it starts the internal state machine, trying to parse the packet flow. Every new packet in the flow (in any direction) is inspected. Note that the NBAR does not classify a flow instantly. It may take some packet exchange until the engine determines that the flow matches specified criteria. As soon as there is enough information about the flow to classify it, the engine "tags" the bi-directional flow with the corresponding class value and reports this decision to the policy-map. Note that this classification applies in both directions – that is to the packets belonging to the flow and heading either ingress or egress.

For example, if a user starts a web sessions ands opens an URL matching any of your NBAR criteria, the engine will classify the flow as soon as it sees the packet with the URL string. After this, both packet flows from the client to server or from the server to the client would the respective class – that is, the policy map could be applied in either direction of the interface.

The engine will remove the "tag" when it faces the flow "end" criteria: e.g. it catches a TCP FIN or TCP RST flag. After this, flow is no longer monitored by NBAR and reported as matching the respective class.

Consider the example below. In this scenario, Fa0/0 is the outside interface, facing the internet. User's traffic flows out of this interface.

```
class-map match-all TEST
 match protocol http url "*.(t?xt|ocx|ex[ea])
!
policy-map TEST
 class TEST
 drop
!
interface FastEthernet0/0
ip address 155.1.37.3 255.255.255.0
service-policy input TEST
```

However, as soon as user opens the URL matching the class-map specification, the engine will classify the flow as matching the class "TEST". After this, the policy will drop all returning packets (server to client) for this flow.

Note that even though this configuration works, it is not the recommended way of applying the URL filtering. This is because the actual GET requests will still get to the destination server and generate response traffic. In some situations, the client may send a FIN packet when server generates the response. At this point, NBAR will report the flow as destroyed and the policy-map will not match the packets. If the server response is slow enough, it will actually bypass the ingress filter! Thus, it is always recommended to apply the NBAR classification and policy action in the direction matching the direction of the protocol commands (e.g. apply the URL filtering in the direction of GET commands).

Copyright © 2010 Internetwork Expert

We are going to use the following single-line expression for our task:

"*.bin|*.t[ea]xt"

This pattern matches either of two strings that ends with ".bin" or ".text", ".taxt". You can also use "match-any" class map and match multiple URL strings on separate lines, resulting in the same effect as separating patterh with the symbol "|".

You can match other fields in the HTTP headers. Specifically you can match a pattern against Client and Server headers, using the expressions:

```
match http protocol c-header-field <string>
match http protocol s-header-field <string>
```

Those expressions match any of the client or server headers. Additionally you can match the "Host" header field separately using the command

```
match http protocol host <string>
```

Finally, you can match MIME types to distinguish certain "types" of files being transferred (e.g. audio/video) using the following command:

```
match http protocol mime <MIME-Type>
```

e.g.

```
match http protocol mime "image/jpeg"
```

Copyright © 2010 Internetwork Expert

However, matching URIs is the most common use of advanced HTTP classification. To verify the configuration, set SW1 as an HTTP server and create several file-names in the flash memory with extensions ".bin", ".text", ".taxt", "tsxt".

```
SW1:
ip http server
ip http server path flash:

Rack1SW1#copy flash:config.text flash:config.bin
Rack1SW1#copy flash:config.text flash:config.tsxt
Rack1SW1#copy flash:config.text flash:config.taxt
```

Now we have four files to try. Using R4 as HTTP client verify that ".bin", ".text" and ".taxt" match our configuration but ".tsxt" does not.

```
Rack1R6#clear counters fastEthernet 0/0
Clear "show interface" counters on this interface [confirm]

Rack1R4#copy http://admin:cisco@155.1.67.7/config.bin null:
Loading http://***********@155.1.67.7/config.BIN !
2438 bytes copied in 0.048 secs (50792 bytes/sec)

Rack1R6#show policy-map interface fastEthernet 0/0.146
 FastEthernet0/0.146

  Service-policy output: SHAPE

    Class-map: EXTENSION (match-all)
      4 packets, 1528 bytes
      5 minute offered rate 1000 bps
      Match: protocol http url "*bin|*t[ea]xt"

    Class-map: class-default (match-any)
      35 packets, 7957 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

Rack1R4#copy http://admin:cisco@155.1.67.7/config.text null:
Loading http://***********@155.1.67.7/config.text !
2438 bytes copied in 0.044 secs (55409 bytes/sec)

Rack1R6#show policy-map interface fastEthernet 0/0.146
 FastEthernet0/0.146

  Service-policy output: SHAPE

    Class-map: EXTENSION (match-all)
      9 packets, 3047 bytes
      5 minute offered rate 1000 bps
      Match: protocol http url "*bin|*t[ea]xt"
```

Copyright © 2010 Internetwork Expert

```
Class-map: class-default (match-any)
      43 packets, 9637 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

Rack1R4#copy http://admin:cisco@155.1.67.7/config.taxt null:
Loading http://***********@155.1.67.7/config.taxt !
2438 bytes copied in 0.048 secs (50792 bytes/sec)

Rack1R6#show policy-map interface fastEthernet 0/0.146
 FastEthernet0/0.146

  Service-policy output: SHAPE

    Class-map: EXTENSION (match-all)
      13 packets, 4642 bytes
      5 minute offered rate 1000 bps
      Match: protocol http url "*bin|*t[ea]xt"

    Class-map: class-default (match-any)
      49 packets, 10897 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

Rack1R4#copy http://admin:cisco@155.1.67.7/config.tsxt null:
Loading http://***********@155.1.67.7/config.tsxt !
2438 bytes copied in 0.044 secs (55409 bytes/sec)

Rack1R6#show policy-map interface fastEthernet 0/0.146
 FastEthernet0/0.146

  Service-policy output: SHAPE

    Class-map: EXTENSION (match-all)
      13 packets, 4642 bytes
      5 minute offered rate 0 bps
      Match: protocol http url "*bin|*t[ea]xt"

    Class-map: class-default (match-any)
      58 packets, 13327 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Note how the packet counter for the class increases for the first three file names.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Security Solutions

> **Note**
>
> Load the Security initial configurations prior to starting working on the tasks.

11.1 AAA Authentication Lists

- Configure R1 to use a TACACS+ server at the IP address 155.X.146.100 and encrypt communications using a key value of CISCO.
- Source TACACS+ packets from the Loopback 0 interface.
- Configure the router so that access to the console line uses local authentication.
- Ensure the VTY lines authenticate users with TACACS+ and then fall back to line authentication.
- Privilege mode authentication should first attempt TACACS+ and then fall back to the local password.
- Create a user named “ADMIN” with a password of “CISCO” in the local database for these configurations.
- Customize the prompts for AAA user authentication and change the default banner message.

Configuration

```
R1:
!
! Enable password and a local user
!
enable secret cisco
username ADMIN password CISCO
!
! Init AAA and configure AAA lists for Console/VTY logins
! Enable the use of TACACS+ for enable authentication but
! provide a fallback to local enable password
!
aaa new-model
aaa authentication login CONSOLE local
aaa authentication login VTY group tacacs+ line
aaa authentication enable default group tacacs+ enable
!
! Customize prompts
!
aaa authentication password-prompt "Please Enter Your Password:"
```

Copyright © 2010 Internetwork Expert

```
aaa authentication username-prompt "Please Enter Your ID:"
!
! Add a new authentication banner
!
aaa authentication banner #
This system requires you to identify yourself.
#
aaa authentication fail-message #
Authentication Failed, Sorry.
#
!
ip tacacs source-interface loopback 0
!
tacacs-server host 155.1.146.100
tacacs-server directed-request
tacacs-server key CISCO
!
line con 0
 login authentication CONSOLE
!
line vty 0 4
 login authentication VTY
 password cisco
```

Verification

Note

Cisco routers and switches may now use a "new" authentication model. The idea of the new model is to apply configurable lists of methods to authenticate, authorize, and/or perform accounting for certain processes. The processes that we are typically most interested in are:

1) Login process – the procedure to obtain access to the router, usually requiring you to present your identity.

2) Privileged mode access - in order to access a certain level of "enable mode" privileges, the system requires user authentication. You can configure local enable passwords for each level of access. You can also configure a named list of methods using the command **aaa authentication <process> {default|<NAME>} <List of Methods>** You may apply this list to a certain subsystem, e.g. to terminal or console lines. Note that a "default" list applies to all subsystems that have no explicit AAA list applied to them.

As for the list of possible methods, we should be most concerned with the following:

1) "Local" method – use the local user database with their passwords. You populate the database using the **username** command.

Copyright © 2010 Internetwork Expert

2) “Local-case” – the same as the local method, but makes passwords case-sensitive.

3) “Line” method – use the password configured on the line used to access the router. This includes VTY lines as well.

4) “Enable” method – use the globally configured list of enable passwords associated with their levels.

5) “Group” TACACS+ or RADIUS – uses the remote AAA servers group configured globally in the router.

6) “None” – do not attempt to validate user identity, just allow access.

Note an important property of the AAA list. The router tries all methods in sequence, switching to the next one if the previous method fails or returns an error. Here “Fail” means authentication method failure, e.g. non-responsive server, not the failure of the user account to be authenticated. This means that this list allows you to provide authentication redundancy. For example, consider a list which uses TACACS+ authentication first, and then switches to method “None”. In the case that your TACACS+ server does not respond, the router tries the next method and allows access. However, if the AAA server returns an authentication failure message, the router terminates the list search process, and denies access to the individual.

In this task configuration, you should verify the authentication process by trying to log in to the console line first:

```
Rack1R1 con0 is now available



Press RETURN to get started.


This system requires you to identify yourself.

Please Enter Your ID:ADMIN
Please Enter Your Password:CISCO

Rack1R1#
```

Note that the privilege level of the console line has been set to 15 by the initial configurations and thus the user logs directly to privileged mode.

Copyright © 2010 Internetwork Expert

Now enable AAA debugging and try logging in via a VTY line:

```
Rack1R1#debug aaa authentication
AAA Authentication debugging is on

Rack1R1#telnet 150.1.1.1
Trying 150.1.1.1 ... Open

AAA/BIND(0000000D): Bind i/f
AAA/AUTHEN/LOGIN (0000000D): Pick method list 'VTY'
This system requires you to identify yourself.

Please Enter Your Password:cisco
AAA/AUTHEN/LINE(0000000D): GET_PASSWORD
```

The router attempts to contact the TACACS+ server, fails and uses the second method, which is the line password.

```
AAA/AUTHEN/LINE(0000000D): PASS
AAA/AUTHOR (0000000D): Method=None for method list id=00000000. Skip
author

Rack1R1>enable
```

Now try entering the wrong password for privileged mode authentication. Note that the system tries the TACACS+ method first, but it returns the "ERROR" message. Thus, the system tries using the enable password for authentication, but the password entered does not match. Note the "Status=FAIL" response when you enter the wrong password.

```
AAA: parse name=tty66 idb type=-1 tty=-1
AAA: name=tty66 flags=0x11 type=5 shelf=0 slot=0 adapter=0 port=66
channel=0
AAA/MEMORY: create_user (0x85646E6C) user='NULL' ruser='NULL' ds0=0
port='tty66' rem_addr='150.1.1.1' authen_type=ASCII service=ENABLE
priv=15 initial_task_id='0', vrf= (id=0)
AAA/AUTHEN/START (2250436155): port='tty66' list='' action=LOGIN
service=ENABLE
AAA/AUTHEN/START (2250436155): using "default" list
AAA/AUTHEN/START (2250436155): Method=tacacs+ (tacacs+)
TAC+: send AUTHEN/START packet ver=192 id=-2044531141
AAA/AUTHEN(2250436155): Status=ERROR

AAA/AUTHEN/START (2250436155): Method=ENABLE
AAA/AUTHEN(2250436155): Status=GETPASS
AAA/AUTHEN/CONT (2250436155): continue_login (user='(undef)')

Please Enter Your Password: 12345
```

Copyright © 2010 Internetwork Expert

```
AAA/AUTHEN(2250436155): Status=GETPASS
AAA/AUTHEN/CONT (2250436155): Method=ENABLE
AAA/AUTHEN(2250436155): password incorrect
AAA/AUTHEN(2250436155): Status=FAIL
AAA/MEMORY: free_user (0x85646E6C) user='NULL' ruser='NULL'
port='tty66' rem_addr='150.1.1.1' authen_type=ASCII service=ENABLE
priv=15 vrf= (id=0)en

% Access denied
```

Try authenticating again, entering the correct password this time. Note that the system tries TACACS+ method again and then switches to the "enable" method. This time passwords match and the reply is "PASS".

```
Rack1R1>enable

AAA: parse name=tty66 idb type=-1 tty=-1
AAA: name=tty66 flags=0x11 type=5 shelf=0 slot=0 adapter=0 port=66
channel=0
AAA/MEMORY: create_user (0x848DC170) user='NULL' ruser='NULL' ds0=0
port='tty66' rem_addr='150.1.1.1' authen_type=ASCII service=ENABLE
priv=15 initial_task_id='0', vrf= (id=0)
AAA/AUTHEN/START (649338587): port='tty66' list='' action=LOGIN
service=ENABLE
AAA/AUTHEN/START (649338587): using "default" list
AAA/AUTHEN/START (649338587): Method=tacacs+ (tacacs+)
TAC+: send AUTHEN/START packet ver=192 id=649338587
AAA/AUTHEN(649338587): Status=ERROR

Please Enter Your Password:cisco

AAA/AUTHEN/START (649338587): Method=ENABLE
AAA/AUTHEN(649338587): Status=GETPASS
AAA/AUTHEN/CONT (649338587): continue_login (user='(undef)')
AAA/AUTHEN(649338587): Status=GETPASS
AAA/AUTHEN/CONT (649338587): Method=ENABLE
AAA/AUTHEN(649338587): Status=PASS
AAA/MEMORY: free_user (0x848DC170) user='NULL' ruser='NULL'
port='tty66' rem_addr='150.1.1.1' authen_type=ASCII service=ENABLE
priv=15 vrf= (id=0)

Rack1R1#
```

In the debugging output above, note that username is "null" or "undef" when you log in via the VTY line. This is because we used the "line" password for authentication and never supplied a username.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.2 AAA Exec Authorization

- Authorize exec privilege levels using the remote TACACS+ server.
- Ensure that console line logins are also authorized. The console line should authorize users with the TACACS+ method first and then the local database configuration.
- User "ADMIN" should be granted a privilege level of 7 if authenticated.
- Use the VTY line settings to authorize any authenticated users in the event the TACACS+ server fails.
- Ensure that if the TACACS+ server is unavailable, authenticated users logging in via VTY lines are placed in privilege level 15.

Configuration

```
R1:
!
! Enable EXEC authorization on the console line
!
aaa authorization console
aaa authorization exec default none
aaa authorization exec CONSOLE group tacacs+ local
aaa authorization exec VTY group tacacs+ if-authenticated
!
username ADMIN privilege 7 password 0 CISCO
!
line con 0
 authorization exec CONSOLE
!
line vty 0 4
 privilege level 15
 password cisco
 authorization exec VTY
 login authentication VTY
```

Copyright © 2010 Internetwork Expert

Verification

Note

Authorization is a procedure for granting certain rights to a process, or granting a permission to perform a certain action. The authorization procedure is only possible for authenticated entities. The identity of a subject is used to look up the policy and determine the permissions. This is why authentication always precedes authorization. In some cases, it is possible to grant some rights to unidentified subjects.

The goal of exec authorization is assigning a privilege level (0-15) to a logged in user. You configure an exec authorization list using the command:

```
aaa authorization exec {default|<NAME>} <Method List>
```

As with authentication, you can define a default list (which is used system wide) or apply a specific list per terminal line. Generally, there are three methods to obtain authorization information:

1) Consult a remote AAA server and download the user attributes. TACACS+ performs this procedure as a separate operation, but RADIUS has no explicit authorization state, and returns authorization information in authentication replies. Here is an example of using TACACS+ as the source of the required information:

```
aaa authorization exec default group tacacs+
```

2) Consult the local username database, looking for the privilege level assigned to the authenticated user:

```
aaa authorization exec default local
```

3) Use default settings, for example, the default privilege level assigned to the terminal line, if the authorization configuration permits. This is commonly used when you disable authorization (method “none”) or authorize settings for any authenticated users (method “if-authenticated”). Note the difference between the method “none” and “if-authenticated” from the following example:

Copyright © 2010 Internetwork Expert

Scenario 1:

```
aaa authentication login default tacacs+ none
aaa authorization exec default none
!
line console 0
 privilege level 15
```

Scenario 2:

```
aaa authentication login default tacacs+ none
aaa authorization exec default if-authenticated
!
line console 0
 privilege level 15
```

In the first case, if the TACACS+ server is not available, the router will allow incoming console connections without authentication. Since there is no exec authorization, the user will be granted the exec shell with privilege 15. In the second case, if the TACACS+ server is not available, the system grants access without authentication but fails authorization of exec shell.

Thus, the difference between "none" and "if-authenticated" authorization cases is that the former always applies the desired authorization parameters without any verification. The latter requires the user to be authenticated, but does not consult the user database to check authorization attributes.

By default, exec authorization is set to "none", so you may need to change it to accomplish your needs. Also, note that IOS routers by default do not authorize exec sessions on the console line. On the contrary, Catalyst IOS always authorizes the exec shell, even on the console line. Therefore, if you disable console authentication in the Catalyst switch, make sure you never apply an AAA authorization list to the console (explicitly or using the default settings). You may enable console exec authorization in IOS routers using the command **`aaa authorization console`**.

Copyright © 2010 Internetwork Expert

For our sample task, first verify the authentication process by trying to log in to the console line.

```
Rack1R1 con0 is now available



Press RETURN to get started.


This system requires you to identify yourself.

Please Enter Your ID:ADMIN
Please Enter Your Password:CISCO

Rack1R1#show privilege
Current privilege level is 7
```

Note that the privilege level assigned to the user is seven, as specified by the local configuration (username command). The router first tries to use TACACS+ for exec authorization and falls back to the local configuration since the TACACS+ server is not reachable.

Now enable AAA debugging and try logging in via a VTY line.

```
Rack1R1#debug aaa authentication
AAA Authentication debugging is on

Rack1R1#telnet 150.1.1.1
Trying 150.1.1.1 ... Open

AAA/BIND(0000000D): Bind i/f
AAA/AUTHEN/LOGIN (0000000D): Pick method list 'VTY'
This system requires you to identify yourself.

Please Enter Your Password:cisco
AAA/AUTHEN/LINE(0000000D): GET_PASSWORD
```

The router attempts to contact the TACACS+ server, fails, and uses the second method, the line password.

```
AAA/AUTHEN/LINE(0000000D): PASS
AAA/AUTHOR (0000000D): Method=None for method list id=00000000. Skip
author

Rack1R1>enable
```

Copyright © 2010 Internetwork Expert

Now try entering the wrong password for enable authentication. Note that the system tries the TACACS+ method first, but it returns the "ERROR" message. Thus, the system tries using the enable password for authentication, but the password entered does not match. Note the "Status=FAIL" response when you enter the wrong password.

```
AAA: parse name=tty66 idb type=-1 tty=-1
AAA: name=tty66 flags=0x11 type=5 shelf=0 slot=0 adapter=0 port=66
channel=0
AAA/MEMORY: create_user (0x85646E6C) user='NULL' ruser='NULL' ds0=0
port='tty66' rem_addr='150.1.1.1' authen_type=ASCII service=ENABLE
priv=15 initial_task_id='0', vrf= (id=0)
AAA/AUTHEN/START (2250436155): port='tty66' list='' action=LOGIN
service=ENABLE
AAA/AUTHEN/START (2250436155): using "default" list
AAA/AUTHEN/START (2250436155): Method=tacacs+ (tacacs+)
TAC+: send AUTHEN/START packet ver=192 id=-2044531141
AAA/AUTHEN(2250436155): Status=ERROR

AAA/AUTHEN/START (2250436155): Method=ENABLE
AAA/AUTHEN(2250436155): Status=GETPASS
AAA/AUTHEN/CONT (2250436155): continue_login (user='(undef)')

Please Enter Your Password: 12345

AAA/AUTHEN(2250436155): Status=GETPASS
AAA/AUTHEN/CONT (2250436155): Method=ENABLE
AAA/AUTHEN(2250436155): password incorrect
AAA/AUTHEN(2250436155): Status=FAIL
AAA/MEMORY: free_user (0x85646E6C) user='NULL' ruser='NULL'
port='tty66' rem_addr='150.1.1.1' authen_type=ASCII service=ENABLE
priv=15 vrf= (id=0)en

% Access denied
```

Copyright © 2010 Internetwork Expert

Try authenticating again, entering the correct password this time. Note that the system tries the TACACS+ method again and then switches to the "enable" method. This time passwords match and the reply is "PASS".

```
Rack1R1>enable

AAA: parse name=tty66 idb type=-1 tty=-1
AAA: name=tty66 flags=0x11 type=5 shelf=0 slot=0 adapter=0 port=66
channel=0
AAA/MEMORY: create_user (0x848DC170) user='NULL' ruser='NULL' ds0=0
port='tty66' rem_addr='150.1.1.1' authen_type=ASCII service=ENABLE
priv=15 initial_task_id='0', vrf= (id=0)
AAA/AUTHEN/START (649338587): port='tty66' list='' action=LOGIN
service=ENABLE
AAA/AUTHEN/START (649338587): using "default" list
AAA/AUTHEN/START (649338587): Method=tacacs+ (tacacs+)
TAC+: send AUTHEN/START packet ver=192 id=649338587
AAA/AUTHEN(649338587): Status=ERROR

Please Enter Your Password:cisco

AAA/AUTHEN/START (649338587): Method=ENABLE
AAA/AUTHEN(649338587): Status=GETPASS
AAA/AUTHEN/CONT (649338587): continue_login (user='(undef)')
AAA/AUTHEN(649338587): Status=GETPASS
AAA/AUTHEN/CONT (649338587): Method=ENABLE
AAA/AUTHEN(649338587): Status=PASS
AAA/MEMORY: free_user (0x848DC170) user='NULL' ruser='NULL'
port='tty66' rem_addr='150.1.1.1' authen_type=ASCII service=ENABLE
priv=15 vrf= (id=0)

Rack1R1#show privilege
Current privilege level is 15
```

In the debugging output above note that the username is always "null" or "undef" when you log in via the VTY line. This is because we used the "line" password for authentication and never supplied a username.

Copyright © 2010 Internetwork Expert

11.3 AAA Local Command Authorization

- Ensure that the user "ADMIN" can use RIP debugging commands and can disable any currently active debugging using a single command.
- The same user should be able to configure any interface IP settings and administratively enable or disable any of these interfaces.
- Ensure the user can see their permitted commands in their running configuration.

Configuration

```
R1:
privilege exec level 7 configure terminal
privilege exec level 7 undebug all
privilege exec level 7 show running-config
privilege exec level 7 debug ip rip
!
privilege configure level 7 interface
!
privilege interface level 7 shutdown
privilege interface level 7 no shutdown
privilege interface all level 7 ip
```

Verification

Note

IOS allows configuring command authorization by using the local configuration database. Command authorization permits groups of users to use specific commands. IOS also supports remote command authorization with the TACACS+ protocol, but this is out of the scope of the CCIE R&S lab exam.

Local command authorization uses the concept of privilege levels. There are sixteen levels supported, 0 to 15. Every next level supports the commands found in all previous levels, e.g. privilege 5 includes levels 0-5, and privilege 15 includes levels 0-15. By default, IOS has three privilege levels pre-configured:

Level 0 - just a few basic commands, such as **`enable`**, **`login`**, and **`exit`**

Level 1 - the default exec user level; has some show commands available, but no configuration commands

Level 15 - the maximum privilege level, also known as privileged mode or enable mode; includes all the commands available in IOS

Copyright © 2010 Internetwork Expert

In order to create custom command sets, you can perform one of the following:

1) Assign some level 15 commands to level 1, effectively making them available to all users that may log in to the router (if they use the default privilege level settings). You may want to use this option if you need to allow all users the use of some special features, e.g. using certain debug commands.

2) Re-assign some commands from level 1 to a higher level, thus disallowing all unprivileged users the use of this command. For example, you may want to disallow the use of the **`show ip access-list`** command for all default privilege users.

3) Assign some level 15 commands to a new custom level, e.g. level 7. By doing this, you still make commands available to level 15, but do not allow any user with the default privilege to use them. After that, you can assign the custom privilege level to a specific user, allowing the use of some privileged commands to this particular user only.

To understand the command authorization process, you need to remember that at all times the IOS exec shell works in one of the interpreter modes. The two most well known modes are "exec mode" and "global configuration mode". The interpreter's mode is shown by default through the router's prompt, such as **`router#`** (exec) or **`router(config)#`** (global configuration). In addition to that, the shell contains many other interpreter modes, such as "interface configuration", "vpdn configuration", "ip extended ACL", "map-class", and so on. Each mode has its own subset of commands, which are only visible in the particular mode.

In order to see how IOS performs authorization, you need to understand the generic command syntax as well. Each command generally has the following structure:

`command sub-command [arguments] [argument-values] [options]`

Here *command* is the first sub-string you send to the interpreter, for example, "**`ip`**" in the "**`ip address`**" command under interface configuration mode. The *sub-command* field may be present in some commands, e.g. "**`ip proxy-arp`**", "**`compress stac`**" etc. The *arguments* cover all named parameters that may take values and that are mandatory. For example, in the "**`ip address`**" command, the "**`address`**" field is an argument and it may take a value such as "**`1.2.3.4`**". The *options* may cover various command attributes that are not mandatory.

Copyright © 2010 Internetwork Expert

From the local command authorization standpoint, you can only match the "non-variable" or "mandatory" fields such as "command", "sub-command", and "arguments". The system will automatically allow any argument values and options if the command that user enters match the configured pattern.

The syntax to re-assign a particular command is as follows:

```
privilege <mode> level <level> <command>
```

This command instructs the shell to assign the command matching the string "command" to the level specified by the "level" argument. Note that the match is performed against all mandatory parts of the command that a user enters in a particular exec mode. For example, if you assigned just the command **snmp-server** to level 7 but not the command **snmp-server host**, then a user will not be able to configure the SNMP traps destination, since "host" is a mandatory (non-optional) part of the command. The following features help you in configuring local command authorization:

1) When you enter commands as a shortcut, such as **privilege exec level 7 conf t**, the shell expands it to the full names, for example to **privilege exec level 7 configure terminal**

2) When you allow a compound command to a particular level, e.g. **privilege interface level 7 ip address** then the shell automatically adds another line allowing the first "sub-component" of the compound command, e.g. adds a **privilege interface level 7 ip** command.

3) You can use the special keyword "all" to allow any sub-command or argument behind a specific keyword. For example, the command **privilege interface all level 7 ip** allows all "ip" subcommands in the interface configuration mode.

The last thing to keep in mind is how the local command authorization interoperates with the **show running-config**. A special feature is that a user may only see the command they are authorized to use. For example, if a user is authorized to use the **interface** command and all "ip" subcommands, then the **show running-config** command will only display those specific commands.

Copyright © 2010 Internetwork Expert

Verification for this task consists of checking the commands that you are allowed to use:

```
Rack1R1 con0 is now available

Press RETURN to get started.

This system requires you to identify yourself.

Please Enter Your ID:ADMIN
Please Enter Your Password:CISCO

Rack1R1#show privilege
Current privilege level is 7

Rack1R1#?
Exec commands:
  access-enable    Create a temporary Access-List entry
  access-profile   Apply user-profile to interface
  call             Voice call
  clear            Reset functions
  configure        Enter configuration mode
  <snip>

Rack1R1#debug ?
  all           Enable all debugging
  call          Call Information
  call-mgmt     Call Management
  ces-conn      Connection Manager CES Client Info
  conn          Connection Manager information
  dspapi        Generic DSP API
  flow-sampler  Debug flow sampler
  hpi           HPI (54x) DSP messages
  ip            IP information
  ncia          Native Client Interface Architecture (NCIA) events
  network       Network related debugging
  rscmon        Resource Monitor

Rack1R1#debug ip rip ?
  <cr>

Rack1R1#undebug all
All possible debugging has been turned off

Rack1R1#conf ?
  terminal  Configure from the terminal
  <cr>
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#conf terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#?
Configure commands:
  atm          Enable ATM SLM Statistics
  call         Configure Call parameters
  default      Set a command to its defaults
  end          Exit from configure mode
  exit         Exit from configure mode
  help         Description of the interactive help system
  interface    Select an interface to configure
  no           Negate a command or set its defaults
  oer          Optimized Exit Routing configuration submodes

Rack1R1(config)#interface fastEthernet 0/0
Rack1R1(config-if)#?
Interface configuration commands:
  default   Set a command to its defaults
  exit      Exit from interface configuration mode
  help      Description of the interactive help system
  ip        Interface Internet Protocol config commands
  no        Negate a command or set its defaults
  shutdown  Shutdown the selected interface

Rack1R1(config-if)#ip ?
Interface IP configuration subcommands:
  access-group         Specify access control for packets
  accounting           Enable IP accounting on this interface
  address              Set the IP address of an interface
  admission            Apply Network Admission Control
  auth-proxy           Apply authentication proxy
  authentication       authentication subcommands
  bandwidth-percent    Set EIGRP bandwidth limit
  bgp                  BGP interface commands
  <snip>
```

Copyright © 2010 Internetwork Expert

Now inspect the running configuration contents:

```
Rack1R1#show run
Building configuration...

Current configuration : 527 bytes
!
! Last configuration change at 12:31:51 UTC Wed Sep 24 2008 by ADMIN
! NVRAM config last updated at 05:01:32 UTC Wed Sep 24 2008
!
boot-start-marker
boot-end-marker
!
!
!
!
!
interface Loopback0
 ip address 150.1.1.1 255.255.255.0
 ip rip advertise 10
!
interface FastEthernet0/0
 ip address 155.1.146.1 255.255.255.0
 ip rip advertise 10
!
interface Serial0/0
 ip address 155.1.0.1 255.255.255.0
 ip rip advertise 10
 ip split-horizon
!
interface Serial0/1
 ip address 155.1.13.1 255.255.255.0
 ip rip advertise 10
```

Note that you only see the **interface** and **ip** commands in the running configuration.

Copyright © 2010 Internetwork Expert

11.4 Traffic Filtering using Standard Access-Lists

- Using the minimum amount of ACL lines, allow access to R1 from the Loopback 0 subnets of R2 through R6 that have an even 3rd octet.
- Deny access from any other "150.X.0.0/16" subnets.
- Any hosts from the "155.X.0.0/16" subnets should still be able to reach R1.
- Apply the access-list inbound on all interfaces of R1.

Configuration

```
R1:
access-list 1 permit 150.1.0.0 0.0.6.255
access-list 1 permit 155.1.0.0 0.0.255.255
!
interface FastEthernet 0/0
 ip access-group 1 in
!
interface Serial 0/1
 ip access-group 1 in
!
interface Serial 0/0
 ip access-group 1 in
```

Verification

Note

Standard ACLs only allow you to match source IP addresses based on "base" IP address and wildcard mask. Because of that "aggregate" behavior, standard ACLs are commonly configured at network nodes close to the "protected" object. One very common task is finding a required "base" IP address and wildcard mask pair based on set of requirements. Let us look at three different examples to illustrate how they could be solved.

Example 1 (the task):
Configure the IP/Wildcard Mask pair to match IP addresses in subnets 155.1.Y.0/24 where Y is an even number and is greater than or equal to 1 and less than or equal to 6.

Note that the 4th octet of the subnets may take any value from 0 to 255, so we just set the corresponding part of the wildcard mask to 255 (all 1's).

Copyright © 2010 Internetwork Expert

Step 1:
We start by finding the fixed part of the subnets range. Since Y changes from 1 to 6, only the last 3 bits of the 3rd octet may vary. Thus, the topmost 6 bits of Y are fixed at zero. For the fixed part of the base IP address, we set wildcard bits to zeroes (meaning it does not change). Now write down all possible values that Y may take:

2=00000010
4=00000100
6=00000110

Step 2:
Find the part of the "bit-vector" that changes between different values. In this case, this part contains bits 1 and 2 (counting from the right and starting at zero). Create a mask vector, putting 1's in the positions where bit values in data vectors may change:

m=00000110=6

Essentially, this is the wildcard mask for the third octet.

Step 3:
Take into account the extra values covered by the mask. Note that the mask always cover 2^N values, where N is the number of 1's in the mask. In our case, we had three values, but the mask contains two non-zero bits resulting in four possible values. Effectively, our wildcard mask cover one extra bit vector:

0=00000000

Strictly speaking you may need a special ACL entry to deny the corresponding subnet. However, in our topology, there is no router with subnet 155.1.0.0/24, so you may ignore this extra value.

Step 4:
Compute the "base" 3rd octet value, taking the common part of all the bit vectors. Effectively you may do so by performing a logical bitwise AND over all values:

2 AND 4 AND 6 = 0.

Thus the resulting IP and Wildcard Mask pair is:

155.1.0.0 0.0.6.255

Copyright © 2010 Internetwork Expert

The resulting ACL may look like:

deny 155.1.0.0 0.0.0.255
permit 155.1.0.0.6.255

Note that the first line is NOT necessary in our case.

Example 2:
Configure the IP/Wildcard Mask pair to match IP addresses in subnets 155.1.Y.0/24 where Y is an odd number and is greater than or equal to 1 and less than or equal to 6.

Note that the 4th octet of the subnets may take any value from 0 to 255, so we just set the corresponding part of the wildcard mask to 255 (all 1's).

Step 1:
We start by finding the fixed part of the subnets range. Since Y changes from 1 to 6, only the last 3 bits of the 3rd octet may vary. Thus, the topmost 6 bits of Y are fixed to zero. For the fixed part of the base IP address, we set wildcard bits to zeroes (meaning it does not change). Now write down all possible values that Y may take:

1=00000001
3=00000011
5=00000101

Step 2:
Find the part of the "bit-vector" that changes between the different values. In this case, this part contains bits 1 and 2 (counting from the right and starting at zero). The zero bit is fixed to "1". Create a mask vector, putting 1's in the positions where bit values in data vectors may change:

m=00000110=6

Essentially, this is the wildcard mask for the third octet.

Step 3:
Take into account the extra values covered by the mask. Note that the mask always cover 2^N values where N is the number of 1's in the mask. In our case, we had three values, but the mask contains two non-zero bits resulting in four possible values. Effectively, our wildcard mask covers one extra bit vector:

7=00000111

Copyright © 2010 Internetwork Expert

Strictly speaking we need a special ACL entry to deny the corresponding subnet, since it is a part of our topology.

Step 4:
Compute the "base" 3rd octet value using the common part of all the bit vectors. Effectively, you may do so by performing a logical bitwise AND over all the values:

1 AND 3 AND 5 = 1.

Thus the resulting IP and wildcard mask pair is:

155.1.1.0 0.0.6.255

and the resulting ACL looks like:

deny 155.1.7.0 0.0.0.255
permit 155.1.1.0 0.0.6.255

In cases where octet values are widely separated, the resulting ACL may contain too many "deny" statements and it may be more effective to use just the entries matching specific values. In our case, the resulting ACL needs to also permit the 155.1.0.0/16 subnet as well.

Note that standard ACLs do not allow you to specify the protocol values and effectively permit any IP protocol for matching sources. The general use of standard ACLs is to route filtering and prefix matching in routing protocol configuration. Note that standard ACLs may be named as well as numbered (1-99).

To verify your configuration, first make sure that the IP access list applies to the interfaces:

```
Rack1R1#sh ip interface serial 0/0 | inc acce
  Outgoing access list is not set
  Inbound  access list is 1
  IP access violation accounting is disabled

Rack1R1#sh ip interface serial 0/1 | inc acce
  Outgoing access list is not set
  Inbound  access list is 1
  IP access violation accounting is disabled

Rack1R1#sh ip interface FastEthernet 0/0 | inc acce
  Outgoing access list is not set
  Inbound  access list is 1
  IP access violation accounting is disabled
```

Copyright © 2010 Internetwork Expert

Verify that you can only reach R1 from the Loopbacks of R2, R4 and R6:

```
Rack1R2#ping 150.1.1.1 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.1.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.2.2
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 84/86/88 ms

Rack1R3#ping 155.1.1.1 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.1.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.3.3
....
Success rate is 0 percent (0/5)

Rack1R4#ping 150.1.1.1 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.1.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.4.4
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1R5#ping 150.1.1.1 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.1.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.5.5
U.U.U
Success rate is 0 percent (0/5)

Rack1R6#ping 150.1.1.1 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.1.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.6.6
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1SW1#ping 150.1.1.1 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.1.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.7.7
UU.UU
Success rate is 0 percent (0/5)

Rack1SW2#ping 150.1.1.1 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.1.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.8.8
.U.U.
Success rate is 0 percent (0/5)
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.5 Traffic Filtering using Extended Access-Lists

- Configure R4 to permit any TCP traffic destined for R5 and sourced from the Loopback0 subnets. Ensure R4 allows returning TCP packets for those connections.
- VLAN 146 contains servers running FTP, WWW, SMTP and DNS applications.
- Configure R4 to allow access to those applications from behind R5.
- Allow DNS zone file transfers in addition to DNS access.
- Ensure you only permit active FTP responses from servers in VLAN 146.
- Permit the exchange of RIP routing updates.
- Permit IOS “traceroute” and “ping” commands to function across R4.
- Ensure that the Path MTU discovery procedure works successfully across your firewall.
- Deny all ICMP “unreachable” messages with your configuration.
- Deny all other traffic.

Configuration

```
R4:
no ip access-list extended INBOUND
ip access-list extended INBOUND
remark ==
remark == Active FTP uses TCP ports 20 and 21
remark ==
permit tcp any 155.1.146.0 0.0.0.255 range 20 21
remark ==
remark == WWW and SMTP use port numbers 80 and 25
remark ==
permit tcp any 155.1.146.0 0.0.0.255 eq 80
permit tcp any 155.1.146.0 0.0.0.255 eq 25
remark ==
remark == DNS uses UDP port 53 and TCP port 53 for zone xfers
remark ==
permit udp any 155.1.146.0 0.0.0.255 eq 53
permit tcp any 155.1.146.0 0.0.0.255 eq 53
remark ==
remark == Established TCP sessions are traffic returning back
remark ==
permit tcp any 150.1.0.0 0.0.255.255 established
remark ==
remark == Inbound traceroute UDP probes
remark ==
permit udp any any range 33434 33474
remark ==
remark == Backscatter ICMP traffic for outbound traceroute UDP probes
remark ==
permit icmp any any port-unreachable
permit icmp any any time-exceeded
```

Copyright © 2010 Internetwork Expert

```
remark ==
remark == ICMP message used by PMTU discovery: Packet too big and DF
bit set
remark ==
permit icmp any any packet-too-big
remark ==
remark == Ping operation packets
remark ==
permit icmp any any echo
permit icmp any any echo-reply
remark ==
remark == RIP routing updates
remark ==
permit udp any any eq 520
remark ==
remark == Deny and log any other packets
remark ==
deny ip any any log
!
no ip access-list extended OUTBOUND
ip access-list extended OUTBOUND
remark ==
remark == Active FTP uses TCP ports 20 and 21
remark ==
permit tcp 155.1.146.0 0.0.0.255 range 20 21 any
remark ==
remark == WWW and SMTP use port numbers 80 and 25
remark ==
permit tcp 155.1.146.0 0.0.0.255 eq 80 any
permit tcp 155.1.146.0 0.0.0.255 eq 25 any
remark ==
remark == DNS uses UDP por 53 and TCP port 53 for zone xfers
remark ==
permit udp 155.1.146.0 0.0.0.255 eq 53 any
permit tcp 155.1.146.0 0.0.0.255 eq 53 any
remark ==
remark == TCP traffic from Loopback0 subnets
remark ==
permit tcp 150.1.0.0 0.0.255.255 any
remark ==
remark == Outbound traceroute UDP probes
remark ==
permit udp any any range 33434 33474
remark ==
remark == Backscatter ICMP traffic for inbound traceroute UDP probes
remark ==
permit icmp any any port-unreachable
permit icmp any any time-exceeded
remark ==
remark == ICMP message used by PMTU discovery: Packet too big and DF
bit set
remark ==
permit icmp any any packet-too-big
```

Copyright © 2010 Internetwork Expert

```
remark ==
remark == Ping operation packets
remark ==
permit icmp any any echo
permit icmp any any echo-reply
remark ==
remark == Deny and log any other packets
remark ==
deny ip any any log
!
interface Serial 0/0/0
 ip access-group INBOUND in
 ip access-group OUTBOUND out
!
interface Serial 0/1/0
 ip access-group INBOUND in
 ip access-group OUTBOUND out
```

Verification

Note

Extended ACLs allow the matching of source/destination IP addresses and source/destination ports. In addition to that, you can match ICMP message types, DSCP values, port ranges, TCP flags, and so on. Essentially, extended ACLs permit configuration of a very flexible stateless packet filter. Due to the stateless nature of the filtering, for each "outbound" rule you need a matching "inbound" rule, permitting the returning traffic. Thus, inbound and outbound extended access-lists usually mirror each other (e.g. port numbers swapped between source and destination IP addresses).

Extended ACLs allow limited "emulation" of stateful behavior for TCP connections. Specifically, by using the **`established`** keyword, you permit all TCP packets that have the "ACK" bit set. This does not include the initial "SYN-only" packet, and thus permits only the packets that are part of an established session.

In order to configure an extended ACL successfully, you need to know the details of application networking logic. At the very least, this includes the TCP/UDP port numbers used by the application. You may also need to know some additional details, for example, how active FTP differs from passive FTP, or how the traceroute utility works, etc. Most of the common ports can be found using the shell prompt and typing the question mark after **`permit tcp any any eq`**. You may also learn application ports by using the commands: **`show ip port-map`** or **`show ip nbar port-map`**.

Copyright © 2010 Internetwork Expert

Another special feature to keep in mind is functionality of outbound access-lists. The router's locally generated traffic is not subject to match against the outbound access-lists, only transit traffic matches outbound lists.

Next, we are going to discuss two applications commonly used in practice. The first one is FTP. Per the design, FTP uses two TCP connections, a control connection to send commands, and a data connection to transfer files. The control connection always uses a destination port 21. The data connection may work in one of two modes: active or passive.

1) In active mode, when a client issues a command requiring data transfer, the server instructs the client to listen on an ephemeral port (above 1024) and initiates a connection to the client sourced from port number 20. In this mode, the server connects to a client on a dynamic destination port.

2) In passive mode, when the client needs to transfer a file, the server opens a new ephemeral port (above 1024) and reports it to the client. The client then connects to the ephemeral port and the data transfer begins. In this mode, the client connects to the server on a dynamic destination port. Note that port 20 is *not used* in passive mode.

The next application to discuss is the traceroute utility. There are different variants of the traceroute utilized by various operating systems. IOS uses the so-called UNIX variant. In this variant, the client sends UDP probes with increasing TTL values starting at the TTL of 1. The client sends probes to incrementing UDP ports starting at 33434 for the first hop. The destination IP address in all packets is the target IP address.

1) If the next hop is not the ultimate destination, the receiving device checks the TTL field. If the TTL field expires, the device sends an ICMP time exceeded message sourced from its own IP address. The source node learns the hop number based on the UDP port number, encapsulated as part of the payload in the ICMP response message.

2) If the hop is the ultimate destination, it sends an ICMP port unreachable message for the port range selected specifically not to match any application. The source node learns the hop count based on the port number.

By default the traceroute utility only probes up to 30 hops, so the default UDP port range is 33434…33464.

Copyright © 2010 Internetwork Expert

Finally, the Path MTU Discovery (PMTUD) process relies on the ICMP message "packet too big", also known as "fragmentation required but DF bit set". Note that you can permit all ICMP unreachable messages using the **unreachable** keyword, or permit any ICMP message selectively.

During the creation of access-lists, use remarks extensively in real life configurations to make management and readability of function easier. In the lab exam, first try typing access-lists in notepad, it is easier to spot mistakes that way. In the lab, it's also a good idea to add the additionally **deny ip any any log** in the end of your access list to catch any unnoticed traffic you may have not permitted.

In this verification, we will configure R6 to simulate some of the services, for example, HTTP and DNS:

```
R6:
ip http server
ip dns server
ip host TEST 150.1.6.6
```

Verify that you can only reach R1 from the Loopbacks of R2, R4 and R6:

```
Rack1R5#ping 155.1.146.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/17/20 ms

Rack1R5#traceroute 155.1.146.6

Type escape sequence to abort.
Tracing the route to 155.1.146.6

  1 155.1.45.4 12 msec
    155.1.0.4 16 msec
    155.1.45.4 8 msec
  2 155.1.146.6 20 msec *  16 msec

Rack1R5#telnet 155.1.146.6 80
Trying 155.1.146.6, 80 ... Open

Rack1R5#disc 1
Closing connection to 155.1.146.6 [confirm]
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#ip name-server 155.1.146.6
Rack1R5(config)#ip domain-lookup
Rack1R5(config)#^Z

Rack1R5#ping TEST

Translating "TEST"...domain server (155.1.146.6) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.6.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/38/40 ms
```

Verify that you can originate TCP connections only from a Loopback 0 interface, but not from the 155.1.0.0/16 subnets. After this verification, ensure that traceroute and ping work from behind R4.

```
Rack1R6#telnet 150.1.5.5
Trying 150.1.5.5 ...
% Destination unreachable; gateway or host down

Rack1R6#telnet 150.1.5.5 /source loopback 0
Trying 150.1.5.5 ... Open


User Access Verification

Password: cisco
Rack1R5>

Rack1R6#ping 150.1.5.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.5.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/38/40 ms

Rack1R6#traceroute 150.1.5.5

Type escape sequence to abort.
Tracing the route to 150.1.5.5

  1 155.1.146.4 0 msec 4 msec 0 msec
  2 155.1.0.5 16 msec *  16 msec
Rack1R6#
```

Copyright © 2010 Internetwork Expert

Be sure to check your access-list match counters:

```
Rack1R4#sh ip access-lists
Extended IP access list INBOUND
    10 permit tcp any 155.1.146.0 0.0.0.255 range ftp-data ftp
    20 permit tcp any 155.1.146.0 0.0.0.255 eq www (6 matches)
    30 permit tcp any 155.1.146.0 0.0.0.255 eq smtp
    40 permit udp any 155.1.146.0 0.0.0.255 eq domain (1 match)
    50 permit tcp any 155.1.146.0 0.0.0.255 eq domain
    60 permit tcp any 150.1.0.0 0.0.255.255 established (20 matches)
    70 permit udp any any range 33434 33474 (6 matches)
    80 permit icmp any any port-unreachable (2 matches)
    90 permit icmp any any time-exceeded
    100 permit icmp any any packet-too-big
    110 permit icmp any any echo (10 matches)
    120 permit icmp any any echo-reply (5 matches)
    130 permit udp any any eq rip (8094 matches)
    140 deny ip any any log
Extended IP access list OUTBOUND
    10 permit tcp 155.1.146.0 0.0.0.255 range ftp-data ftp any
    20 permit tcp 155.1.146.0 0.0.0.255 eq www any (3 matches)
    30 permit tcp 155.1.146.0 0.0.0.255 eq smtp any
    40 permit udp 155.1.146.0 0.0.0.255 eq domain any (1 match)
    50 permit tcp 155.1.146.0 0.0.0.255 eq domain any
    60 permit tcp 150.1.0.0 0.0.255.255 any (21 matches)
    70 permit udp any any range 33434 33474 (3 matches)
    80 permit icmp any any port-unreachable (2 matches)
    90 permit icmp any any time-exceeded
    100 permit icmp any any packet-too-big
    110 permit icmp any any echo (5 matches)
    120 permit icmp any any echo-reply (10 matches)
    130 deny ip any any log (1 match)
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.6 Traffic Filtering using Reflexive Access-Lists

- Configure R5 to permit TCP, UDP, and ICMP packets sourced from behind the VLAN 58 interface.
- Do not create explicit access-list entries to permit returning TCP, UDP, and ICMP traffic.
- Ensure that Telnet and ICMP traffic originated from the router are also permitted by the access-list, but do not create any entries in the interface access-lists.
- Ensure RIP routing traffic is also permitted by your configuration.

Configuration

```
R5:
no ip access-list extended OUTBOUND
ip access-list extended OUTBOUND
 permit tcp any any reflect MIRROR
 permit icmp any any reflect MIRROR
 permit udp any any reflect MIRROR
 deny ip any any log
!
no ip access-list extended INBOUND
ip access-list extended INBOUND
 permit udp any any eq 520
 evaluate MIRROR
!
interface Serial 0/0/0
 ip access-group INBOUND in
 ip access-group OUTBOUND out
!
interface Serial 0/1/0
 ip access-group INBOUND in
 ip access-group OUTBOUND out
!
! Local policy to divert telnet connections over the loopback
!
no ip access-list extended LOCAL_TRAFFIC
ip access-list extended LOCAL_TRAFFIC
 permit tcp any any eq 23
 permit icmp any any echo
!
route-map DIVERT_LOCAL
 match ip address LOCAL_TRAFFIC
 set ip next-hop 150.1.5.254
!
ip local policy route-map DIVERT_LOCAL
```

Copyright © 2010 Internetwork Expert

Verification

Note

Reflexive access-lists add "stateful" behavior to the IOS packet filtering features. The overall idea is simple: when a packet matches an ACL entry, instruct the router to create a "mirrored" entry in the access list applied in the opposite direction. For example, if a packet hits an inbound ACL, take the source/destination IP addresses and ports, swap them, and add dynamic entries to the outbound access list with the new parameters. This procedure automatically permits returning traffic for symmetric flows. It will not work for "non-standard" TCP applications, which are complicated protocols such as RCP, TFTP, and H323, which open dynamic channels with no way for a router to predict their behavior.

The reflexive access list implementation is as follows. When you configure a regular ACL entry for TCP, UDP, or ICMP traffic, you may add the command "**`reflect <ACL_NAME>`**". When a packet matches this entry, the router adds the dynamic "mirrored" ACL entry to the ACL named <ACL_NAME>. Later, you can add the command "**`evaluate <ACL_NAME>`**" in the access-list applied in the opposite direction. This command instructs the router to evaluate the entries in the dynamically populated ACL to find a match for a returning packet. There is no need to pre-create the dynamically populated ACL, the IOS creates it automatically.

Note that you may only reflect UDP, TCP and ICMP session traffic. All dynamic entries persist as long as the session is active. When a session becomes inactive, the router ages out the corresponding entry in the time specified by the global command:

```
ip reflexive-list timeout <TIMEOUT>
```

The default timeout value is 300 seconds.

Note the important feature that the dynamic access list is only populated when a packet hits an ACL entry. Also recall that locally generated router traffic does not match any outgoing ACLs. Thus, if you use reflexive ACLs, you may need to account for that feature and provide any of the following workarounds:

1) Create a static entry in the inbound access-list to permit the returning routing traffic. This works for protocols like RIP, OSPF, and EIGRP which send broadcast hello packets.

Copyright © 2010 Internetwork Expert

2) Use local policy routing to divert the local traffic across the loopback interface and make it re-enter the router, in effect, becoming transit traffic. This will cause a match for the outgoing ACLs and populate dynamic entries. This works for unicast session traffic, such as Telnet sessions off the router, ICMP packet flows, or BGP connections.

For verification, source ICMP packets off R5 and originate a telnet session. In both cases, look at the dynamically populated access-list MIRROR.

```
Rack1R5#ping 150.1.3.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.3.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 60/61/64 ms

Rack1R5#show ip access-list MIRROR
Reflexive IP access list MIRROR
     permit icmp host 150.1.3.3 host 155.1.0.5  (20 matches) (time left 296)

Rack1R5#telnet 150.1.3.3
Trying 150.1.3.3 ... Open


User Access Verification

Password:
Rack1R3>exit

[Connection to 150.1.3.3 closed by foreign host]

Rack1R5#show ip access-list MIRROR
Reflexive IP access list MIRROR
     permit tcp host 150.1.3.3 eq telnet host 155.1.0.5 eq 12864 (77 matches)
(time left 2)
     permit icmp host 150.1.3.3 host 155.1.0.5  (20 matches) (time left 282)
```

Next, check the total statistics accumulated by all access-lists. Note the matches for RIP updates.

```
Rack1R5#show ip access-list
Extended IP access list INBOUND
    10 permit udp any any eq rip (185 matches)
    20 evaluate MIRROR
Extended IP access list LOCAL_TRAFFIC
    10 permit tcp any any eq telnet (20 matches)
    20 permit icmp any any echo (5 matches)
Reflexive IP access list MIRROR
     permit icmp host 150.1.3.3 host 155.1.0.5  (20 matches) (time left 272)
Extended IP access list OUTBOUND
    10 permit tcp any any reflect MIRROR (20 matches)
    20 permit icmp any any reflect MIRROR (5 matches)
    30 permit udp any any reflect MIRROR
    40 deny ip any any log
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.7 Filtering Fragmented Packets

- Ensure R3 prevents fragmented packets from reaching its local HTTP server.
- Only configure the Frame-Relay interface for this task.

Configuration

```
R3:
no ip access-list extended NO_FRAGMENTS
ip access-list extended NO_FRAGMENTS
 permit udp any any eq rip
 deny ip any any fragments
 permit tcp any any eq 80
!
interface Serial 1/0
 ip access-group NO_FRAGMENTS in
```

Verification

Note

By default, the IP protocol allows packet fragmenting. This feature has been a long time known weakness, exploited by various types of attacks (e.g. ping of death). Overlapping fragments, fragments exceeding the assembly buffer, and fragments arriving out of order, are just a few examples of traffic that can severely degrade end system performance when an attacker sends them at high rates.

Additionally, fragmented packets are often used to bypass IDS or firewall systems. The attacker splits a packet in such a way that the firewall or IDS system is not able to extract information about something like the port numbers. Advanced firewalls and IDS systems support packet stream reassembly, but this procedure may degrade overall security system performance.

Due to these reasons, it is a good practice to avoid traffic fragmentation in your network and protect your servers against fragmented packets. One solution would be to configure matching MTU values and enable the PMTU Discovery process.

The IOS firewall is able to classify IP packets as one of the following:

1) Non-fragmented packets or initial fragments. These packets have a fragment offset of zero and commonly contain some upper-level protocol payload (TCP) to allow the extraction of information about application ports.

Copyright © 2010 Internetwork Expert

2) Non-initial fragments have a non-zero fragment offset and are the remaining parts of a fragmented packets. These packets do not have upper level protocol port information and the IOS firewall cannot match them against ACL entries configured with TCP/UDP port numbers. Instead of matching the port numbers, the firewall uses only Layer 3 information in an ACL entry (e.g. source/destination IP addresses) to match against the source/destination IP addresses in the packet.

This is why any non-initial fragment sourced from 1.1.1.1 to 2.2.2.2 matches the following ACL entry:

```
permit tcp host 1.1.1.1 host 2.2.2.2 eq 80
```

Even though it does not contain any port information, the firewall only matches the source and destination IP addresses for non-initial fragments.

You can match non-initial fragments using the **fragments** keyword in your ACL entry, for example:

```
deny ip any host 2.2.2.2 fragments
permit tcp host 1.1.1.1 host 2.2.2.2 eq 80
```

This configuration ensures that only non-fragmented packets and initial fragments may reach port 80 of the target system.

You can configure R5 so that outgoing TCP packets are fragmented. To accomplish this, ensure that PMTU Discovery is disabled on SW2 and the MTU is set to a minimum value on the Frame-Relay interface of R5. This will force R5 to fragment TCP/IP packets exceeding 68 bytes in size. You may need to remove the inbound ACL on R5 to permit returning traffic for the HTTP connection.

After this configuration, try connecting to R3 on port 80 from SW2 and download a file from the flash memory. The IP+TCP header size is 40 bytes and the payload size is most likely more than 28 bytes (the “GET” command and the URL) so the packets will surely exceed the IP MTU.

```
R3:
ip http server
ip http path flash:

Rack1SW2(config)#no ip tcp path-mtu-discovery
```

Copyright © 2010 Internetwork Expert

```
Rack1R5(config)#interface Serial 0/0/0
Rack1R5(config-if)#ip mtu 68
Rack1R5(config-if)#no ip access-group INBOUND in

Rack1SW2#copy http://admin:cisco@150.1.3.3/c2600-adventerprisek9-
mz.124-10.bin null:
%Error opening http://admin:cisco@150.1.3.3/c2600-adventerprisek9-
mz.124-10.bin (I/O error)
Rack1SW2#
```

The connection times out, and looking at the access-list statistics on R3 you can see exactly why. The reason is the packets are denied.

```
Rack1R3#show ip access-lists
Extended IP access list NO_FRAGMENTS
    10 permit udp any any eq rip (234 matches)
    20 deny ip any any fragments (27 matches)
    30 permit tcp any any eq www (24 matches)
```

Now change the MTU back to normal and try downloading again:

```
Rack1R5(config)#interface Serial 0/0/0
Rack1R5(config-if)#ip mtu 1500

Rack1SW2#copy http://admin:cisco@150.1.3.3/c2600-adventerprisek9-
mz.124-10.bin null:
Loading http://admin:cisco@150.1.3.3/c2600-adventerprisek9-mz.124-
10.bin
```

Now R5 does not fragment the packets and the connection completes normally.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.8 Filtering Packets with Dynamic Access-Lists

- R6 should restrict access to the outside web servers for the users located on VLAN 67.
- Prior to being allowed to connect to a WWW server, a user should log in to the router using the name “ENABLE” and password of “CISCO”.
- The above procedure should only create an access-list entry for the IP address of the authenticated user.
- Alternatively, any user may connect to port 7001 of R6 and enter the password of “CISCO”.
- This procedure should enable any user on VLAN 67 and behind this VLAN to access the WWW servers.
- Close inactive connections after 5 minutes and do not allow sessions of more than 15 minutes in length.
- A user should be able to extend the maximum duration by logging into R6 repeatedly.

Configuration

```
R6:
no aaa new-model
!
! Enable absolute timeout extension
!
access-list dynamic-extended
!
! Access-list with dynamic entries
!
no ip access-list extended 100
ip access-list extended 100
 permit tcp any any eq telnet
 permit tcp any any eq 7001
 permit udp any any eq 520
 dynamic ACCESS timeout 15 permit tcp any any eq 80
 deny ip any any log
!
! Apply the access-list inbound
!
interface FastEthernet 0/0.67
 ip access-group 100 in
!
! Note that password and autocommand are on separate lines
! This is because sometimes IOS thinks "autocommand" is a part
! of the password
!
username ENABLE password CISCO
username ENABLE autocommand access-enable host timeout 5
```

Copyright © 2010 Internetwork Expert

```
!
! First three lines authenticate users against local database
!
line vty 0 3
 login local
!
! Dedicate a special line for line-based authentication
! This line could be accessed on port 7001 (7000+rotary group number)
!
line vty 4
 rotary 1
 password CISCO
 login
 autocommand access-enable timeout 5
```

Verification

Note

Dynamic access-lists (also known as "lock and key") allow special types of entries, activated using the CLI command **access-enable**. Until this command triggers the entries, they are inactive and the IOS ignores them while inspecting an access-list. The **access-enable** command unlocks the dynamic entries, hence the name.

Note that once you execute **access-enable**, it activates all dynamic entries in all access-lists. Commonly, the activation command is bound to a specific user or router-line using the **auto-command** syntax. This results in a specific user logging in to a router and enabling the dynamic access-list entries.

Dynamic access-list entries time out after a period time. There are two timeouts defined for dynamic ACLS:

1) The inactivity timeout; you specify this timeout using the command **access-enable timeout <TIMEOUT>**. This timeout only applies when no packets match the entry for the specified amount of time. The default is no idle timeout (only absolute timeout).

2) Absolute timeout; you specify this timeout under the access-list entry, for example

access-list 100 dynamic <SOME_NAME> timeout <TIMEOUT> …

This is the maximum amount of time that the entry may stay active. After this timeout expires, the user my log in and activate the command again. The default absolute timeout is infinite.

Copyright © 2010 Internetwork Expert

The common implementation of dynamic access-lists looks as follows:

1) Create an extended access-list (either named or numbered) and ensure it permits Telnet (or any other remote access method, for example SSH) to the local router. Populate the access-list with dynamic entries, but make sure it allows remote access to the router.

2) Apply the access-list inbound on the interface controlling user access. Dynamic entries only work in the inbound direction, and you cannot use them in outbound ACLs.

3) Create a local user with **`autocommand access-enable`** or apply this command to terminal lines (or a selected terminal line, using rotary groups). Tune the timeout values, if you need.

4) When a user wants to activate the set of access rules, he or she logs in via Telnet or SSH to the router and authenticates using name and password (or authenticates per the VTY line settings). After that, the **`auto-command`** triggers the dynamic ACL entries and terminates user connections immediately. Now the user may access outside resources per the activated dynamic rules.

By default, if you execute the command **`access-enable`** it activates the entry configured in the access-list. If you want to insert the source IP address of the user logging into the router, use the command **`access-enable host`**. This command replaces the source IP address specification (e.g. **any**) in the dynamic access-list entry with the IP address of the authenticated user.

Another special feature is absolute timeout extension. If you configure a global command **`access-list dynamic-extended`**, then a user configured with the **`access-enable`** auto-command may re-login to the router to extend the *absolute* timeout by the value configured under the dynamic ACL entry. This allows session prolongation by a user, without terminating any existing connections.

Another option is a manual clear feature. It only works with numbered extended ACLs and allows an administrator to deactivate selected dynamic access-list entries manually. The command syntax is

```
clear access-template <ACL-NUMBER> <DYNAMIC-ENTRY-NAME>
<SRC-IP> <SRC-MASK> <DST-IP> <DST-MASK>
```

Copyright © 2010 Internetwork Expert

For example the following command:

clear access-template 100 TEST any 10.0.0.0 0.0.0.255

will clear any dynamic entry cloned from the above template. Note that you do not need to specify the "permit" or "deny" keyword or specify the protocol name.

For verification here, first telnet to R6 directly and log in as "ENABLE" user. This should activate the host entry in the access-list.

```
Rack1SW1#telnet 150.1.6.6
Trying 150.1.6.6 ... Open

User Access Verification

Username: ENABLE
Password: CISCO
[Connection to 150.1.6.6 closed by foreign host]

Rack1R6#show ip access-lists
Extended IP access list 100
    10 permit tcp any any eq telnet (87 matches)
    20 permit tcp any any eq 7001
    30 permit udp any any eq rip (15 matches)
    40 Dynamic ACCESS permit tcp any any eq www
       permit tcp host 155.1.67.7 any eq www
    50 deny ip any any log
```

Now try connecting to the port allowed by the new dynamic entry. Note that as soon as you Telnet through, the inactivity timer starts (300 seconds).

```
Rack1SW1#telnet 150.1.4.4 80
Trying 150.1.4.4, 80 ... Open
GET /

WWW-Authenticate: Basic realm="level_15_access"

401 Unauthorized

[Connection to 150.1.4.4 closed by foreign host]
Rack1SW1#

Rack1R6#show ip access-lists
Extended IP access list 100
    10 permit tcp any any eq telnet (87 matches)
    20 permit tcp any any eq 7001
    30 permit udp any any eq rip (24 matches)
    40 Dynamic ACCESS permit tcp any any eq www
       permit tcp host 155.1.67.7 any eq www (12 matches) (time left
295)
    50 deny ip any any log
```

Copyright © 2010 Internetwork Expert

Now clear the access-template:

```
Rack1R6#clear access-template 100 ACCESS host 155.1.67.7 any
Rack1R6#show ip access-lists 100
Extended IP access list 100
    10 permit tcp any any eq telnet (87 matches)
    20 permit tcp any any eq 7001
    30 permit udp any any eq rip (57 matches)
    40 Dynamic ACCESS permit tcp any any eq www
    50 deny ip any any log
```

Try connecting to the rotary line (port 7001). As we remember, it enables the actual dynamic entry and does not insert the host IP address.

```
Rack1SW1#telnet 150.1.6.6 7001
Trying 150.1.6.6, 7001 ... Open


User Access Verification

Password: CISCO
[Connection to 150.1.6.6 closed by foreign host]
Rack1SW1#

Rack1R6#show ip access-list 100
Extended IP access list 100
    10 permit tcp any any eq telnet
    20 permit tcp any any eq 7001 (42 matches)
    30 permit udp any any eq rip (6 matches)
    40 Dynamic ACCESS permit tcp any any eq www
       permit tcp any any eq www
    50 deny ip any any log (1 match)
```

As you can see now anyone can connect across the router to the HTTP port.

Pitfall

Note that you can have only one dynamic entry per access-list. In addition, if you are using dynamic ACLs with AAA enabled, make sure you configure local AAA exec authorization when you bind auto-commands to user names (or use the "none" or "if-authenticated" methods if you bind the command to VTY lines).

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.9 Filtering Traffic with Time-Based Access Lists

- Restrict the users behind R2 from accessing BB2 on weekends.
- Additionally, disallow access to BB2 from 6pm to 9am on the remaining days.

Configuration

```
R2:
!
! Evenings and nights on weekdays.
!
time-range WEEKDAYS_EVES
 periodic weekdays 18:00 to 23:59
 periodic weekdays 0:00 to 8:59
!
! Weekends
!
time-range WEEKENDS
 periodic weekend 0:00 to 23:59
!
ip access-list extended OUTBOUND
 deny ip any any time-range WEEKDAYS_EVES
 deny ip any any time-range WEEKENDS
 permit ip any any
!
interface FastEthernet 0/0
 ip access-group OUTBOUND out
```

Verification

Note

Time-based ACLs allow the use of time ranges, bound to a specific entry. This allows the entry to be active only during the time-range specified. There are two types of time-ranges: absolute and periodic. Absolute time-ranges define non-repeating time intervals (e.g. from Jan 10 to Jan 12). Periodic time-ranges define recurring time-intervals that may repeat infinitely (e.g. every Wednesday from 6am to 11am). The latter type is based on the concept of weekdays: e.g. you may define periods based on days of week (Mon, Tue, Fri etc) and specific ranges like daily, weekends, weekdays (Mon-Fri). Absolute intervals define starting dates/times and ending dates/times.

Copyright © 2010 Internetwork Expert

It is important to note that time-ranges start with the first second of the first minute in the range and end with the last second of the last minute in the range. For example, the interval from 00:01 to 02:00 will last from 00:01:00 to 02:00:59 (hh:mm:ss). Therefore, if you want some interval to end at exactly 4:00pm use the value "15:59" (note that IOS uses the 24 hour time format). Note that the exact boundaries of time ranges may vary based on implementation.

In our scenario, we define two time ranges. The first one covers weekends and the second one covers the time interval from 6pm to 9am every weekday.

As usual, remember that local traffic is not subject to an outbound ACL check. Therefore, try connecting from R3. At first, the connection fails since one of the time-ranges is active:

```
Rack1R3#telnet 192.10.1.254
Trying 192.10.1.254 ...
% Destination unreachable; gateway or host down

Rack1R3#

Rack1R2#show clock
05:14:07.194 UTC Sun Sep 28 2008

Rack1R2#sh ip access-lists
Extended IP access list OUTBOUND
    10 deny ip any any time-range WEEKDAYS_EVES (inactive)
    20 deny ip any any time-range WEEKENDS (active) (2 matches)
    30 permit ip any any
Rack1R2#
```

Now change the time on R2 to Monday during "business hours" and try connecting again:

```
Rack1R3#telnet 192.10.1.254
Trying 192.10.1.254 ... Open
<output omitted>
```

The connection is now successful. Check the access-list counters again:

```
Rack1R2#show ip access-lists
Extended IP access list OUTBOUND
    10 deny ip any any time-range WEEKDAYS_EVES (inactive)
    20 deny ip any any time-range WEEKENDS (inactive) (2 matches)
    30 permit ip any any (12 matches)
```

Copyright © 2010 Internetwork Expert

11.10 Traffic Filtering with Policy Based Routing

- Configure R6 to drop ICMP packets of 100 bytes in size entering the VLAN 146 interface.
- Ensure R6 only drops packets leaving via the VLAN 67 interface.
- The router should not send ICMP unreachable notifications about dropped packets.

Configuration

```
R6:
ip access-list extended ICMP
 permit icmp any any
!
route-map DROP
 match ip address ICMP
 match interface FastEthernet 0/0.67
 match length 100 100
 set interface Null0
!
interface FastEthernet 0/0.146
 ip policy route-map DROP
!
interface Null0
 no ip unreachables
```

Verification

Note

Policy Based Routing allows the implementation of select packet filtering based on various criteria. Among the most important are:

1) Based on any access-list (standard/extended)
2) Based on packet size
3) Based on the packet ToS byte
4) Based on the output interface (allows sub-interface granularity)

The packet drop behavior with PBR is achieved by using **`set interface Null0`** command. Based on the configuration of the Null0 interface, the router may generate an ICMP unreachable message. You can disable this feature to conserve router resources by using the command **`no ip unreachables`** under the Null0 interface. However, the router only sends unreachables for process-switched packets.

Copyright © 2010 Internetwork Expert

For verification, send ICMP packets of different sizes from R4 to SW1. You may need to remove the access-list applied to the R6 VLAN 67 interface:

```
Rack1R4#ping 150.1.7.7 size 200

Type escape sequence to abort.
Sending 5, 200-byte ICMP Echos to 150.1.7.7, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1R4#ping 150.1.7.7 size 100

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.7.7, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)
Rack1R4#

Rack1R6#show route-map
route-map DROP, permit, sequence 10
  Match clauses:
    ip address (access-lists): ICMP
    interface FastEthernet0/0.67
    length 100 100
  Set clauses:
    interface Null0
  Policy routing matches: 5 packets, 590 bytes
```

Note that the byte count is 590 not 500. The counters actually account for Layer 2 overhead, which is 18 bytes for 802.1q tagged frames.

Now verify that R6 sends IP unreachables if you enable them back on the Null0 interface and disable CEF switching on the VLAN 146 interface.

```
R6:
interface Null 0
 ip unreachables
!
interface FastEthernet 0/0
 no ip route-cache

Rack1R4#ping 150.1.7.7 size 100

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.7.7, timeout is 2 seconds:
U.U.U
Success rate is 0 percent (0/5)
```

Copyright © 2010 Internetwork Expert

11.11 Preventing Packet Spoofing with uRPF

- R4 connects to your ISP at the border of your network.
- Ensure that R4 does not accept packets with IP addresses of the internal subnets on its connection to the ISP.
- There is another connection to the same ISP in the network, so account for possible asymmetric routing issues on R4.
- At the same time, R4 should only accept packets from legitimate subnets of 150.X.0.0/16 and 155.X.0.0/16 learned via IGP on its internal connections.
- Log all packets with spoofed sources.

Configuration

```
R4:
!
! Access-List to deny and log any violating packet
!
access-list 100 deny ip any any log
!
! Apply Loose uRPF to the upstream link
!
interface FastEthernet 0/0
ip verify unicast source reachable-via any 100
!
! Apply strict uRPF to all internal interfaces
!
interface FastEthernet 0/1
 ip verify unicast source reachable-via rx 100
!
interface Serial 0/0/0
 ip verify unicast source reachable-via rx  100
!
interface Serial 0/1/0
 ip verify unicast source reachable-via rx  100
```

Verification

Note

uRPF or Unicast Reverse Path Forwarding is the concept of verifying the routing path for the source IP address found in an IP packet. Usually routers only use destination IP addresses to look up the next hop for an IP packet. Reverse-Path Forwarding, however, is used with multicast routing. From a security perspective, uRPF is useful for checking IP address spoofing conditions.

Copyright © 2010 Internetwork Expert

Generally, packets arrive on the interfaces that are on the shortest path to the source of the packets. This is the natural result of the IGP routing protocol at work on the device. However, with IP spoofing attacks, a malicious user may inject packets with IP addresses not belonging to its segment or network area. From the router perspective, such packets may appear on the interfaces not on the shortest path to their source. You should note that packets may have spoofed sources and appear on the shortest-path interface in cases when you have just one connection to all destinations.

Based on its function, uRPF could be a very useful security feature to prevent spoofing attacks, especially on routers that have a diverse number of connections, for example a system at an Internet peering point. The feature has two general modes of operation:

1) Strict mode - when you enable uRPF on an interface with the **`ip verify unicast source reachable-via rx`** command (or the legacy format **`ip verify unicast reverse-path`**), the router applies the uRPF check to the source IP addresses of incoming packets. The source IP address must match an explicit IP route in the routing table, and the next hop for this entry should point out of the interface the packet was received from. Otherwise, the router will drop the packets.

2) Loose mode - you enable the loose mode with the command **`ip verify unicast source reachable-via any`**. When a router receives a packet on the interface with loose uRPF enabled, it just checks that it has an IP route matching the source address in the packet. It does not matter whether the next hop for this route points out the receiving interface or not. The exception to this checking is that if the route is to Null0, the packet is dropped. This feature is useful in enterprises that have more then one ISP uplink and use asymmetric routing. With asymmetric routing, packets may take paths not allowed with strict uRPF.

uRPF does have additional features. The first one is uRPF exemptions and violation logging. With this feature, you may specify a standard or extended access-list as follows: **`ip verify unicast source reachable-via {rx|any} <ACL-NUM>`**. The uRPF feature consults this access-list for packets *violating* the uRPF condition. If the ACL permits a packet, it is allowed to pass through. If the ACL denies the packet, the router drops it. You may use the **`log`** keyword to log the packets allowed or denied by the uRPF access-list.

Copyright © 2010 Internetwork Expert

The other two features are:

1) Allowing the router to do a self-ping with strict uRPF enabled on the interfaces (i.e. accepting packets sourced from the router on the interface they were sent from).

2) Allowing the use of a default route with strict uRPF. In this case, the router will match source IP addresses against the default-route entry (0.0.0.0/0) as well. This is useful when you want to accept all packets on your Internet connections, but feature protection from spoofed packets coming from your internal network.

To verify the feature, first make sure it has been applied to all interfaces:

```
Rack1R4#show ip interface fastEthernet 0/1 | inc verify
  IP verify source reachable-via RX, ACL 100

Rack1R4#show ip interface fastEthernet 0/0 | inc verify
  IP verify source reachable-via ANY, ACL 100

Rack1R4#show ip interface Serial 0/0/0 | inc verify
  IP verify source reachable-via RX, ACL 100

Rack1R4#show ip interface Serial 0/1/0 | inc verify
  IP verify source reachable-via RX, ACL 100
```

Now generate some spoofed traffic from R1 (the inside network) across R4. Create a special Loopback interface on R1 to accomplish this:

```
R1:
interface Loopback1
 ip address 150.2.1.1 255.255.255.0

R4:
ip access-list log-update threshold 1

Rack1R1#ping 150.1.4.4 source loopback 1 repeat 10

Type escape sequence to abort.
Sending 10, 100-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
Packet sent with a source address of 150.2.1.1
```

Copyright © 2010 Internetwork Expert

Observe the logging and statistics on R4:

```
Rack1R4#
%SEC-6-IPACCESSLOGDP: list 100 denied icmp 150.2.1.1 -> 150.1.4.4
(0/0), 1 packet
%SEC-6-IPACCESSLOGDP: list 100 denied icmp 150.2.1.1 -> 150.1.4.4
(0/0), 1 packet
%SEC-6-IPACCESSLOGDP: list 100 denied icmp 150.2.1.1 -> 150.1.4.4
(0/0), 1 packet

Rack1R4#show ip interface FastEthernet 0/1
<snip>
  IP verify source reachable-via RX, ACL 100
  10 verification drops
  0 suppressed verification drops

Rack1R4#show ip access-lists 100
Extended IP access list 100
    10 deny ip any any log (10 matches)
```

Copyright © 2010 Internetwork Expert

11.12 Using NBAR for Content-Based Filtering

- Configure R6 to drop any potentially dangerous HTTP downloads requested by users on and behind VLAN 146.
- The dangerous files have extensions ".exe", ".com", or ".bin".
- Use a single pattern to match all the filenames at once.

Configuration

```
R6:
class-map match-all EXTENSION
 match protocol http url "*.bin|*.exe|*.com"
!
!
policy-map DROP
 class EXTENSION
  drop
!
interface FastEthernet 0/0.146
  service-policy output DROP
```

Verification

Note

The NBAR (Network Based Application Recognition) protocol classification mechanism allows deep inspection for some protocol types. One notable feature is matching various fields inside HTTP headers. In recent IOS versions, this feature evolved into the full-blown Flexible Packet Matching (FPM) feature, which allows for comprehensive inspection of any IP packets.

Commonly you see HTTP inspection used for matching the URL field against certain patterns. The URL matching occurs for HTTP GET/PUT/POST requests. It is important to note that the NBAR engine classifies bi-directional traffic flows. That is, if you apply a policy map matching the URL pattern inbound on an interface, the policy map will classify both incoming HTTP requests as well as packets returning in server replies. This is because NBAR classification occurs independently of policy-map direction, and the classification decision applies to both parts of a bi-directional traffic flow.

The command to match a URL is:

match http protocol url <pattern>

Copyright © 2010 Internetwork Expert

The URL matching feature uses special wildcard symbols:

"*" – matches any sequence of characters (non-empty)
"?"– matches any single character (you need to press Ctrl-V in order to enter "?")
"|" – alternative, logical OR
"[]" – range, e.g. [ab] matches either "a" or "b"
"()" – grouping; delimit the logical end of a pattern; or example, you can use "*.(exe|bin)" as equivalent to "*.exe|*.bin"

All matching is case insensitive. The pattern "text" matches "TEXT" as well. The engine matches your URL pattern against the directory path and the file name in the URL. For example, if the URL string is "http://www.cisco.com/pub/uploads/image.jpeg", the matching procedure will only use the "pub/uploads/image.jpeg" part of the URL. When you submit a request like the above URL, it translates into the following headers (there are actually more, but this is the bare minimum):

```
GET /pub/uploads/image.jpeg HTTP/1.1
Host: www.cisco.com
```

Thus if you need to match a host name, you should use the **`match protocol http host`** statement. You can use the same wildcard characters to match the HTTP Host header.

We are going to use the following single-line expression for our task:

"*.bin|*.exe|*.com"

This pattern matches files with extension "*.com", "*.exe" or "*.bin". You can also use the "match-any" class map and match multiple URL strings on separate lines, resulting in the same effect as separating a pattern with the symbol "|".

The second interesting part of using this feature is the MQC **`drop`** operation. Under any MQC class, you can apply this operation to drop the matching packets.

Copyright © 2010 Internetwork Expert

For verification of our task, simulate an HTTP server with SW1 and create some files in the flash with extensions “exe”, “com”, and “bin”.

```
SW1:
ip http server
ip http path flash:
!
Rack1SW1#copy flash:vlan.dat flash:vlan.exe
Destination filename [vlan.exe]?
Copy in progress...C
1216 bytes copied in 1.082 secs (1124 bytes/sec)

Rack1SW1#copy flash:vlan.dat flash:vlan.com
Destination filename [vlan.com]?
Copy in progress...C
1216 bytes copied in 0.025 secs (48640 bytes/sec)

Rack1SW1#copy flash:vlan.dat flash:vlan.BIN
Destination filename [vlan.BIN]?
Copy in progress...C
1216 bytes copied in 0.025 secs (48640 bytes/sec)
```

Now try downloading those files on R1 across R6:

```
Rack1R1#copy http://admin:cisco@150.1.7.7/vlan.exe null:
%Error opening http://admin:cisco@150.1.7.7/vlan.exe (I/O error)

Rack1R6#show policy-map interface FastEthernet 0/0.146
 FastEthernet0/0.146

  Service-policy output: DROP

    Class-map: EXTENSION (match-all)
      9 packets, 1827 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol http url "*.bin|*.exe|*.com”
      drop

    Class-map: class-default (match-any)
      15 packets, 2210 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

Rack1R1#copy http://admin:cisco@150.1.7.7/vlan.BIN null:
%Error opening http://admin:cisco@150.1.7.7/vlan.BIN (I/O error)

Rack1R1#copy http://admin:cisco@150.1.7.7/vlan.com null:
%Error opening http://admin:cisco@150.1.7.7/vlan.com (I/O error)
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#show policy-map interface FastEthernet 0/0.146
 FastEthernet0/0.146

  Service-policy output: DROP

    Class-map: EXTENSION (match-all)
      52 packets, 9884 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol http url "*.bin|*.exe|*.com"
      drop

    Class-map: class-default (match-any)
      57 packets, 11676 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Now attempt to transfer a legitimate file and ensure that R6 allows it to pass through:

```
Rack1R1#copy http://admin:cisco@150.1.7.7/vlan.dat null:
Loading http://***********@150.1.7.7/vlan.dat !
1216 bytes copied in 0.037 secs (32865 bytes/sec)
```

Copyright © 2010 Internetwork Expert

11.13 TCP Intercept

- Configure R6 to protect the WWW servers on VLAN 146 from a SYN flood attack.
- Intercept connections as they go to the servers and close connections after 30 seconds of inactivity.
- Allow for a maximum of 100 semi-established connections and drop their number down to 80 when the threshold is crossed.
- The router should start dropping half-open connections when they cross the rate threshold of one connection per second. The router should not stop until the average rate is one per two minutes.
- The router should not account for connection age when dropping them.

Configuration

```
R6:
access-list 146 permit ip any 155.1.146.0 0.0.0.255
ip tcp intercept list 146
ip tcp intercept connection-timeout 30
ip tcp intercept max-incomplete low 80 high 100
ip tcp intercept one-minute low 30 high 60
ip tcp intercept drop-mode random
```

Verification

Note

The TCP Intercept feature protects against TCP SYN flood attacks targeted at servers located behind the router. The idea of SYN-flooding is to send a huge number of SYN packets towards the server, without ever finishing the 3-way TCP handshake. The server then exhausts its resources and may eventually refuse to accept a connection from a legitimate user.

There are two modes of the TCP Intercept feature. In the first mode, the router acts as a proxy between clients and the server. For every incoming SYN packet, the router immediately answers with a SYN-ACK packet to the client and sends a SYN packet to the server. The router then waits for a final ACK from the client and a SYN-ACK from the server. When both “parts” of the connection establish, the router joins them and routes packets transparently.

Copyright © 2010 Internetwork Expert

If one of the endpoints does not respond, the router probes it repeatedly using an exponential backoff algorithm. The router sends the second SYN packet in 1 second; 3rd SYN in 2 seconds, 4th SYN in 4 seconds, 5th retransmit in 8 seconds and 6th retransmit in 16 seconds. Overall, this takes 1+2+4+8+16=31 seconds.

You can enable the TCP intercept mode using the command:

```
ip tcp intercept mode intercept
```

In this mode the router limits the total number of half-open (3-way handshake unfinished) connections from clients. The limit is set using the command:

```
ip tcp intercept max-incomplete {low|high} <Thresh>
```

This limit only applies to the half-open connections and does not affect the established connections. If there is a new connection forming and the number reaches the high threshold, the router starts dropping the half-open connections until their number falls down to the low threshold.

You may also limit the rate of the incoming SYN-packets using a per-minute value. Specifically, you can set the maximum number of half-open sessions allowed per-minute using the command:

```
ip tcp intercept one-minute {low|high} <Thresh>
```

When the number of semi-established sessions opened during the last minute exceeds the high threshold, the router starts dropping them until the measured rate drops below the low threshold. The router keeps track of both thresholds at the same time, and drops the connections until their number reaches the low threshold.

Depending on the drop-mode, the router may drop either the oldest half-open session or any random session (until it reaches the low threshold).

```
ip tcp intercept drop-mode {random|oldest}
```

In addition to deleting the excessive connection states, the router also changes the backoff algorithm behavior. It reduces all retransmit timers in half, so the additional probes are sent in 0.5 seconds, 1 second, 2 seconds, 4 seconds, and 8 seconds. This totals to 15 seconds of probing for a non-answering endpoint. Altogether, dropping the connections and reducing the backoff timers is called “Aggressive” mode behavior.

Copyright © 2010 Internetwork Expert

The other settings related to the TCP Intercept mode are the following:

1) FIN/RST timeout - this is the amount of time the router keeps a connection entry after it receives a FIN packet from a client or server. After this amount of time, the router sends forceful RST packets to the server (making it free the resources) and removes the entry. The command to set this timeout is:

ip tcp intercept finrst-timeout <Timeout>

2) Limit the scope of TCP interception:

ip tcp intercept list <ACL-NUM>|<ACL-NAME>

Only the connections matching the access-list are subject to TCP interception. By default, TCP Intercept does not apply to any transit connection, so make sure you specified a list of packets to match.

3) The timeout for inactive connections - defines the amount of time that TCP Intercept maintains an inactive connection (no data in it):

ip tcp intercept connection-timeout <TIMEOUT>

After the timeout expires, the router sends RST packets to both sides of the connection, effectively terminating it.

For verification in our task, try connecting to R4 (a legitimate connection) across R6 and keep the connection inactive for 30 seconds:

```
Rack1R6#debug ip tcp intercept
TCP intercept debugging is on

Rack1SW1#telnet 155.1.146.4
Trying 155.1.146.4 ... Open


User Access Verification

Password: cisco
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#
05:27:09: INTERCEPT: new connection (155.1.67.7:11008 SYN ->
155.1.146.4:23)
05:27:09: INTERCEPT(*): (155.1.67.7:11008 <- ACK+SYN 155.1.146.4:23)
05:27:09: INTERCEPT: 1st half of connection is established
(155.1.67.7:11008 ACK -> 155.1.146.4:23)
05:27:09: INTERCEPT(*): (155.1.67.7:11008 SYN -> 155.1.146.4:23)
05:27:09: INTERCEPT: 2nd half of connection established
(155.1.67.7:11008 <- ACK+SYN 155.1.146.4:23)
```

First, the router adds the connection to the intercept list and ensures that both parts of the connections (client <-> router and router <-> server) have been established. After that, the router counts the time of inactivity.

```
05:27:09: INTERCEPT(*): (155.1.67.7:11008 ACK -> 155.1.146.4:23)
05:27:09: INTERCEPT(*): (155.1.67.7:11008 <- WINDOW 155.1.146.4:23)
05:27:41: INTERCEPT: ESTAB timing out (155.1.67.7:11008 <->
155.1.146.4:23)
```

After 30 seconds, the router terminates the connection by sending RST packets to both the client and the server.

```
05:27:41: INTERCEPT(*): (155.1.67.7:11008 <- RST 155.1.146.4:23)
05:27:41: INTERCEPT(*): (155.1.67.7:11008 RST -> 155.1.146.4:23)
```

Now we block ACK packets going from the client to the server, effectively preventing the established connection formation.

```
R6:
ip access-list extended NO_ACK
 deny tcp any any established
 permit ip any any
!
interface FastEthernet 0/0.67
 ip access-group NO_ACK in
```

Open a connection to R4 from SW1. It seems to be open, because R4 responds with a SYN+ACK.

```
Rack1SW1#telnet 155.1.146.4
Trying 155.1.146.4 ... Open

05:36:02: INTERCEPT: new connection (155.1.67.7:11009 SYN ->
155.1.146.4:23)
```

However, the last ACK from SW1 is blocked by R6. Therefore, R6 tries to re-send the SYN+ACK to SW1 five times, and gives up after 31 seconds. After that, it sends the RST and terminates the connection.

Copyright © 2010 Internetwork Expert

```
05:36:02: INTERCEPT(*): (155.1.67.7:11009 <- ACK+SYN 155.1.146.4:23)
05:36:03: INTERCEPT(*): SYNRCVD retransmit 1 (155.1.67.7:11009 <-
ACK+SYN 155.1.146.4:23)
05:36:05: INTERCEPT(*): SYNRCVD retransmit 2 (155.1.67.7:11009 <-
ACK+SYN 155.1.146.4:23)
05:36:09: INTERCEPT(*): SYNRCVD retransmit 3 (155.1.67.7:11009 <-
ACK+SYN 155.1.146.4:23)
05:36:17: INTERCEPT(*): SYNRCVD retransmit 4 (155.1.67.7:11009 <-
ACK+SYN 155.1.146.4:23)
05:36:33: INTERCEPT: SYNRCVD retransmitting too long (155.1.67.7:11009
<-> 155.1.146.4:23)
05:36:33: INTERCEPT(*): (155.1.67.7:11009 <- RST 155.1.146.4:23)
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.14 TCP Intercept Watch Mode

- Modify your TCP intercept configuration in order to reset any connection that does not reach the established state in 20 seconds.

Configuration

```
R6:
ip tcp intercept mode watch
ip tcp intercept watch-timeout 20
```

Verification

Note

TCP Intercept Watch Mode is a "lightweight" version of the full TCP Intercept Mode. Instead of sitting in the path of TCP connections, the router just watches them as they go. The major timeout value for the watch-mode is the "watch timeout". If the client and server did not negotiate a connection during this time, the router forcefully injects RST messages to the server, causing it to free up the allocated resources. Note that this is in contrast to the intercept mode, where the router attempts communicating with the client and the server by itself, using an exponential backoff. The command is:

ip tcp intercept watch-timeout <TIMEOUT>.

This command only makes sense when you configure the Watch mode using the command:

ip tcp intercept mode watch

Watch mode utilizes the same timers as the Intercept mode. The semi-established connection thresholds are treated in the same way, switching to Aggressive mode when their number reaches the high threshold. When in Aggressive mode, the configured watch timeout is reduced by half as well. There are no retransmission attempts, as the router is not acting as a proxy.

Copyright © 2010 Internetwork Expert

To verify this task, configure R6 to block the ACK packets from SW1:

```
R6:
ip access-list extended NO_ACK
 deny tcp any any established
 permit ip any any
!
interface FastEthernet 0/0.67
 ip access-group NO_ACK in
```

Now enable TCP Intercept debugs and try connecting from SW1 to R4:

```
Rack1R6#debug ip tcp intercept
TCP intercept debugging is on

Rack1SW1#telnet 155.1.146.4
Trying 155.1.146.4 ... Open
```

R6 does not intervene with the connection, but passively watches it. As the watch timeout passes (20 seconds), R6 sends a RST to the server and terminates the connection, freeing resources on the server:

```
Rack1R6#
06:23:48: INTERCEPT: new connection (155.1.67.7:11010 SYN ->
155.1.146.4:23)
06:23:48: INTERCEPT: (155.1.67.7:11010 <- ACK+SYN 155.1.146.4:23)
06:23:50: INTERCEPT: server packet passed in SYNRCVD (155.1.67.7:11010
<- 155.1.146.4:23)
06:23:54: INTERCEPT: server packet passed in SYNRCVD (155.1.67.7:11010
<- 155.1.146.4:23)
06:24:02: INTERCEPT: server packet passed in SYNRCVD (155.1.67.7:11010
<- 155.1.146.4:23)
06:24:08: INTERCEPT: SYNRCVD timing out (155.1.67.7:11010 <->
155.1.146.4:23)
06:24:08: INTERCEPT(*): (155.1.67.7:11010 RST -> 155.1.146.4:23)
```

Copyright © 2010 Internetwork Expert

Open a connection to R4 from SW1. It seems to be open, because R4 responds with a SYN+ACK. However, the last ACK from SW1 is blocked by R6. Therefore, R6 tries to re-send the SYN+ACK to SW1 five times, and gives up after 31 seconds. After that, it sends a RST and terminates the connection.

```
05:36:02: INTERCEPT(*): (155.1.67.7:11009 <- ACK+SYN 155.1.146.4:23)
05:36:03: INTERCEPT(*): SYNRCVD retransmit 1 (155.1.67.7:11009 <-
ACK+SYN 155.1.146.4:23)
05:36:05: INTERCEPT(*): SYNRCVD retransmit 2 (155.1.67.7:11009 <-
ACK+SYN 155.1.146.4:23)
05:36:09: INTERCEPT(*): SYNRCVD retransmit 3 (155.1.67.7:11009 <-
ACK+SYN 155.1.146.4:23)
05:36:17: INTERCEPT(*): SYNRCVD retransmit 4 (155.1.67.7:11009 <-
ACK+SYN 155.1.146.4:23)
05:36:33: INTERCEPT: SYNRCVD retransmitting too long (155.1.67.7:11009
<-> 155.1.146.4:23)
05:36:33: INTERCEPT(*): (155.1.67.7:11009 <- RST 155.1.146.4:23)
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.15 Packet Logging with Access-Lists

- Configure R6 to log all ICMP packets entering its VLAN 146 interface.
- Aggregate logging messages so that a log entry is generated after five access-list entry hits.
- Reduce the burden on the router CPU by process switching only one packet per second.
- Ensure the logged messages allow you to track the MAC address of the host sending the packet.

Configuration

```
R6:
ip access-list log-update threshold 5
ip access-list logging interval 1000
!
ip access-list extended LOGGING
 permit icmp any any log-input
 permit ip any any
!
interface FastEthernet0/0.146
 ip access-group LOGGING in
```

Verification

Note

You can configure an access-list entry with the **log** keyword, making it log the matching packets via syslog. Using the **log-input** keyword, you can also log the input interface and the source MAC address. There are two configuration "knobs" to be used with access-list based logging. The first one is the logging update-threshold. It specifies how many times a packet must hit the access-list entry to generate a logging message. By default, this value is zero, which means the router would generate an entry based on a periodic timeout of 5 minutes. This command serves the purpose of aggregating the hits on the entry and the format is:

ip access-list log-update threshold <HIT-COUNT>

When this value is non-zero, the router generates a log entry for every number of hits and also produces the periodic 5 minute logging message.

Copyright © 2010 Internetwork Expert

The second knob relates to the behavior of packet logging. You should remember that every logged packet is process-switched, thus a large packet flow may easily eat all your CPU resources. For this reason, you may want to rate-limit the amount of process-switched packets using the command:

ip access-list logging interval <TIME-INTERVAL>

This command allows only one packet per the interval to be process-switched. All exceeding packets are not process-switched and thus are not accounted for logging purposes. By default, this feature is off and all packets are process-switched. Note that this interval does not apply to packets destined to the router itself, since these are process-switched by default. Using this command limits the effect of packet logging on the CPU, but may result in an unpredictable number of packets being logged. In reality it is useful when you just need a general hint of packet matches, not the detailed statistics.

You may also limit the amount of all syslog messages (including the ACL logging messages) using the generic syslog rate-limiting feature.

To verify, try sending a barrage of ping packets from R4 to SW1 across R6:

```
Rack1R4#ping 150.1.7.7 repeat 1000

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 150.1.7.7, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
<snip>
Success rate is 100 percent (1000/1000), round-trip min/avg/max =
1/2/12 ms
Rack1R4#

Rack1R6#
07:19:15: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 1 packet
```

Now change the logging interval and set the log-update threshold to 1 packet:

```
R6:
ip access-list logging interval 1
ip access-list log-update threshold 1

Rack1R4#ping 150.1.7.7 repeat 1000 size 1500

Type escape sequence to abort.
Sending 1000, 1500-byte ICMP Echos to 150.1.7.7, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!<
<snip>
Success rate is 100 percent (1000/1000), round-trip min/avg/max =
4/5/12 ms
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#
07:31:35: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 1 packet
07:31:38: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 1 packet
07:31:38: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 1 packet
07:31:39: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 1 packet
07:31:39: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 1 packet
07:31:40: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 1 packet
```

As you can see, in both cases the source MAC address and the input interface were logged. To get the true count of aggregate packet matches, disable the logging interval:

```
Rack1R4#ping 150.1.7.7 repeat 50 size 1500

Type escape sequence to abort.
Sending 50, 1500-byte ICMP Echos to 150.1.7.7, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (50/50), round-trip min/avg/max = 4/7/12 ms
Rack1R4#

Rack1R6#show logging
<snip>

07:37:27: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 5 packets
07:37:27: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 5 packets
07:37:27: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 5 packets
07:37:27: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 5 packets
07:37:27: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 5 packets
07:37:27: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 5 packets
07:37:27: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 5 packets
07:37:27: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 5 packets
07:37:27: %SEC-6-IPACCESSLOGDP: list LOGGING permitted icmp 155.1.146.4
(FastEthernet0/0.146 0011.21c8.ec41) -> 150.1.7.7 (0/0), 5 packets
```

Now you see 10 logging entries with 5 packet counts each.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.16 Stateful Filtering with CBAC

- Configure R4 as a stateful firewall. R4's VLAN 146 interface is the protected network and the Serial interface between R4 and R5 is the connection to outside networks.
- Ensure that TFTP and FTP protocols function across the firewall.
- Collect connection statistics for HTTP transfers.
- Allow pinging across the firewall but do not add any special ACL entries to accomplish this.
- Allow UDP sessions across the firewall with a 30 second inactivity timeout.
- Ensure that inspection entries for DNS requests expire in 10 seconds but do not inspect the DNS protocol for this.
- Disable CBAC alerts for all protocols with the exception of FTP.
- Do not allow any other TCP-based protocols except for those mentioned above.
- In the future some users are going to use the Cisco ezVPN client from inside the firewall. Ensure you do not need to add any ACL entries in order to allow ezVPN connections across the firewall.

Configuration

```
R4:
!
! Disable alerts globally and configure DNS timeout
!
ip inspect alert-off
ip inspect dns-timeout 10
!
! Define inspection rule and protocol-specific settings
! Enable alerts for FTP only
!
ip inspect name INSPECT ftp alert on
!
! Enable audit trails for HTTP only
!
ip inspect name INSPECT http audit-trail on
!
! Inspect all UDP sessions and expire them after 30 seconds
! of inactivity
!
ip inspect name INSPECT udp timeout 30
ip inspect name INSPECT tftp
!
! Inspecting ISAKMP dynamically opens a hole for IPsec Phase 2 traffic
!
ip inspect name INSPECT isakmp
ip inspect name INSPECT icmp
!
```

Copyright © 2010 Internetwork Expert

```
ip access-list extended INBOUND
 permit udp any any eq 520
 deny ip any any
!
interface Serial 0/1/0
 ip inspect INSPECT out
 ip access-group INBOUND in
```

Verification

Note

CBAC or context-based access-control introduces true stateful inspection to IOS code. You can compare it to the reflexive ACL feature as being an advanced inspection technology. The general idea is to inspect protocol-specific information in traffic flows going across the router and dynamically open holes in access-lists for returning traffic. CBAC configurations commonly consist of two parts:

1) The CBAC inspection rule (defines traffic classes to inspect) applied to an interface in the direction matching the initial flow of traffic (usually from the protected network to the outside world).

2) An access-group applied to an interface in the direction matching the returning traffic flow (usually from the outside network to the protected network).

Commonly, the initial flow of the traffic is from the protected network (inside of the firewall) to the unprotected network (outside of the firewall). For example, think of an office network initiating HTTP connections to the Internet across the filtering router.

You define an inspection rule using the command:

```
ip inspect name <RULE-NAME> <protocol> …
ip inspect name <RULE-NAME> <protocol> …
```

and apply it to an interface using the interface-level command

```
ip inspect <RULE-NAME> {in|out}
```

Copyright © 2010 Internetwork Expert

The rule inspects traffic flowing in the direction specified and installs a special ACL bypass entry in the packet switching path to allow returning packets to pass through the router, even if there is an ACL denying them (in older IOS releases CBAC was simply adding new ACL entries dynamically, but now it's a bypass rule which performs more efficiently). Therefore, even if there is a local ACL in the path of returning packets, it will be bypassed by packets identified as part of the returning protocol traffic flow. For example, with the FTP protocol, CBAC dynamically permits the active data transfer connection (which is in the reverse direction), even without knowing the communication port in advance.

You may put the inspection rule and the access-list on different interfaces. As long as they apply to the initial connections and returning traffic flows respectively it's fine. For example, you can apply the inspection rule as outgoing on the outside interface and an access-list in the incoming direction. You can also put the inspection rule in the incoming direction on the inside interface and the access-list in incoming direction on the outside interface.

CBAC protocol inspection can provide you with alarms, notifying you of protocol violations and potential attacks. By default alerts are enabled for all inspected protocols. You can disable alarms globally and configure them selectively per-protocol.

In addition to providing you with alarm messages, CBAC can perform basic traffic accounting (duration, bytes transferred) by logging audit trails in syslog. This feature is disabled by default, and could be enabled globally for all inspected protocols or tuned selectively per protocol in an inspection rule.

CBAC may inspect generic TCP/UDP connections (without looking for specific upper level protocol information) just to check their integrity and open a hole for returning traffic. If you enable TCP inspection along with HTTP inspection at the same time, HTTP inspection would be used for connections made on port 80.

By default, CBAC (as well as outgoing ACLs) do not apply to router-generated traffic. You need to add manual ACL entries in order to permit returning traffic for router-generated packets. Alternatively, you can use local policy routing or a special CBAC feature for router-generated traffic inspection. We are going to discuss this feature in a separate task.

CBAC has a number of configurable inspection timeouts. In this task we are going to discuss just the application inactivity timeout. For every inspected protocol you can set up an inactivity timeout. CBAC will close the inspection entry after the timeout expires.

Copyright © 2010 Internetwork Expert

There is a special command **ip inspect dns-timeout <TIMEOUT>** which applies when you inspect generic UDP sessions. Since DNS represents a special case of the UDP protocol, it has a separate timeout. DNS "sessions" are quick request/response, so by default when CBAC sees UDP packet on port 53 it holds the inspection entry for 5 seconds, not the default interval specified for the UDP protocol. In the later releases of IOS, you can define separate DNS inspection rules with their own timeouts.

Check the general information on the configured inspection rules. You can use show commands to see the timeout values (also you can use them to learn other default values).

```
Rack1R4#show ip inspect config
Session audit trail is disabled
Session alert is disabled
one-minute (sampling period) thresholds are [400:500] connections
max-incomplete sessions thresholds are [400:500]
max-incomplete tcp connections per host is 50. Block-time 0 minute.
tcp synwait-time is 30 sec -- tcp finwait-time is 5 sec
tcp idle-time is 3600 sec -- udp idle-time is 30 sec
dns-timeout is 10 sec
Inspection Rule Configuration
 Inspection name INSPECT
    ftp alert is on audit-trail is off timeout 3600
    http alert is off audit-trail is on timeout 3600
    dns alert is off audit-trail is off timeout 30
    tftp alert is off audit-trail is off timeout 30
    isakmp alert is off audit-trail is off timeout 30
    icmp alert is off audit-trail is off timeout 10
    udp alert is off audit-trail is off timeout 30

Rack1R4#show ip inspect interfaces
Interface Configuration
 Interface Serial0/1/0
  Inbound inspection rule is not set
  Outgoing inspection rule is INSPECT
    ftp alert is on audit-trail is off timeout 3600
    http alert is off audit-trail is on timeout 3600
    dns alert is off audit-trail is off timeout 30
    tftp alert is off audit-trail is off timeout 30
    isakmp alert is off audit-trail is off timeout 30
    icmp alert is off audit-trail is off timeout 10
    udp alert is off audit-trail is off timeout 30
  Inbound access list is INBOUND
  Outgoing access list is not set
```

Copyright © 2010 Internetwork Expert

To verify the functionality of CBAC, you can use the following commands

```
Rack1R1#telnet 155.1.45.5 80
Trying 155.1.45.5, 80 ... Open

Rack1R4#show ip inspect sessions
Established Sessions
 Session 852BD3D4 (155.1.146.1:42652)=>(155.1.45.5:80) http SIS_OPEN
Rack1R4#

[Resuming connection 1 to R1 ... ]

Rack1R1#
GET / HTTP/1.1

HTTP/1.1 400 Bad Request
Date: Thu, 02 Oct 2008 12:14:39 GMT
Server: cisco-IOS
Connection: close
Accept-Ranges: none

400 Bad Request

[Connection to 155.1.45.5 closed by foreign host]

Rack1R4#
CBAC: Finding pregen session for src_tableid:0, src_addr:155.1.146.1,
src_port:42652, dst_tableid:0, dst_addr:155.1.45.5, dst_port:80
%FW-6-SESS_AUDIT_TRAIL_START: Start http session: initiator
(155.1.146.1:42652) -- responder (155.1.45.5:80)
%FW-6-SESS_AUDIT_TRAIL: Stop http session: initiator
(155.1.146.1:42652) sent 18 bytes -- responder (155.1.45.5:80) sent 141
bytes
```

The above output demonstrates an open HTTP session inside the CBAC tables, as well as audit trails generated for our simulated HTTP connection.

Copyright © 2010 Internetwork Expert

Now you can verify that TFTP is working across the stateful firewall. TFTP is a protocol that uses dynamically generated ports for data transfer, so it is a perfect candidate for CBAC inspection testing.

Configure R5 as a TFTP server and try transferring a file across the firewall:

```
R5:
tftp-server flash:c2600-adventerprisek9-mz.124-10.bin alias test

Rack1R1#copy tftp://155.1.45.5/test null:
Accessing tftp://155.1.45.5/test...
Loading test from 155.1.45.5 (via FastEthernet0/0): !

Rack1R4#show ip inspect sessions
Established Sessions
 Session 852BCC54 (155.1.45.5:49511)=>(155.1.146.1:49822) tftp-data SIS_OPEN
Half-open Sessions
 Session 852BD154 (155.1.146.1:51100)=>(155.1.45.5:69) tftp SIS_OPENING
 Session 852BD654 (155.1.146.1:49564)=>(155.1.45.5:69) tftp SIS_OPENING
 Session 852BCED4 (155.1.146.1:49822)=>(155.1.45.5:69) tftp SIS_OPENING
 Session 852BD3D4 (155.1.146.1:57759)=>(155.1.45.5:69) tftp SIS_OPENING
```

Note that the initial connection appears as a UDP half-open session. This is because TFTP never replies directly to the original source port from port 69, but rather starts additional data connections. However, overall TFTP works across the firewall, even though it uses dynamic ports.

Finally, test that ping operations work across the firewall by the virtue of ICMP inspection:

```
Rack1R1#ping 155.1.45.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.45.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/29/32 ms
```

Copyright © 2010 Internetwork Expert

11.17 Advanced CBAC Features

- Configure R4 so that you do not have to create manual openings for router-generated UDP traffic in the inbound access-lists.
- You expect no more than 5000 concurrent sessions in the busiest hours. Optimize CBAC performance to handle this configured maximum number of sessions.
- Only allow Java applets downloaded from the WWW servers in the subnet 150.X.5.0/24.
- Someone runs an FTP server listening on port 80. The IP address of the server is 150.X.5.25. Ensure that CBAC correctly inspects FTP sessions to this host.
- Ensure that Java filtering applies to connections through an outside HTTP proxy using port 8080.

Configuration

```
R4:
!
! Enable inspection of router-generated UDP traffic (includes RIP)
!
ip inspect name INSPECT udp router-traffic
!
! Increase the hash-table size twice
!
ip inspect hashtable-size 2048
!
! A list of servers allowed for java applet download
!
access-list 99 permit 150.1.5.0 0.0.0.255
!
! Block java applets from all servers with except to the list
!
ip inspect name INSPECT http java-list 99
!
! The server that runs FTP on port 80
!
access-list 98 permit 150.1.5.25
!
! Use port 80 for FTP application on the above server
!
ip port-map ftp port 80 list 98
!
! Inspect HTTP protocol on port 8080 as well
!
ip port-map http port 8080
```

Copyright © 2010 Internetwork Expert

Verification

Note

CBAC uses an internal hash table to store connection information. Every active connection has an entry there. Entries are mapped into hash buckets using a special hash function. The smaller the hash table size (in buckets), the more connection entries map (collide) in the same bucket. On a very busy firewall this may lead to performance degradations, since the lookup through a bucket uses a linear search.

The default hash table size is 1024 buckets. It may vary in size from 2048, 4096 or 8192 buckets. When the number of concurrent connections across the firewall exceeds 4000, you may want to adjust the default size to match approximately half of the maximum connections count. In our case with 5000 sessions, the optimal hash table size would be 2048 buckets.

Java blocking is a special feature that allows the CBAC HTTP inspection engine to block Java applet downloads from certain sites. You specify a list of allowed sites using a standard ACL. This ACL lists servers that users may use to download Java applets.

CBAC uses a global port-mapping table that maps applications to their respective TCP/UDP port numbers. The IOS features a default system mapping for every application that CBAC may inspect. You cannot delete a system defined port mapping, but you can extend it by adding new ports. You may also use an access-list to limit the scope of new port definitions. Port overloading with an access-list applies to sessions destined to servers matching the standard access-list. While you cannot assign a system-defined port to any other application (for example, you cannot assign port 80 to FTP application) you can overload a system-defined port to another application using an access-list. In this scenario, we assign port 80 to an FTP application bound to a particular IP address.

These changes can be verified as follows:

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ip inspect all
Session audit trail is disabled
Session alert is disabled
one-minute (sampling period) thresholds are [400:500] connections
max-incomplete sessions thresholds are [400:500]
max-incomplete tcp connections per host is 50. Block-time 0 minute.
tcp synwait-time is 30 sec -- tcp finwait-time is 5 sec
tcp idle-time is 3600 sec -- udp idle-time is 30 sec
dns-timeout is 10 sec
Inspection Rule Configuration
 Inspection name INSPECT
    ftp alert is on audit-trail is off timeout 3600
    http java-list 99 alert is off audit-trail is on timeout 3600
    dns alert is off audit-trail is off timeout 30
    tftp alert is off audit-trail is off timeout 30
    isakmp alert is off audit-trail is off timeout 30
    icmp alert is off audit-trail is off timeout 10
    udp alert is off audit-trail is off timeout 30
 inspection of router local traffic is enabled
 Inspection name TEST
    http alert is off audit-trail is off timeout 30

Interface Configuration
 Interface Serial0/1
  Inbound inspection rule is not set
  Outgoing inspection rule is INSPECT
    ftp alert is on audit-trail is off timeout 3600
    http java-list 99 alert is off audit-trail is on timeout 3600
    dns alert is off audit-trail is off timeout 30
    tftp alert is off audit-trail is off timeout 30
    isakmp alert is off audit-trail is off timeout 30
    icmp alert is off audit-trail is off timeout 10
    udp alert is off audit-trail is off timeout 30
 inspection of router local traffic is enabled
  Inbound access list is INBOUND
  Outgoing access list is not set
```

Copyright © 2010 Internetwork Expert

To verify the user-defined port-mappings use the following commands:

```
Rack1R4#show ip port-map | inc http
Default mapping:  http                  tcp port 80
system defined
Default mapping:  http                  tcp port 8080
user defined
Default mapping:  https                 tcp port 443
system defined

Rack1R4#show ip port-map | inc ftp
Default mapping:  ftps                  tcp port 990
system defined
Default mapping:  tftp                  udp port 69
system defined
Default mapping:  ftp                   tcp port 21
system defined
Host specific:    ftp                   tcp port 80              in list
98  user defined
```

Copyright © 2010 Internetwork Expert

11.18 CBAC TCP/UDP Intercept Feature

- Configure R6 to protect the servers on VLAN 146 subnet from TCP/UDP based DoS attacks per the requirements below.
- Allow for a maximum of 100 half-open connections and configure the firewall to keep dropping the sessions until their number reaches 80.
- Limit the new connections rate to 30 per minute and clamp it down to 15 connections per second once the threshold has been crossed.
- Limit the maximum number of half-open TCP connections per server to 10. Block any new connections for 60 seconds when the number exceeds this limit.

Configuration

```
R6:
ip inspect max-incomplete low 80
ip inspect max-incomplete high 100
ip inspect one-minute low 15
ip inspect one-minute high 30
ip inspect tcp max-incomplete host 10 block-time 1
ip inspect tcp synwait-time 10
!
ip inspect name DEFEND tcp
ip inspect name DEFEND udp
!
interface FastEthernet 0/0.146
 ip inspect DEFEND out
```

Verification

Note

By default, with every inspection rule, CBAC enables DoS prevention features very similar to the TCP Intercept feature. However, with CBAC you cannot disable this behavior (only in some recent IOS releases, since 12.4(10) and later 12.4T trains). Another difference is that CBAC applies DoS prevention to UDP sessions as well. A UDP session is considered half-open if no packets have been sent in response to the initial packet. Both UDP and TCP half-open sessions count against maximum connection thresholds at the same time. Therefore, if you set up a limit of 100 half-open sessions the router will count both numbers of UDP and TCP sessions against this threshold.

Copyright © 2010 Internetwork Expert

CBAC supports maximum thresholds and per-minute rates for half-open sessions. When the current number of connections exceeds any of the thresholds, the code starts deleting half-open connection entries and sending RST packets to TCP session endpoints. The code does not delete any sessions that have already been established. For TCP sessions, the only supported mode of operations is "watch" mode. The code does not intervene with TCP sessions, just watches them as they get established. Offending sessions (sessions that did not complete in time or exceed thresholds) receive RST messages as well when they exceed configured timeouts. The amount of time CBAC waits for TCP connection establishment is configured using the command:

```
ip inspect tcp synwait-time <N>
```

Note this is the equivalent of "watch-timeout" for the TCP Intercept feature.

One feature special to CBAC is per-host TCP connection limits. CBAC bounds the number of half-open connections per-host, in addition to having the global thresholds. As the number of half-open sessions destined to the same IP address exceeds the per-host threshold, the router can block any further connections from establishing for the amount of time specified. The command syntax is:

```
ip inspect tcp max-incomplete host <N> block-time <T>
```

Lastly, remember that the CBAC interception feature applies whether you configure any inspection rule. In our case, the inspection rule simply inspects any TCP or any UDP traffic.

Copyright © 2010 Internetwork Expert

To verify how CBAC works, configure R6 to drop half-open sessions if they do not establish in 1 second.

```
R6:
ip inspect tcp synwait-time 1

Rack1R4#show ip inspect all
Rack1SW1#debug ip tcp transactions
TCP special event debugging is on

Rack1SW1#telnet 155.1.146.44
Trying 155.1.146.44 ...
TCB03A09440 created
TCB03A09440 setting property TCP_VRFTABLEID (14) 3A914FE
TCB03A09440 setting property TCP_TOS (1) 3A91458
TCB03A09440 bound to 0.0.0.0.11015
TCP: sending SYN, seq 3329226702, ack 0
TCP0: Connection to 155.1.146.44:23, advertising MSS 536
TCP0: state was CLOSED -> SYNSENT [11015 -> 155.1.146.44(23)]
TCP0: bad seg from 155.1.146.44 -- Rst bit set.: port 11015 seq 0 ack 0
rcvnxt 0 rcvwnd 4128 len 0
TCP0: timeout #1 - timeout is 4000 ms, seq 3329226702
TCP0: bad seg from 155.1.146.44 -- Rst bit set.: port 11015 seq 0 ack 0
rcvnxt 0 rcvwnd 4128 len 0
TCP0: timeout #2 - timeout is 8000 ms, seq 3329226702
TCP0: bad seg from 155.1.146.44 -- Rst bit set.: port 11015 seq 0 ack 0
rcvnxt 0 rcvwnd 4128 len 0
TCP0: timeout #3 - timeout is 15991 ms, seq 3329226702
TCP0: bad seg from 155.1.146.44 -- Rst bit set.: port 11015 seq 0 ack 0
rcvnxt 0 rcvwnd 4128 len 0
% Connection timed out; remote host not responding
```

Note the RST packets sent from the IP address “155.1.146.44”. Those are packets actually generated by R6 to terminate connections initiated from SW1.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.19 VLAN Filtering for IP Traffic

- Restrict traffic on VLAN 146 so that only R1, R4, and R6 are allowed to communicate.
- Allow RIP routing updates as well as transit TCP connections on VLAN 146.
- Disallow any other traffic to cross VLAN 146.

Configuration

```
SW1:
!
! The following filter denies any IP traffic
! that does not match the access-list in the first entry
! All Layer 2 traffic is permitted by default
!
ip access-list extended ALLOWED_L3_TRAFFIC
 permit udp any any eq 520
 permit tcp any any
!
vlan access-map VLAN146_FILTER 10
 match ip address ALLOWED_L3_TRAFFIC
 action forward
!
! Apply the filter to the VLAN
!
vlan filter VLAN146_FILTER vlan-list 146
```

Verification

Note

VLAN filters on the Catalyst switches apply to any traffic ingress on the VLAN they attach to. The concept of a VLAN filter is an extension of access-lists to a whole bridged domain. It is important to remember the following key features of VLAN filters:

1) VLAN filters are ingress and thus conflict with any other ingress filtering features, such as port access-lists, either IP or MAC-based, and ingress SVI access-lists. You can still use outbound IP access-lists on SVIs.

2) VLAN filters are organized as a sequence of entries. Each entry matches either an IP or MAC access-list and you can specify an action, which is either forward or drop. Optionally, you can omit matching any access-list and then the entry would matches all IP and non-IP traffic.

Copyright © 2010 Internetwork Expert

If a packet does not match the entry, the next vlan-filter entry in sequence is attempted. Note that if a packet is denied in the access-list, it does not automatically mean it is going to be dropped. It just does not match the particular VLAN filter entry.

3) VLAN filters distinguish between IP and non-IP traffic based on the access-list matched. By default, a VLAN filter permits both types of traffic, until you match an access-list in a vlan-filter entry. If there is a particular type of access-list in a vlan-filter (e.g. IP access-list) then all non-matching traffic of this type is dropped by the filter. The other type of traffic (e.g. non-IP) can still pass the filter if there are no explicit access-lists of this type.

4) VLAN filters apply to both local traffic (inside the VLAN) and transit traffic. Also remember to re-apply your vlan-filter once you have changed its configuration since those filters are programmed in hardware.

This configuration can be verified as follows:

```
Rack1R4#ping 155.1.146.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.1, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)
```

ICMP traffic from R4 to R1 is dropped as it enters SW1.

```
Rack1R4#telnet 155.1.146.1
Trying 155.1.146.1 ... Open


User Access Verification

Password:
Rack1R1>
```

TCP traffic, such as the above telnet session, is still allowed.

Copyright © 2010 Internetwork Expert

11.20 VLAN Filters for Non-IP Traffic

- Allow only the following Layer 2 protocols across VLAN 146
  - STP BPDUs (consider both ISL and 802.1q trunks)
  - CDP, VTP, and UDLD
  - ARP
- Disallow any other Layer 2 packets.

Configuration

```
SW1:
!
! The list below matches STP and ARP packets on the VLAN
!
mac access-list extended ALLOWED_L2_TRAFFIC
 permit any any lsap 0x4242 0x0
 permit any any 0x010B 0x0
 permit any any 0x806 0x0
!
! The list is then applied using the vlan filter
!
vlan access-map VLAN146_FILTER 20
 match mac address ALLOWED_L2_TRAFFIC
 action forward
!
! Apply the filter to the VLAN
!
vlan filter VLAN146_FILTER vlan-list 146
```

Verification

Note

When you perform Layer 2 filtering, you usually do that based on Ethertypes or LSAP (Link Service Access Point) values. LSAP value consists of two parts (one byte each): SSAP and DSAP (Source SAP) and (Destination SAP). You can apply Layer 2 filtering based on MAC addresses, but this is not very useful since MAC access-lists only match non-IP traffic (e.g. ARP, STP, VTP). A list of the most commonly seen Layer 2 protocols is outlined below:

IEEE STP BPDUs: These BPDUs use 802.2 LLC encapsulation with SSAP/DSSP values of 0x42 or LSAP value of 0x4242. Cisco switches and routers run IEEE STP over access-ports and run it across the native VLAN of a 802.1q trunk. You can also see STP packets sent across ISL trunks using the same LSAP value of 0x42. The Cisco implementation does not modify the STP packets sent over ISL trunks, it just adds the ISL encapsulation.

Copyright © 2010 Internetwork Expert

PVST+ SSTP BPDUs: These BPDUs use 802.2 SNAP encapsulation (LSAP=0xAAAA) with SNAP PID (Protocol ID) value of 0x010B. You can match this Protocol ID value as the Ethertype number in MAC ACLs in the Catalyst Switches. Cisco switches send PVST+ BPDUs across 802.1q only trunks in addition to IEEE STP BPDUs. The PVST+ BPDUs appear on native as well as non-native VLANs of a trunk port.

ARP protocol: This uses an Ethernet II frame format with the Ethertype value of 0x806

The protocols below use 802.2 SNAP encapsulation with the SNAP Protocol ID values listed below:

VTP: 0x2003
CDP: 0x2000
DTP: 0x2004
UDLD: 0x0111

All SNAP-encapsulated packets can be matched using an LSAP value of 0xAAAA. The above mentioned packet types have no VLAN tag header, so you can filter them on the native VLAN of a trunk, which is usually VLAN 1.

Pitfall

Be very careful when filtering Layer 2 protocols as you can block STP BPDUs and effectively introduce topology loops and broadcast traffic storms.

Note

For verification, make sure that you can see SW1 correctly executing the STP protocol. To accomplish this, configure SW2 as a STP root bridge and make sure that SW1 blocks redundant uplink ports.

```
SW2:
spanning-tree vlan 146 root primary
```

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show spanning-tree vlan 146

VLAN0146
  Spanning tree enabled protocol ieee
  Root ID     Priority    24722
              Address     0019.55af.7100
              Cost        19
              Port        15 (FastEthernet0/13)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32914  (priority 32768 sys-id-ext 146)
              Address     001a.a174.1e00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 15

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/1            Desg FWD 19        128.3    P2p
Fa0/13           Root LRN 19        128.15   P2p
Fa0/14           Altn BLK 19        128.16   P2p
Fa0/15           Altn BLK 19        128.17   P2p
Fa0/16           Altn BLK 19        128.18   P2p
Fa0/17           Altn BLK 19        128.19   P2p
Fa0/18           Altn BLK 19        128.20   P2p

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------

Fa0/19           Altn BLK 19        128.21   P2p
Fa0/20           Altn BLK 19        128.22   P2p
Fa0/21           Altn BLK 19        128.23   P2p
```

Now try running some protocol that the Catalyst switches treat as non-IP, e.g. IPX. Configure IPX on R1 and R4 and try pinging R1's IPX addresses from R4:

```
R1:
ipx routing
!
interface FastEthernet 0/0
 ipx network 146 encapsulation snap

R4:
ipx routing
!
interface FastEthernet 0/1
 ipx network 146 encapsulation snap

Rack1R1#show ipx interface fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPX address is 146.000d.edc8.4f60, SNAP [up]
  Delay of this IPX network, in ticks is 1
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#ping 146.000d.edc8.4f60

Translating "146.000d.edc8.4f60"

Type escape sequence to abort.
Sending 5, 100-byte IPX Novell Echoes to 146.000d.edc8.4f60, timeout is
2 seconds:
 ...
Success rate is 0 percent (0/5)
```

The IPX packets are filtered as they arrive into VLAN 146 on SW1. Now remove the filter on SW1 and try pinging again.

```
Rack1SW1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#no vlan filter VLAN146_FILTER vlan-list 146
Rack1SW1(config)#

Rack1R4#ping 146.000d.edc8.4f60

Translating "146.000d.edc8.4f60"

Type escape sequence to abort.
Sending 5, 100-byte IPX Novell Echoes to 146.000d.edc8.4f60, timeout is
2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

Copyright © 2010 Internetwork Expert

11.21 Port Security

- Configure the respective switches to guard VLAN146 from MAC address flooding attacks originated at R1, R4, or R6.
- Limit the number of MAC addresses learned simultaneously on a single port to one.
- In case of a policy violation, apply the shutdown action on the port connected to R1 but recover the port after 3 minutes.
- For the port connected to R4, drop offending packets and generate log records of the violation.
- For the port connected to R6, simply drop the offending traffic.
- Age the learned secure entries after 10 minutes of inactivity.
- Retain the MAC addresses learned on the port connected to R4 in the switch configuration.

Configuration

```
SW1:
!
! R1 will be shut down in case of violation
!
interface FastEthernet0/1
 switchport mode access
 switchport port-security
 switchport port-security aging time 10
 switchport port-security aging type inactivity
!
! Recover the port from shutdown after 3 minutes
!
errdisable recovery cause psecure-violation
errdisable recovery interval 180

SW2:
!
! Protect the switch from port-security violations
! R6 connects via a trunk port so we specify the total amount
! plus the per-VLAN limits
!
interface FastEthernet0/6
 switchport mode trunk
 switchport port-security
 switchport port-security aging time 10
 switchport port-security aging type inactivity
 switchport port-security violation protect
 switchport port-security maximum 1 vlan 146
 switchport port-security maximum 1 vlan 67
 switchport port-security maximum 2
```

Copyright © 2010 Internetwork Expert

```
SW4:
!
! Learn tha MAC address of R4 as sticky
! Generate logging messages from violating packets
!
interface FastEthernet0/4
 switchport mode access
 switchport port-security
 switchport port-security aging time 10
 switchport port-security aging type inactivity
 switchport port-security violation restrict
 switchport port-security mac-address sticky
```

Verification

Note

Port Security is a Layer 2 feature that enforces a limit on the number of MAC addresses allowed per a particular switch port. The two main purposes of this feature is preventing unauthorized connections on a shared port and thwarting MAC-address flooding attacks. The MAC address flooding attack consists of sending a barrage of packets with different MAC addresses forcing the switch to overpopulate its MAC address table. This attack may result in switch performance degradation and privacy violation. The latter may occur in cases when the switch starts behaving like a hub, flooding frames out all ports. This occurs when the MAC address table overflows.

Note that port-security works only on ports configured as static access or static trunks. The port-security feature does not work on dynamic ports. Port-security logic is outlined below.

1) On a port configured for port-security, the switch keeps a table of secure MAC address entries. The total number of entries allowed in this table is configurable using the following command:

switchport port-security maximum-address <N>

On trunk ports, the above command specifies the maximum number of MAC addresses active on all VLANs at the same time – the aggregate limit. Note that the switch treats the same MAC address on different VLANs as two different MAC addresses. For trunk ports, you may additionally specify the maximum number of MACs per VLAN using the command:

switchport port-security maximum <N> vlan <VLAN>

Copyright © 2010 Internetwork Expert

If the port is an access-port configured with access and voice VLANs, you can use commands to impose restrictions on just two VLANs, without ever mentioning their numbers:

```
switchport port-security maximum <N> vlan {access|voice}
```

When a switch has learned all allowed addresses on the port or a VLAN and a frame with a new source MAC address arrives on the port, the switch may take any of the following actions:

a) Shutdown the port (the default “shutdown” action).

b) Silently discard the frame (if configured for the “protect” action). Protect mode is not recommended for trunks ports, although we use it in this task. The problem is that as soon as any VLAN on a trunk reaches its MAC address limit, the port stops learning MAC addresses on any other VLAN. The worst thing about this mode is that the switch does not notify you about this via any logging message.

c) Discard the frame and generate a syslog message or SNMP trap (if configured for the “restrict” action). You may need to configure SNMP destination hosts to send the actual traps.

2) The switch does not allow the same MAC address to appear on more than one secure port at the same time. Thus, if a switch has learned a MAC address on a secure port it will not allow the same address to appear on other switch ports until the secure entry has expired.

3) The switch ages out secure MAC address entries using a configurable timeout. You can set this timeout per-port using two commands:

```
switchport port-security aging timeout <TIMEOUT>
switchport port-security aging type {absolute|inactivity}
```

Together these commands define the aging behavior. Absolute aging means age each entry starting at the time it has been learned. Inactivity aging starts aging time only after each frame is received. If no frames have been received during the timeout, the switch removes the secure entry and opens a slot for a different MAC address.

Copyright © 2010 Internetwork Expert

If the port-security feature has shut down a port, the port can be restored to an operational state using the error-disable recovery procedure. You can set one global recovery timeout (used for all types of error-disable recovery) by using the command:

```
errdisable recovery interval <seconds>
```

In order to enable a particular type of recovery procedure use the following command:

```
errdisable recovery cause <cause>
```

In addition to setting up dynamic learning of secure MAC addresses, you may configure static secure MAC address entries using the interface-level command:

```
switchport port-security mac-address H.H.H
```

These entries also count against the maximum number of allowed MAC addresses on an interface. You may configure a port to age static secure MAC address entries as well using the command:

```
switchport port-security aging static
```

This may be useful when you need to set up guaranteed access for a specific MAC address for some amount of time.

The last port-security feature is known as sticky learning. It allows you to learn “static” secure MAC address entries. When a switch learns new MAC addresses on a port in sticky mode, it generates a configuration line for a corresponding static entry. This line appears in the running configuration so you need to save it to make the static entry truly permanent.

For verification in our task, display the port-security configuration for all ports configured:

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show port-security interface fastEthernet 0/1
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Shutdown
Aging Time                 : 10 mins
Aging Type                 : Inactivity
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 1
Configured MAC Addresses   : 0
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 000c.31ef.4e60:146
Security Violation Count   : 0

Rack1SW2#show port-security interface fastEthernet 0/6
Rack1SW2#show port-security interface fastEthernet 0/6
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Protect
Aging Time                 : 10 mins
Aging Type                 : Inactivity
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 2
Total MAC Addresses        : 2
Configured MAC Addresses   : 0
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 0011.200a.f000:146
Security Violation Count   : 0

Rack1SW2#show port-security interface fastEthernet 0/6 vlan 146
Default maximum: not set, using 6272
VLAN  Maximum    Current
  146          1          1

Rack1SW2#show port-security interface fastEthernet 0/6 vlan 67
Default maximum: not set, using 6272
VLAN  Maximum    Current
   67          1          1

Rack1SW4#show port-security interface fastEthernet 0/4
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Restrict
Aging Time                 : 10 mins
Aging Type                 : Inactivity
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 1
Configured MAC Addresses   : 0
Sticky MAC Addresses       : 1
Last Source Address:Vlan   : 000f.23d5.2e81:146
Security Violation Count   : 0
```

Copyright © 2010 Internetwork Expert

Verify the sticky MAC address configuration:

```
Rack1SW4#show running-config interface fastEthernet 0/4
Building configuration...

Current configuration : 349 bytes
!
interface FastEthernet0/4
 switchport access vlan 146
 switchport mode access
 switchport port-security
 switchport port-security aging time 10
 switchport port-security violation restrict
 switchport port-security aging type inactivity
 switchport port-security mac-address sticky
 switchport port-security mac-address sticky 000f.23d5.2e81
```

Configure R4 to violate port-security. For this to happen, change the MAC address of the port on R4 connected to SW4. This will prevent R4 from communicating with other hosts until the old secure entry expires.

```
Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface fastEthernet 0/1
Rack1R4(config-if)#mac-address 000f.23d5.2e82

Rack1SW4#
%PORT_SECURITY-2-PSECURE_VIOLATION: Security violation occurred, caused by MAC
address 000f.23d5.2e82 on port FastEthernet0/4.
%PORT_SECURITY-2-PSECURE_VIOLATION: Security violation occurred, caused by MAC
address 000f.23d5.2e82 on port FastEthernet0/4.
```

Copyright © 2010 Internetwork Expert

11.22 HSRP and Port-Security

- Reconfigure R3 and R4 temporarily for this task.
- Configure R1 and R3 to emulate an HSRP virtual router on VLAN 146 with the IP address 155.1.146.254.
- Configure the R4's Fa0/1 interface and the R6's VLAN 146 interface to emulate a virtual HSRP router on VLAN 146 with the IP address 155.1.146.253.
- Ensure your new configuration works with the port-security enabled ports.
- Do not use the **`switchport port-security maximum`** command on the port connected to R4.

Configuration

```
R1:
interface FastEthernet 0/0
 standby 1 ip 155.1.146.254

R3:
interface FastEthernet 0/1
 standby 1 ip
 no shutdown

R4:
!
! Temporarily shutdown R4's VLAN146 interface
!
interface FastEthernet 0/1
 shutdown
!
! We use a different group and IP for R4 and R6
! as they share the same VLAN as R1 and R3
!
! Allow R4 to use burned in MAC address for HSRP
!
interface FastEthernet 0/0
 ip address 155.1.146.4 255.255.255.0
 standby 2 ip
 standby use-bia

R6:
!
! Allow R6 to use burned in MAC address for HSRP
!
interface FastEthernet 0/0.146
 standby 2 ip 155.1.146.253
 standby use-bia
```

Copyright © 2010 Internetwork Expert

```
SW1:
!
! Allow R1 to use a virtual MAC address
!
interface FastEthernet 0/1
 switchport port-security maximum 2
!
! Temporarily reconfigure R3's connection
!
interface FastEthernet 0/3
 switchport access vlan 146
 switchport mode access

SW2:
!
! Temporarily reconfigure R4's connection
!
interface FastEthernet 0/4
 switchport access vlan 146
 switchport mode access
```

Verification

> **Note**
>
> Virtual router protocols (HSRP, VRRP and GLBP) use virtual MAC addresses in addition to physical interface addresses. This requires small changes in port-security configurations. At the very least, you need to increase the maximum MAC address count to permit additional entries in the secure MAC address table. HSRP offers another option allowing you to use only physical MAC-addresses, even for the virtual IP address.
>
> First look at the output of the switch show commands and logging buffer contents before you changed the MAC address limit on SW1. Also, check the HSRP status on R4 before you enable the use of a BIA on R6:

```
Rack1SW1#
%PORT_SECURITY-2-PSECURE_VIOLATION: Security violation occurred, caused
by MAC address 0000.0c07.ac01 on port FastEthernet0/1.
```

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show port-security interface fastEthernet 0/1
Port Security              : Enabled
Port Status                : Secure-shutdown
Violation Mode             : Shutdown
Aging Time                 : 10 mins
Aging Type                 : Inactivity
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 0
Configured MAC Addresses   : 0
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 0000.0c07.ac01:146
Security Violation Count   : 1
```

```
Rack1R4#show standby
FastEthernet0/0 - Group 2
  State is Learn
  Virtual IP address is unknown
  Active virtual MAC address is unknown
    Local virtual MAC address is 0000.0c07.ac02 (v1 default)
  Hello time 3 sec, hold time 10 sec
  Preemption disabled
  Active router is unknown
  Standby router is unknown
  Priority 100 (default 100)
  IP redundancy name is "hsrp-Fa0/0-2" (default)
```

Check the state of the R4 HSRP group once again and note the virtual MAC address, it is the address of R4:

```
Rack1R4#show standby
FastEthernet0/0 - Group 2
  State is Active
    2 state changes, last state change 00:13:50
  Virtual IP address is 155.1.146.253 (learnt)
  Active virtual MAC address is 000f.23d5.2e80
    Local virtual MAC address is 000f.23d5.2e80 (bia)
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 0.893 secs
  Preemption disabled
  Active router is local
  Standby router is 155.1.146.6, priority 100 (expires in 7.857 sec)
  Priority 100 (default 100)
  IP redundancy name is "hsrp-Fa0/0-2" (default)
```

Copyright © 2010 Internetwork Expert

In addition, R1's port should recover by this time and show the total count of MAC addresses as two:

```
Rack1SW1#
%PM-4-ERR_RECOVER: Attempting to recover from psecure-violation err-
disable state on Fa0/1
%LINK-3-UPDOWN: Interface FastEthernet0/1, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1,
changed state to up

Rack1SW1#show port-security interface fastEthernet 0/1
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Shutdown
Aging Time                 : 10 mins
Aging Type                 : Inactivity
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 2
Total MAC Addresses        : 2
Configured MAC Addresses   : 0
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 000c.31ef.4e60:146
Security Violation Count   : 0
```

Revert all the changes we made so far for R4 and R3 to be part of VLAN 146.

Copyright © 2010 Internetwork Expert

11.23 DHCP Snooping

- Configure R4 as a DHCP server and R1 as a DHCP client on VLAN 146.
- Configure SW1 to prevent potential DHCP attacks in such a way that it dynamically accounts for future hosts added to the VLAN 146 segment.
- SW1 should store the binding database in flash with the filename dhcp-bindings.txt, and use a 15 second delay between changes.
- Limit the amount of DHCP messages SW1 can receive from R1 to 10 per second.

Configuration

```
R1:
interface FastEthernet 0/0
 ip address dhcp

R4:
ip dhcp relay information trust-all
!
ip dhcp pool VLAN146
 network 155.1.146.0 /24

SW1:
ip dhcp snooping vlan 146
ip dhcp snooping
ip dhcp snooping database flash:/dhcp-bindings.txt
ip dhcp snooping database write-delay 15
!
interface FastEthernet 0/1
 ip dhcp snooping limit rate 10
!
! Trust DHCP on uplink that connect SW1 to R4
!
interface range FastEthernet 0/13 - 21
 ip dhcp snooping trust
```

Verification

Note

DHCP Snooping is a security feature that inspects DHCP packets transiting a Layer 2 switch. The switch itself does not participate in DHCP packet exchange but enforces DHCP packet flow integrity. Before we start with an overview of DHCP snooping, let us quickly recap different agent roles for IP address allocation procedures using DHCP.

Copyright © 2010 Internetwork Expert

1) A DHCP client discovers a DHCP server and requests an IP address as well as other configuration parameters using a DHCPDISCOVER packet. The client may then select an IP address using a DHCPREQUEST message. When the client no longer needs the IP address it returns it back to the server using a DHCPRELEASE packet.

2) A DHCP Relay acts as a middleman agent between the DHCP clients and the server. The primary function of a DHCP Relay is to forward messages from local clients to the remote DHCP Server. When a DHCP Relay receives a broadcast packet from a connected client it forwards it as a unicast packet to the IP address of the server. In the Cisco IOS, you can specify the IP address of the server using the interface-level command **`ip helper-address X.X.X.X`**. The DHCP relay changes the "giaddr" field in DHCP packets from zero to the IP address of the interface that forwards the packet. The DHCP server later uses this IP address to respond with a DHCP packet. A Catalyst switch may act as a DHCP Relay if you configure it as a Layer 3 device with an IP address and helper-address on some interface.

3) A DHCP Server responds to discover and request messages from DHCP clients with DHCPOFFER and DHCPACK/DHCPNAK messages. The former message contains the IP address for the client and the latter two acknowledge or reject a request message. The server uses the "giaddr" field to respond back provided that it is non-zero. A server may also explicitly request a client to release and IP address with a DHCPRELEASEREQUERY message.

Every DHCP message also carries the MAC address of the client explicitly stored in one of DHCP fields. This MAC address may identify a client to the server if the Client ID field is set in the packet.

The primary goal of using DHCP Snooping is to enforce DHCP security. When you enable DHCP Snooping on a switch, it starts treating every port as connected to a DHCP client (e.g. workstation, or some downstream switch). For such ports, also called "untrusted" ports, the switch applies DHCP message filtering, only accepting messages expected from DHCP Clients (DHCPREQUEST, DHCPDISCOVER, DHCPRELEASE).

The switch allows any type of DHCP messages on trusted ports. You may explicitly configure a port as trusted by the DHCP Snooping process using the interface-level command **`ip dhcp snooping trust`**. Usually, you need this command on the ports connected to DHCP servers or uplink ports on access switches.

Copyright © 2010 Internetwork Expert

On any type of port you can configure the command **ip dhcp snooping limit rate <pps>** to control the number of DHCP packets per second received on a particular port. If the port exceeds the threshold set, the switch will put the port into an error-disabled state. The port can be recovered from this state using common error-disable recovery procedures.

In addition, if an incoming DHCP message contains a non-zero "giaddr", meaning it has been relayed, then it must be received on a trusted port. Untrusted ports simply discard such messages.

In addition to filtering messages based on their types, DHCP snooping also populates a special DHCP snooping table based on messages exchanged across untrusted ports. This table contains the IP addresses that were allocated to clients by the DHCP server and other information such as client MAC address, client port, VLAN number, and lease duration. The switch consults this table when it receives DHCP messages such as DHCPRELEASE to make sure that no one would spoof a DHCP message for another host. The switch also ensures that MAC addresses in a DHCP packet match the MAC address of the host sending the message. This feature could be disabled using the command **no ip dhcp snooping verify mac-address**. Note that messages received on trusted ports do not create any DHCP Snooping binding entries.

You may instruct the switch to save the DHCP snooping database contents in flash memory or on external server (e.g. via TFTP). This allows the switch to preserve DHCP binding mappings across reloads. The global command to specify the external storage is **ip dhcp snooping database**. You may also control the interval between database updates using the command **ip dhcp snooping database write-interval <seconds>**.

In order to start the DHCP Snooping process, you need to enable it globally as well as activate it per-VLAN. You need to do *both* in order for DHCP Snooping to start working. Note that by default, when DHCP Snooping is enabled, the switch also adds DHCP Information Option (or Option 82) to all received packets. The purpose of this option is to identify the device and port that the client connects to. In short, the option contains the ID of the switch (e.g. MAC address or hostname) that inserted this option and the port identifier. This can be an interface name that the DHCP Client connects to. This information allows the DHCP Server to allocate IP addresses based on additional information, for example, split a subnet between two port ranges in the switch that inserted the option.

Copyright © 2010 Internetwork Expert

One issue that may arise here is that the switch inserts the option but leaves the "giaddr" field at zero. Thus, a DHCP Server may assume that option has been formatted incorrectly, since a DHCP Relay is supposed to set the "giaddr" field to its own IP address. Thus, an IOS DHCP server will reject by default such DHCP messages. To overcome this issue you may use either of the following methods:

1) Instruct the IOS DHCP Server to accept DHCP messages with a zero "giaddr" using the global command: **`ip dhcp relay information trust-all`** or the interface-level command **`ip dhcp relay information trusted`**.

2) Configure the DHCP Snooping feature in the switch not to insert Option 82. This is accomplished using the command: **`no ip dhcp-snooping information option`**

3) Trust the port where you receive the original DHCP message. The DHCP Snooping feature does not insert any Information Option into the received packets.

Here we verify the status of DHCP snooping on SW1:

```
Rack1SW1#show ip dhcp snooping
Switch DHCP snooping is enabled
DHCP snooping is configured on following VLANs:
146
Insertion of option 82 is enabled
   circuit-id format: vlan-mod-port
    remote-id format: MAC
Option 82 on untrusted port is not allowed
Verification of hwaddr field is enabled
Interface                    Trusted     Rate limit (pps)
------------------------     -------     ----------------
FastEthernet0/1              no          10
FastEthernet0/13             yes         unlimited
FastEthernet0/14             yes         unlimited
FastEthernet0/15             yes         unlimited
FastEthernet0/16             yes         unlimited
FastEthernet0/17             yes         unlimited
FastEthernet0/18             yes         unlimited
FastEthernet0/19             yes         unlimited
FastEthernet0/20             yes         unlimited
FastEthernet0/21             yes         unlimited
```

Copyright © 2010 Internetwork Expert

```
Rack1SW1#show ip dhcp snooping binding
MacAddress          IpAddress        Lease(sec)  Type           VLAN
Interface
------------------  ---------------  ----------  -------------  ----
00:0C:31:EF:4E:60   155.1.146.5      85188       dhcp-snooping  146
FastEthernet0/1
Total number of bindings: 1
```

Note the limits set for the port connected to R1. You can also check the DHCP Snooping database contents using the command below:

```
Rack1SW1#show ip dhcp snooping database detail
Agent URL : flash:/dhcp-bindings.txt
Write delay Timer : 15 seconds
Abort Timer : 300 seconds

Agent Running : No
Delay Timer Expiry : Not Running
Abort Timer Expiry : Not Running

Last Succeded Time : 00:31:44 UTC Mon Mar 1 1993
Last Failed Time : None
Last Failed Reason : No failure recorded.

Total Attempts       :        2   Startup Failures :        0
Successful Transfers :        2   Failed Transfers :        0
Successful Reads     :        1   Failed Reads     :        0
Successful Writes    :        1   Failed Writes    :        0
Media Failures       :        0

First successful access: Read

Last ignored bindings counters :
Binding Collisions    :        0   Expired leases    :        0
Invalid interfaces    :        0   Unsupported vlans :        0
Parse failures        :        0
Last Ignored Time : None

Total ignored bindings counters:
Binding Collisions    :        0   Expired leases    :        0
Invalid interfaces    :        0   Unsupported vlans :        0
Parse failures        :        0
```

Copyright © 2010 Internetwork Expert

```
Rack1SW1#more flash:/dhcp-bindings.txt
2b915970
TYPE DHCP-SNOOPING
VERSION 1
BEGIN
155.1.146.5 146 000c.31ef.4e60 2B92AAB2 Fa0/1
4ba698f0
END
```

The last command shows the raw database contents, as they are stored in flash memory.

Copyright © 2010 Internetwork Expert

11.24 DHCP Snooping and the Information Option

- Configure R6 to obtain IP address information via DHCP from R4.
- Ensure that SW2 inserts Option 82 in DHCP requests received from R6.
- SW2 should be configured so that it uses the string “SWITCH2” for Option 82 Remote-ID and the string “ROUTER6” as the Circuit-ID for the port connected to R6.
- SW2 should accept information options in DHCP packets even on non-trusted ports.

Configuration

```
R6:
interface FastEthernet 0/0.146
 ip address dhcp

SW2:
interface FastEthernet0/6
 ip dhcp snooping vlan 146 information option format-type circuit-id
string ROUTER6
!
ip dhcp snooping information option format remote-id string SWITCH2
ip dhcp snooping information option allow-untrusted
!
ip dhcp snooping
ip dhcp snooping vlan 146
!
interface range FastEthernet 0/13 - 18 , FastEthernet 0/21
 ip dhcp snooping trust
```

Verification

Note

DHCP Information Option (Option 82) is used in large Enterprise and Metro Ethernet deployments to enhance functionality of IP address allocation processes. The option itself has no strict format, rather the RFC defines a very flexible structure for a relay agent to insert any information in this option. Two commonly defined and used sub-fields are Remote-ID and Circuit-ID. The first field identifies the remote device that inserted the option – that is, the DHCP relay that redirected the original request. The second field identifies the point of client’s attachment, which is commonly a port name or some other port identifier.

Copyright © 2010 Internetwork Expert

Over time, Cisco has been changing the format of Option 82 used in their switches. The most recent code uses the hostname as the default remote ID, but this can be changed to any ASCII string. For every port, you can also set a per-VLAN circuit ID as another ASCII string. This allows the use of different IDs for different VLANs on a trunk port.

The switch does not accept DHCP packets with a non-zero "giaddr" on untrusted ports. Remember, these are the ports facing DHCP clients. In addition, it does not accept DHCP packets with Option 82 on untrusted ports. However, the latter behavior can be modified using the command: **`ip dhcp snooping information option allow-untrusted`** which allows the switch to receive DHCP packets with Option 82 even on untrusted ports.

However, packets with a "giaddr" of zero are still rejected by the switch even with this command active in the configuration. You need to mark the respective port as "trusted" in order to accept such packets.

When you trust a port for DHCP Snooping operations, the switch does not create a snooping entry nor does it add any information option by itself for IP packets received on the trusted port.

For verification in our case, enable a DHCP packet content dump on R4 to read the Option 82 contents:

Copyright © 2010 Internetwork Expert

```
Rack1R4(config)#access-list 100 permit udp any any eq bootps

Rack1R4#debug ip packet 100 dump

IP: s=0.0.0.0 (FastEthernet0/1), d=255.255.255.255, len 352, rcvd 2
07E3C1A0:                     FFFF FFFFFFFF            . ...
07E3C1B0: 0011200A F0000800 45000160 87D00000  .. .p...E..`.P..
07E3C1C0: FE1133BD 00000000 FFFFFFFF 00440043  ~.3=. .......D.C
07E3C1D0: 014CBE6D 01010600 00001907 00008000  .L>m. .........
07E3C1E0: 00000000 00000000 00000000 00000000  . ............
07E3C1F0: 0011200A F0000000 00000000 00000000  .. .p. ........
07E3C200: 00000000 00000000 00000000 00000000  . ............
07E3C210: 00000000 00000000 00000000 00000000  . ............
07E3C220: 00000000 00000000 00000000 00000000  . ............
07E3C230: 00000000 00000000 00000000 00000000  . ............
07E3C240: 00000000 00000000 00000000 00000000  . ............
07E3C250: 00000000 00000000 00000000 00000000  . ............
07E3C260: 00000000 00000000 00000000 00000000  . ............
07E3C270: 00000000 00000000 00000000 00000000  . ............
07E3C280: 00000000 00000000 00000000 00000000  . ............
07E3C290: 00000000 00000000 00000000 00000000  . ............
07E3C2A0: 00000000 00000000 00000000 00000000  . ............
07E3C2B0: 00000000 00000000 00000000 00000000  . ............
07E3C2C0: 63825363 35010139 0204803D 1F006369  c.Sc5..9...=..ci
07E3C2D0: 73636F2D 30303131 2E323030 612E6630  sco-0011.200a.f0
07E3C2E0: 30302D46 61302F30 2E313436 0C075261  00-Fa0/0.146..Ra
07E3C2F0: 636B3152 36370801 060F2C03 21962B52  ck1R67....,.!.+R
07E3C300: 16010901 07524F55 54455236 02090107  . ...ROUTER6....
07E3C310: 53574954 434832FF                    SWITCH2.
```

Note the Client-ID that contains the name of the VLAN 146 interface of R6 and the Option 82 contents containing the two strings configured in SW2.

Now try simulating a DHCP request with a non-zero "giaddr" field across SW2. To accomplish this, configure a new DHCP pool in R4 and request a DHCP address for SW3 on VLAN 79:

```
R4:
ip dhcp pool VLAN79
 network 155.1.79.0 /24

SW1:
interface Vlan79
 ip helper-address 155.1.146.4

SW3:
interface Vlan 79
 ip address dhcp

Rack1SW2#debug ip dhcp snooping event
DHCP Snooping Event debugging is on

DHCP_SNOOPING: received new DHCP packet from input interface
(FastEthernet0/6)
DHCP_SNOOPING: process new DHCP packet, message type: DHCPDISCOVER,
input interface: Fa0/6, MAC da: 0015.c634.6c61, MAC sa: 000c.ce65.50a0,
IP da: 155.1.146.4, IP sa: 155.1.79.7, DHCP ciaddr: 0.0.0.0, DHCP
yiaddr: 0.0.0.0, DHCP siaddr: 0.0.0.0, DHCP giaddr: 155.1.79.7, DHCP
chaddr: 000f.f76d.ac80
%DHCP_SNOOPING-5-DHCP_SNOOPING_NONZERO_GIADDR: DHCP_SNOOPING drop
message with non-zero giaddr or option82 value on untrusted port,
message type: DHCPDISCOVER, MAC sa: 000c.ce65.50a0
```

As the debugging output reveals, the switch does not accept those packets on untrusted ports. It would still accept the packets with a zero "giaddr". These are the packets with a DHCP option inserted by some other Layer 2 device.

Copyright © 2010 Internetwork Expert

11.25 Dynamic ARP Inspection

- Configure SW2 to prevent ARP poisoning attacks on VLAN 146.
- Do not trust any trunk ports to other switches, but ensure that the switch enforces ARP security for R4 and R1.
- Log all ARP packets permitted by the ARP access-list but limit their rate to four per ten seconds.
- Additionally, log all packets matched against the DHCP snooping database.
- Store, at maximum, 16 entries in the ARP logging buffer.
- Enable all additional sanity checks for ARP packets.

Configuration

```
SW2:
!
! ARP Access-List
!
arp access-list ARP_VLAN146
 permit ip host 155.1.146.1 mac host 000d.edc8.4f60 log
 permit ip host 155.1.146.4 mac host 0015.c634.6c61 log
!
! Apply ARP Inspection
!
ip arp inspection vlan 146
ip arp inspection vlan 146 logging acl-match matchlog
ip arp inspection log-buffer entries 16
ip arp inspection log-buffer logs 4 interval 10
ip arp inspection validate src-mac dst-mac ip
!
! Apply the ARP ACL
!
ip arp inspection filter ARP_VLAN146 vlan  146
```

Verification

Note

Dynamic ARP inspection is a security feature that fixes some well-known weaknesses of the ARP protocol. Generally, ARP operates on a broadcast Ethernet segment, and allows any host to spoof a MAC address for any IP address on the segment. These attacks, commonly known as “Man-in-the-Middle” (MiM) attacks, cannot be prevented by using just port-security, access-lists, or other well-known features.

Copyright © 2010 Internetwork Expert

ARP Inspection creates a special IP to MAC address binding table in the switch. This table is dynamically populated based on the DHCP snooping database contents. You can also add static entries to the database manually, using the configuration commands for ARP Inspection access-lists.

When the switch receives an ARP packet on an ARP-untrusted (the default state) port it inspects the packet contents. Based on the IP to MAC address binding information in the packet, the switch permits the packet only if it matches the ARP Inspection table. This prevents ARP poisoning attacks.

Note that implementing ARP Inspection may break some services, such as Proxy ARP. To resolve these issues, ARP Inspection allows you to configure some ports as trusted for ARP Inspection. On trusted ports, the switch does not inspect any ARP message. It is common to trust ARP messages on switch uplink ports, pointing toward the network core. The command to make a port trusted is **`ip arp inspection trust`**.

The ARP Inspection feature may perform additional checks on ARP packets. It can check that destination MAC addresses inside ARP packet bodies match the destination MAC address in Ethernet frames for ARP responses. In addition to this, the switch check source MAC addresses in ARP packets and Ethernet frames for ARP requests. You can enable these two checks using the commands

```
ip arp inspection validate src-mac
ip arp inspection validate dst-mac
```

Additionally, you may enable IP address consistency checks for ARP packets using the command **`ip arp inspection validate ip.`** This command checks source/destination IP addresses for consistency. Specifically, it ensures that no host tries to bind MAC addresses to IP addresses such as "255.255.255.255" or "0.0.0.0".

When enabled by default, the IP ARP Inspection feature builds all ARP mapping information based on the DHCP bindings table. If there are hosts on the segment not using DHCP for address allocation, you need to configure ARP access-lists. The syntax to create and apply a filter is:

```
arp access-list <ACL_NAME>
 permit [request|response] ip <address> <mask> mac
<address> <mask> [log]

ip arp inspection filter <ACL_NAME> vlan <VLAN_ID> [static]
```

Copyright © 2010 Internetwork Expert

For the access-list, you specify an IP address and MAC address binding. Note that you commonly do not match ARP requests or responses and usually use just a host IP to host MAC address mapping. The ARP Inspection feature first checks the ARP access-list for a given VLAN to see if a given ARP packet is legitimate. If there are no permit matches found for the given IP and MAC address pair, and there is NO explicit **deny ip any mac any** statement in the end of the access-list, the feature also checks the DHCP bindings database. However, if there is an explicit deny statement in the end of access-list or the access-list has been applied with the **static** keyword, then ARP Inspection does not consult the DHCP Snooping database.

Logging of denied or allowed packets is somewhat complicated with ARP Inspection. First, you may specify a **log** keyword in the ARP access-list. Then you need to enable ARP inspection using the command:

```
ip arp inspection vlan <VLAN_ID> logging acl-match
{matchlog|none}
```

This will enable or disable (the default) logging of ARP packets matching ACL entries configured with the **log** keyword. In parallel with the above command, you may configure two additional commands:

```
ip arp inspection vlan <VLAN_ID> logging dhcp-bindings
{all|permit|none}
ip arp inspection vlan <VLAN_ID> logging arp-probe
```

Those two respective commands enable logging of packets matching the DHCP bindings database and the logging of special ARP probe packets (packets with a source IP of 0.0.0.0). By default, the switch logs ARP packets denied by the DHCP snooping database. You may change this mode by disabling the logging of DHCP denied packets or enable logging of permitted packets. You may even log all packets matched against the DHCP snooping entries.

The switch accumulates logged ARP packets in a special internal buffer. You can regulate the size of this buffer using the command:

```
ip arp inspection log-buffer entries <N>
```

The switch empties this buffer using a rate-limited procedure based on the following command settings:

```
ip arp inspection log-buffer logs <number> <interval>
```

Copyright © 2010 Internetwork Expert

Based on these settings, the switch will generate at maximum <*number*> of syslog messages during <*interval*> time. If you set the <*interval*> to zero, the switch will generate a message for every ARP packet to be logged.

The last security feature is ARP message rate-limiting. This feature is on by default on untrusted ports and disabled on trusted ports. On both types of ports you can enable this feature using the command:

ip arp inspection limit rate {<pps> [burst interval <seconds>]|none}

This command will restrict the rate of received ARP packets to <*pps*> per <*seconds*> interval. When the port exceeds this rate, the switch will bring it to the error-disable state. By default, the rate limit on untrusted ports is 15 pps. The feature limits the aggregate rate on trunks or EtherChannels, so you may need to adjust it in real-life situations.

For our tasks, verify that ARP is enabled for the particular VLAN:

```
Rack1SW2#show ip arp inspection vlan 146

Source Mac Validation      : Enabled
Destination Mac Validation : Enabled
IP Address Validation      : Enabled

 Vlan     Configuration    Operation   ACL Match          Static ACL
 ----     -------------    ---------   ---------          ----------
  146     Enabled          Active      ARP_VLAN146        No

 Vlan     ACL Logging      DHCP Logging      Probe Logging
 ----     -----------      ------------      -------------
  146     Acl-Match        Deny              Off
```

Note that ARP inspection is enabled on VLAN 146 and ARP logging for ACL entries is active. By default, DHCP Logging is also enabled.

Now clear the ARP cache on R6 and send ping packets to R1 and R4. These packets should be matched and logged by the ARP ACL.

```
Rack1R6#clear arp-cache
Rack1R6#ping 155.1.146.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#ping 155.1.146.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 m

Rack1SW2#show logging
<snip>

Log Buffer (4096 bytes):

%SW_DAI-6-ACL_PERMIT: 1 ARPs (Res) on Fa0/16, vlan
146.([000d.edc8.4f60/155.1.146.1/000c.ce65.50a0/155.1.146.2/14:27:51
UTC Fri Mar 5 1993])
%SW_DAI-6-ACL_PERMIT: 1 ARPs (Res) on Fa0/16, vlan
146.([0015.c634.6c61/155.1.146.4/000c.ce65.50a0/155.1.146.2/14:27:51
UTC Fri Mar 5 1993])
```

Now change the MAC address of the R6 VLAN 146 interface and observe how the switch denies the violating ARP packets:

```
Rack1R6#show interfaces fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  Hardware is AmdFE, address is 000c.ce65.50a0 (bia 000c.ce65.50a0)
<snip>

Rack1R6#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface fastEthernet 0/0
Rack1R6(config-if)#mac-address 000c.ce65.50a1

Rack1SW2#
%SW_DAI-4-DHCP_SNOOPING_DENY: 1 Invalid ARPs (Req) on Fa0/6, vlan
146.([000c.ce65.50a1/155.1.146.2/0015.c634.6c61/155.1.146.4/14:32:51 UTC Fri
Mar 5 1993])
%SW_DAI-4-DHCP_SNOOPING_DENY: 1 Invalid ARPs (Req) on Fa0/6, vlan
146.([000c.ce65.50a1/155.1.146.2/000d.edc8.4f60/155.1.146.1/14:32:51 UTC Fri
Mar 5 1993])
%SW_DAI-4-DHCP_SNOOPING_DENY: 1 Invalid ARPs (Res) on Fa0/6, vlan
146.([000c.ce65.50a1/155.1.146.2/ffff.ffff.ffff/155.1.146.2/14:32:51 UTC Fri
Mar 5 1993])
```

As you can see, there is no DHCP snooping entry to match the new R6 MAC address, so the ARP packets are dropped by the switch.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.26 IP Source Guard

- Configure SW2 to prevent IP address spoofing on the port connected to R6.
- Ensure that your solution accounts for dynamic IP addresses obtained via DHCP.
- Enforce Layer 2 filtering for the MAC addresses corresponding to secured IP addresses at the same time.
- Do not shutdown ports in case of a security violation.

Configuration

```
SW2:
!
! Note that we set maximum allowed secure MAC addresses to 2
! since we have two VLANs on the trunk
!
interface FastEthernet 0/6
 ip verify source port-security
 switchport port-security maximum 2
 switchport port-security
!
! Enable DHCP snooping on both VLANs to ensure proper filtering
!
ip dhcp snooping
ip dhcp snooping vlan 67,146
!
! Since R6's VLAN67 IP is static, we need a static IP/MAC/Port binding
! The MAC address here is R6's physical MAC address
!
ip source binding 000c.ce65.50a0 vlan 67 155.1.67.6 interface
FastEthernet 0/6
```

Verification

Note

IP Source Guard is a feature intended to prevent IP packet spoofing at Layer 2 (MiM attacks). When you enable IP Source Guard on a port, the switch applies a Layer 3 filter to this port, only accepting the packets with source IP addresses matching DHCP snooping bindings created for the port. Enabling DHCP snooping is a prerequisite, but you may bind an IP address to a port manually, even if the host does not use DHCP.

Copyright © 2010 Internetwork Expert

As soon as you enable IP Source Guard, the switch only permits IP packets that match the DHCP snooping database or static IP to MAC addresses and port bindings. The switch also allows ingress DHCP packets for hosts to obtain IP addresses.

IP Source Guard relieves you from the need of applying any IP ingress filtering on individual ports to prevent IP address spoofing. Combined with Dynamic ARP Inspection and DHCP Snooping it allows much more secure environment than default Layer 2 deployments.

You may enable IP Source Guard per-interface using the command:

`ip verify-source [port-security]`

The **`port-security`** option requires port-security be enabled on the switch port as well. With this option, the switch filters packets both based on the source IP and MAC addresses, and the secure MAC address is taken from the DHCP snooping database or a static mapping entry.

Note that you may enable IP Source Guard on a trunk port as well. In this case, it requires DHCP snooping to be enabled on all trunked VLANs for filtering to work properly.

In our solution, we apply source guard to the ports connected to R6. Note that we need to enable DHCP snooping on both VLANs active on the trunk to ensure proper filtering. In addition, the port should be allowed to permit at least two MAC addresses, as there are two VLANs on the trunk.

Here we verify that IP Source Guard is active on the port:

```
Rack1SW2#show ip verify source
Interface  Filter-type  Filter-mode  IP-address       Mac-address        Vlan
---------  -----------  -----------  ---------------  -----------------  ------
Fa0/6      ip-mac       active       155.1.67.6       00:0C:CE:65:50:A0  67
Fa0/6      ip-mac       active       155.1.146.3      00:0C:CE:65:50:A0  146
```

Copyright © 2010 Internetwork Expert

Ensure filtering actually prevents IP address spoofing by changing the IP address of R6:

```
Rack1R6#ping 155.1.146.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms

Rack1R6#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface fastEthernet 0/0.146
Rack1R6(config-subif)#ip address 155.1.146.254 255.255.255.0
Rack1R6(config-subif)#^Z
Rack1R6#ping 155.1.146.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
Rack1R6#
```

```
Rack1SW2#show ip verify source
Interface  Filter-type  Filter-mode  IP-address       Mac-address        Vlan
---------  -----------  -----------  ---------------  -----------------  ------
Fa0/6      ip-mac       active       155.1.67.6       00:0C:CE:65:50:A0  67
Fa0/6      ip-mac       active       deny-all         deny-all           146
```

Note the IP source guard entry for VLAN 146 has changed to deny all. Only DHCP packets are permitted in this situation, to allow the end static MAC to obtain an IP address.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.27 Using Catalyst Ingress Access-Lists

- Allow R1 to only send ARP and ICMP packets out of its VLAN 146 connection.

Configuration

```
SW1:
!
! Allow ARP traffic only
!
mac access-list extended ARP_ONLY
 permit any any 0x806 0x0
!
! Allow ICMP traffic only
!
ip access-list extended ICMP_ONLY
 permit icmp any any
!
! Apply the access-list
!
interface FastEthernet0/1
 ip access-group ICMP_ONLY in
 mac access-group ARP_ONLY in
```

Verification

Note

You may apply IP or MAC access lists ingress on any Layer 2 port. You cannot apply them outbound on Layer 2 ports, only on SVI interfaces. In addition, the use of Layer 2 access-lists is limited to Layer 2 ports only.

Note one important difference. In the 3550 platform, MAC-based access-lists match only non-IP traffic such as ARP, STP, and IPv6. In the 3560 platform, they match only non-IP/IPv6 traffic such as ARP, STP, IPX and so on. You cannot filter IP traffic based on MAC address unless you use the port-security feature.

Be careful when applying MAC-based access-lists, you can filter management traffic such as STP BPDUs.

Copyright © 2010 Internetwork Expert

Verify that the access-list permits only ICMP traffic first:

```
Rack1R1#ping 155.1.146.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1R1#telnet 155.1.146.4
Trying 155.1.146.4 ...
% Connection timed out; remote host not responding
```

Now check that IPX traffic is blocked:

```
R1:
ipx routing
!
interface FastEthernet 0/0
 ipx network 146 encapsulation snap

R4:
ipx routing
!
interface FastEthernet 0/1
 ipx network 146 encapsulation snap

Rack1R1#ping 146.0015.c634.6c61

Translating "146.0015.c634.6c61"

Type escape sequence to abort.
Sending 5, 100-byte IPX Novell Echoes to 146.0015.c634.6c61, timeout is
2 seconds:
 ...
Success rate is 0 percent (0/5)
```

Remove the MAC access-list and check again that IPX traffic is now permitted:

```
Rack1SW1#interface FastEthernet 0/1
Rack1SW1(config-if)#no mac access-group ARP_ONLY in

Rack1R1#ping 146.0015.c634.6c61

Translating "146.0015.c634.6c61"

Type escape sequence to abort.
Sending 5, 100-byte IPX Novell Echoes to 146.0015.c634.6c61, timeout is
2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

Copyright © 2010 Internetwork Expert

11.28 Controlling Terminal Line Access

- Only allow Telnet connections to R2 from the Loopback0 subnets.
- Authenticate remote users locally using the name “TELNET” and the password of “CISCO”.
- Only allow this user to connect to R1 from R2.
- Log any attempt to connect from R2 to any destination on port 80.

Configuration

```
R2:
access-list 99 permit 150.1.0.0 0.0.255.255
!
access-list 100 permit ip any host 150.1.1.1
access-list 100 permit ip any host 155.1.146.1
access-list 100 permit ip any host 155.1.0.1
access-list 100 permit ip any host 155.1.13.1
access-list 100 deny tcp any any eq 80 log
!
username TELNET password CISCO
username TELNET access-class 100
!
line vty 0 4
 access-class 99 in
 login local
```

Verification

Note

You can apply standard or extended access-lists to any VTY line or username using the respective access-class. Based on the direction of the access-class command, it can control connections to the router (inbound), or from the router (outbound). When an access-list applies to a particular user, it controls the user’s outgoing connections from the router.

Using an extended access-list may be useful when you want to filter connections to a particular port. For example, when you want to deny connections to any rotary group port like 700X, or filter outgoing connections to port 80. When you use an extended access-list to match an incoming connection, use “any” as the destination IP address. The router will actually use the destination IP address 0.0.0.0 to match against the access-list, but using “any” is just a safeguard measure.

Copyright © 2010 Internetwork Expert

By specifying the **log** keyword, you may register packets matching a particular line in the access-list. This may be useful if you want to trace connections going to a particular port.

Try connecting to R2, sourcing the connection off the various interfaces:

```
Rack1R5#telnet 150.1.2.2
Trying 150.1.2.2 ...
% Connection refused by remote host

Rack1R5#telnet 150.1.2.2 /source Loopback0
Trying 150.1.2.2 ... Open


User Access Verification

Username: TELNET
Password: CISCO
```

Try connecting from R2 to different destinations and ensure that you can only connect to R1:

```
Rack1R2>telnet 150.1.3.3
Trying 150.1.3.3 ...
% Connections to that host not permitted from this terminal

Rack1R2>telnet 150.1.1.1
Trying 150.1.1.1 ... Open

User Access Verification

Password: cisco

Rack1R1>exit

[Connection to 150.1.1.1 closed by foreign host]
```

Now check that the router logs outgoing connections to port 80:

```
Rack1R2#terminal monitor
Rack1R2#telnet 150.1.3.3 80
Trying 150.1.3.3, 80 ...
% Connections to that host not permitted from this terminal
Rack1R2#
%SEC-6-IPACCESSLOGP: list 100 denied tcp 0.0.0.0(20034) ->
150.1.3.3(80), 1 packet
```

Copyright © 2010 Internetwork Expert

11.29 IOS Login Enhancements

- After 3 unsuccessful attempts to log into R3 in 30 seconds, block all further attempts for 40 seconds.
- Ensure that sessions sourced from the R5 Loopback0 are exempted from this restriction.
- Log every successful login attempt and every 3rd unsuccessful attempt.
- Enforce a delay of 2 seconds between successive login attempts.
- Create a local username "TEST" with the password of "TEST" for this task.

Configuration

```
R3:
username TEST password TEST
access-list 99 permit 150.1.5.5
!
login block-for 40 attempts 3 within 30
login quiet-mode access-class 99
login on-failure log every 3
login on-success log
login delay 2
!
line vty 0 4
 login local
```

Verification

Note

Login enhancements or IOS login block provides protection from brute-force dictionary attacks. This feature counts the number of failed login attempts during a time interval and prevents any further attempts for the duration of the block time interval. The command to set this behavior is:

login block-for <seconds> attempts <tries> within <interval>

This command will block any further login attempts for <*seconds*> if there were <*tries*> failed attempts during the <*interval*>. During this "quiet" interval, all login attempts are blocked with the exception of the hosts mentioned in the following access-list:

login quiet-mode access-class {acl-name|acl-number}

This is a special access-list which allows some important hosts to connect to the router even though it is blocking logins.

Copyright © 2010 Internetwork Expert

Additional commands for this feature include:

```
login on-failure log [every <login>]
login on-success log [every <login>]
login delay <seconds>
```

The first command enables logging every failed <*login*> attempt and the second command enables logging every unsuccessful <*login*> attempt. For example, if <*login*> is 3, then every 3rd attempt is logged. Without the parameter, these commands log every next attempt. The last command specifies the delay to enforce between every next login attempt to the router.

Note that these features are configured independent of any AAA settings. Additionally, banners and messages are also set separately from this feature.

Note that this feature does not work with line-based authentication, only with AAA or local database authentication.

Try logging in to R3 and fail authentication three times. Note that the router has logged the 3rd failed attempt.

```
Rack1R3#telnet 150.1.3.3
Trying 150.1.3.3 ... Open

User Access Verification

Username: TEST
Password:
% Login invalid

Username: TEST
Password:
% Login invalid

Username: TEST
Password:
% Login invalid

*Mar  6 12:23:49.920: %SEC_LOGIN-4-LOGIN_FAILED: Login failed [user:
TEST] [Source: 150.1.3.3] [localport: 23] [Reason: Login Authentication
Failed - BadPassword] at 12:23:49 UTC Wed Mar 6 2002
```

Copyright © 2010 Internetwork Expert

At the same time, the router will enter the quiet mode, blocking any further login attempts. After 40 seconds the quiet mode expires:

```
*Mar  6 12:23:49.920: %SEC_LOGIN-1-QUIET_MODE_ON: Still timeleft for
watching failures is 11 secs, [user: TEST] [Source: 150.1.3.3]
[localport: 23] [Reason: Login Authentication Failed - BadPassword]
[ACL: 99] at 12:23:49 UTC Wed Mar 6 2002
[Connection to 150.1.3.3 closed by foreign host]

Rack1R3#telnet 150.1.3.3
Trying 150.1.3.3 ...
% Connection refused by remote host

Rack1R3#
*Mar  6 12:24:29.922: %SEC_LOGIN-5-QUIET_MODE_OFF: Quiet Mode is OFF,
because block period timed out at 12:24:29 UTC Wed Mar 6 200
```

Now try logging in correctly and see if this attempt gets logged:

```
Rack1R3#telnet 150.1.3.3
Trying 150.1.3.3 ... Open


User Access Verification

Username: TEST
Password: TEST
Rack1R3>
*Mar  6 12:29:19.781: %SEC_LOGIN-5-LOGIN_SUCCESS: Login Success [user:
TEST] [Source: 150.1.3.3] [localport: 23] at 12:29:19 UTC Wed Mar 6
2002
```

Copyright © 2010 Internetwork Expert

Test the exemption for the quiet period. First try logging in three times incorrectly to R3 from R5:

```
Rack1R5#telnet 150.1.3.3
Trying 150.1.3.3 ... Open


User Access Verification

Username: TEST
Password:
% Login invalid

Username: TEST
Password:
% Login invalid

Username: TEST
Password:
% Login invalid

[Connection to 150.1.3.3 closed by foreign host]

Rack1R5#telnet 150.1.3.3
Trying 150.1.3.3 ...
% Connection refused by remote host

Rack1R5#telnet 150.1.3.3 /source Loopback0
Trying 150.1.3.3 ... Open


User Access Verification

Username: TEST
Password: TEST
Rack1R3>
```

While testing, note that the delay between each attempt is 2 seconds.

Copyright © 2010 Internetwork Expert

11.30 Role Based CLI

- Create roles in R4 named “SUPER”, “DEBUG”, “INTERFACE1” and “INTERFACE2”.
- The role “INTERFACE1” should have access to all IP configuration commands of interface Fa0/0 only. For “INTERFACE2” do the same with the exception that it applies to interface Fa0/1.
- The role “DEBUG” should be able to use any **debug** and **undebug** commands, and it should be able to inspect the running configuration.
- The last role name “SUPER” should be a sum of all the other roles.
- Use the password of “CISCO” to authenticate all roles.

Configuration

```
R4:
aaa new-model
!
exit
!
enable view
!
conf t
!
parser view DEBUG
 secret CISCO
 commands exec include show running-config
 commands exec include all debug
 commands exec include all undebug
!
parser view INTERFACE1
 secret CISCO
 commands interface include all ip
 commands configure include interface
 commands exec include configure terminal
 commands configure include interface FastEthernet0/0
!
parser view INTERFACE2
 secret CISCO
 commands interface include all ip
 commands configure include interface
 commands exec include configure terminal
 commands configure include interface FastEthernet0/1
!
parser view SUPER superview
 secret CISCO
 view DEBUG
 view INTERFACE1
 view INTERFACE2
```

Copyright © 2010 Internetwork Expert

Verification

Note

Role-based CLI enhances the local command authorization model. Instead of using hierarchical privilege levels, you may define user roles in Cisco IOS, called "views", and assign command sets accessible to each view. You may further group views in hierarchies using the concept of a super-view that contains other views instead of a list of the allowed commands.

Each view is password-protected, and users are required to enter this password when switching to the view. Alternatively, a view may be associated with a user utilizing the local database or special external AAA attribute, specifying the role name. Note that enabling AAA with **`aaa new-model`** is mandatory for the role-based access-control to work.

One special "root" view always exists. By default, even when in privileged exec mode, you are not part of the root view. You need to explicitly switch to the root view to be able to create other views. This is the only difference from the regular enable privilege mode. You can switch to the root view using the command **`enable view`**.

You can define a new view using the command parser **`view <VIEW_NAME>`**. Under the view configuration mode, you define the password for the view using the command **`secret <password>`** and define commands accessible to the view using the syntax similar to command-authorization:

`commands` *`parser-mode`* **`{include|exclude|include-exclusive} [all] [interface`** *`interface-name`***`] [command]`**

Here **`include-exclusive`** means the commands included in this view are not accessible to other views. By using the **`interface`** keyword you can limit the interfaces accessible to the role, provided that the user has access to the configure mode **`interface`** command. You can switch to the view using the command **`enable view <VIEW_NAME>`** in the router CLI.

If you need to define a superview, use the command **`parser view <NAME> superview`** and add views using the nested command **`view <SUBVIEW_NAME>`**. Note that all views, including superview, must have passwords defined in order to work.

Copyright © 2010 Internetwork Expert

To verify, try switching between the views in the router CLI. First try the view that has access only to one interface:

```
Rack1R4#enable view INTERFACE1
Password: CISCO

Rack1R4#
%PARSER-6-VIEW_SWITCH: successfully set to view 'INTERFACE1'.
Rack1R4#?
Exec commands:
  configure  Enter configuration mode
  enable     Turn on privileged commands
  exit       Exit from the EXEC
  show       Show running system information

Rack1R4#configure t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#?
Configure commands:
  do         To run exec commands in config mode
  exit       Exit from configure mode
  interface  Select an interface to configure

Rack1R4(config)#interface fastEthernet 0/0
Rack1R4(config-if)#?
Interface configuration commands:
  exit  Exit from interface configuration mode
  ip    Interface Internet Protocol config commands

Rack1R4(config-if)#interface FastEthetnet 0/1
                                  ^
% Invalid input detected at '^' marker.
```

Now try accessing the superview and check what commands are available:

```
Rack1R4#enable view SUPER
Password: CISCO

Rack1R4#?
Exec commands:
  configure  Enter configuration mode
  debug      Debugging functions (see also 'undebug')
  enable     Turn on privileged commands
  exit       Exit from the EXEC
  show       Show running system information
  undebug    Disable debugging functions (see also 'debug')
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#
%PARSER-6-VIEW_SWITCH: successfully set to view 'SUPER'.
Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface fastEthernet 0/0
Rack1R4(config-if)#?
Interface configuration commands:
  exit  Exit from interface configuration mode
  ip    Interface Internet Protocol config commands

Rack1R4(config-if)#interface fastEthernet 0/1
Rack1R4(config-if)#?
Interface configuration commands:
  exit  Exit from interface configuration mode
  ip    Interface Internet Protocol config commands

Rack1R4(config-if)#do sh run
Building configuration...

Current configuration : 171 bytes
!
!
!
!
!
!
interface FastEthernet0/0
 ip address 204.12.1.4 255.255.255.0
 ip rip advertise 10
!
interface FastEthernet0/1
 ip address 155.1.146.4 255.255.255.0
!
!
end

Rack1R4(config-if)#
```

11.31 IP Source Tracker

- You suspect that R4 is under a distributed DoS attack from unidentified sources.
- Configure R1 to collect attack information for the VLAN 146 IP address of R4.
- Export the statistics via syslog messages every 5 minutes.

Configuration

```
R1:
ip source-track syslog-interval 5
ip source-track 155.1.146.4
```

Verification

Note

IP source tracking offers a quick way to collect basic attack vector information for a given destination IP address. This is commonly needed in situations of distributed DoS attack where a single host is flooded by multiple sources. By enabling IP source tracking systematically on the path to the real source, you can backtrack every source of the attack to the network edge. There you can apply the actual filtering. The command to track a given destination IP address is:

`ip source-track <IP>`

The statistics collected include input interfaces and a packet count/packet rate. Based on this information you can track the next router closest to the attack source.

You may define the interval used by the feature to export the collected statistics as syslog entries using the command **`ip source-track syslog-interval`**.

Copyright © 2010 Internetwork Expert

To verify, generate some traffic from R6 towards the target IP address:

```
Rack1R3#ping 155.1.146.4 source serial 1/2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
Packet sent with a source address of 155.1.13.3
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms

Rack1R3#ping 155.1.146.4 source FastEthernet 0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
Packet sent with a source address of 155.1.37.3
!!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/4 ms

Rack1R3#ping 155.1.146.4 source Serial 1/3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
Packet sent with a source address of 155.1.23.3
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/4 ms

Rack1R3#ping 155.1.146.4 source lo 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
Packet sent with a source address of 150.1.3.3
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms
Rack1R4#

Rack1R1#show ip source-track
Address         SrcIF          Bytes    Pkts     Bytes/s     Pkts/s
155.1.146.4     Se0/1          1000      10          13          0
```

Note the aggregate statistics collected for the IP address, including the packet count, the total bytes switched, and the bytes per second rate.

Copyright © 2010 Internetwork Expert

11.32 Router IP Traffic Export

- Configure R1 to export the traffic sent and received on the Frame-Relay interface to the MAC address of R4.
- Configure a traffic filter that only allows TCP and ICMP packets to be monitored.

Configuration

```
R1:
ip access-list extended FILTER
 permit icmp any any
 permit tcp any any
!
ip traffic-export profile EXPORT
  interface FastEthernet0/0
  bidirectional
  incoming access-list FILTER
  outgoing access-list FILTER
  mac-address 0015.c634.6c61
 !
interface Serial 0/0
 ip traffic-export apply EXPORT
```

Verification

Note

Router IP traffic export (RITE) is a feature to monitor IP traffic received or sent on any WAN or LAN interface. This allows the monitoring of IP traffic on non-shared interfaces, such as Frame-Relay or ATM connections. Commonly this is needed for the purpose of IDS/IPS deployment or traffic monitoring.

One added benefit of using RITE is the ability to specify access-list filters for input or output traffic flows, allowing the router to select only the required traffic and minimize overhead.

Copyright © 2010 Internetwork Expert

RITE configuration consists of the following steps:

1) Defining an access-list to filter either inbound or outbound traffic or both.

2) Defining a RITE profile that specifies the exporting direction (inbound or bidirectional). By default the profile exports inbound traffic only.

3) Associate access-lists with inbound and outbound filters.

4) Define the export destination which is an output Ethernet interface and a MAC-address using the commands **`interface`** and **`mac-address`**.

RITE sends all IP packets unmodified and encapsulated into an Ethernet frame with the destination MAC address specified in the frame. Optionally, you may request the router to export every 1 of N packets in flows, in order to collect just statistical information and minimize the load on the router.

For this task, learn the MAC address of R4 using the command **`show ip arp 155.X.146.4`**.

You need a monitoring station with an Ethernet interface in promiscuous mode in order to capture the exported packets. From the router CLI, you can verify the configured traffic-export statistics.

Copyright © 2010 Internetwork Expert

Generate some ICMP traffic across R1 and check the statistics:

```
Rack1R6#ping 155.1.0.5 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 155.1.0.5, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max =
56/60/100 ms

Rack1R1#show ip traffic-export
Router IP Traffic Export Parameters
Monitored Interface             Serial0/0
        Export Interface                FastEthernet0/0
        Destination MAC address 0015.c634.6c61
        bi-directional traffic export is on
Output IP Traffic Export Information    Packets/Bytes Exported
100/10000
        Packets Dropped           0
        Sampling Rate             one-in-every 1 packets
        Access List         FILTER [named extended IP]
Input IP Traffic Export Information     Packets/Bytes Exported    0/0
        Packets Dropped           10
        Sampling Rate             one-in-every 1 packets
        Access List         FILTER [named extended IP]
        Profile EXPORT is Active
```

Note the increasing counter of exported packets. You may also check the increasing count of packets matching the filtering access-list:

```
Rack1R1#show ip access-lists FILTER
Extended IP access list FILTER
    10 permit icmp any any (200 matches)
    20 permit tcp any any
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.33 Controlling the ICMP Messages Rate

- In the future, you plan to apply an input traffic filter on R4's VLAN 146 interface.
- Configure this interface not to respond with an ICMP message when the filter drops a packet.
- The rate of these messages sent out of all other interfaces should not exceed 100 per second.
- In order to ensure that PMTUD works correctly, increase the rate of ICMP messages used by this feature to 1000 per second on R4.

Configuration

```
R4:
interface FastEthernet 0/1
 no ip unreachables
!
ip icmp rate-limit unreachable 10
ip icmp rate-limit unreachable DF 1
```

Verification

Note

Since routers operate at Layer 3, they can generate various ICMP messages when processing IP packets. Most commonly, routers respond with ICMP unreachable messages when dropping packets due to some reason. Most commonly the reasons are as follows:

1) Network/host/port unreachable - the router cannot route the packet to the destination.

2) Administratively Prohibited - the filter configuration on the router drops the packet.

3) Fragmentation required but DF-bit set - this message is critical in the PMTU discovery procedure.

There are two ways to control the generation of ICMP unreachable messages.

1) Disable generation of ICMP unreachables out of the specified interface. The interface-level command is **`no ip unreachables`**. You may want to enable this command in order to make it harder to map your network for an outside attacker by hiding the information of unreachable/filtered hosts or segments.

Copyright © 2010 Internetwork Expert

2) Control the rate of ICMP unreachable messages sent by the router. The global command is **ip icmp rate-limit unreachable <once per this ms>**. This command limits the total rate of all router-generated unreachable messages and prevents attacks designed to exhaust router resources.

3) Control the rate of "packet-too-big" messages (DF bit set but fragmentation required). Since this message type is crucial to PMTUD, you may want to set a separate rate-limit for it using the command **ip icmp rate-limit unreachable DF <once per this ms>.**

You can see if the unreachables are disabled on the interface using the following show command:

```
Rack1R4#show ip interface fastEthernet 0/1 | include unreach
  ICMP unreachables are never sent
```

Now try tracing the route to different R4 interfaces:

```
Rack1R1#traceroute 155.1.146.4 probe 5 timeout 1

Type escape sequence to abort.
Tracing the route to 155.1.146.4

  1  *  *  *  *  *
  2  *  *  *  *  *

Rack1R1#traceroute 155.1.0.4 probe 5 timeout 1

Type escape sequence to abort.
Tracing the route to 155.1.0.4

  1 155.1.0.5 32 msec 28 msec 28 msec 28 msec 32 msec
  2 155.1.0.4 69 msec 84 msec 96 msec 108 msec 116 msec
```

Traceroute sends UDP packets towards special UDP ports numbers. Since the router is the ultimate traceroute destination, it should respond with ICMP unreachables. In the first case, all probes timed out, since unreachables are disabled. In the second case, all probes are successful, since the router is configured to generate 100 messages per second.

Copyright © 2010 Internetwork Expert

11.34 Control Plane Policing

- Limit the aggregate rate of ARP traffic towards R1's route processor to 100 packets per second.
- The routing control traffic marked with an IP Precedence value of 6 should be limited to 50 packets per second.
- Drop all Telnet connections sourced from R5's Frame-Relay IP address.
- Limit the rate of outgoing ICMP messages to 10 per second.

Configuration

```
R1:
class-map ARP
 match protocol arp
!
ip access-list extended ICMP
 permit icmp any any
!
class-map ICMP
 match access-group name ICMP
!
ip access-list extended TELNET
 permit tcp host 155.1.0.5 any eq telnet
!
class-map TELNET
 match access-group name TELNET
!
class-map ROUTING
 match ip precedence 6
!
policy-map CPP_INPUT
 class ARP
  police rate 100 pps
 class TELNET
  drop
 class ROUTING
  police rate 50 pps
!
policy-map CPP_OUTPUT
 class ICMP
  police rate 10 pps
!
control-plane
 service-policy input CPP_INPUT
 service-policy output CPP_OUTPUT
```

Copyright © 2010 Internetwork Expert

Verification

Note

Control plane policing (CoPP) allows you to control the rate for traffic destined to or originated from the router process. This traffic includes all process-switched traffic, such as IP packets with options or packets logged by an access-list. Other types of traffic directed to the router process include routing updates, Telnet sessions, and any other traffic directed to the router itself.

The router may generate outgoing traffic, including ICMP responses, returning Telnet session traffic, outgoing IGP/BGP updates, outgoing Telnet sessions, etc.

In order to control this traffic, you need to define traffic classes using MQC syntax and apply a special control-plane policy-map. The only applicable policy-map actions are "drop" or "police". For classification, you may use an IP access-list with DSCP/IP precedence matching. Additionally, you can match protocol ARP directly in the class-maps. Do not try to use NBAR to classify the control plane traffic, for it may have unpredictable results.

For the "police" action, CoPP supports a unique packet-per-second rate-limiting feature. Using the command **police rate <N> pps** you can specify the aggregate rate for a class in packets per second. You can optionally specify the burst size (amount of packets received instantly) if you need to fine-tune your configuration. Note that this feature only works with CoPP policy-maps.

To verify the CoPP feature, configure R4 to send ICMP packets to R1.

```
Rack1R4#ping 155.1.146.1 repeat 1000 timeout 1

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 155.1.146.1, timeout is 1 seconds:
!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!
!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!
.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.!!.
Success rate is 66 percent (126/189), round-trip min/avg/max = 1/3/92
ms
```

Copyright © 2010 Internetwork Expert

In order to understand this behavior, look at the **show policy-map control-plane** command output at R1:

```
Rack1R1#show policy-map control-plane
 Control Plane

  Service-policy input: CPP_INPUT

    Class-map: ARP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol arp
      police:
          rate 100 pps, burst 24 packets
        conformed 0 packets; actions:
          transmit
        exceeded 0 packets; actions:
          drop
        conformed 0 pps, exceed 0 pps

    Class-map: TELNET (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name TELNET
      drop

    Class-map: ROUTING (match-all)
      832 packets, 96612 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: ip precedence 6
      police:
          rate 50 pps, burst 12 packets
        conformed 832 packets; actions:
          transmit
        exceeded 0 packets; actions:
          drop
        conformed 1 pps, exceed 0 pps

    Class-map: class-default (match-any)
      238 packets, 28914 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

```
Service-policy output: CPP_OUTPUT

    Class-map: ICMP (match-all)
      207 packets, 23598 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name ICMP
      police:
          rate 10 pps, burst 2 packets
        conformed 138 packets; actions:
          transmit
        exceeded 69 packets; actions:
          drop
        conformed 0 pps, exceed 0 pps

    Class-map: class-default (match-any)
      1013 packets, 119604 bytes
      5 minute offered rate 1000 bps, drop rate 0 bps
      Match: any
```

Note the default burst size of 10 pps – it is two packets. This is why R1 only accepts two packets in a row and drops the third packet. It takes one second for the token bucket to completely refill and then permit two more packets.

Now try ping and Telnet to R1 from R5's Frame-Relay IP address:

```
Rack1R5#telnet 155.1.0.1
Trying 155.1.0.1 ...
% Connection timed out; remote host not responding

Rack1R5#ping 155.1.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 72/199/392
ms
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#show policy-map control-plane
 Control Plane

  Service-policy input: CPP_INPUT

    Class-map: ARP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol arp
      police:
          rate 100 pps, burst 24 packets
        conformed 0 packets; actions:
          transmit
        exceeded 0 packets; actions:
          drop
        conformed 0 pps, exceed 0 pps

    Class-map: TELNET (match-all)
      4 packets, 192 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name TELNET
      drop

    Class-map: ROUTING (match-all)
      1256 packets, 146110 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: ip precedence 6
      police:
          rate 50 pps, burst 12 packets
        conformed 1256 packets; actions:
          transmit
        exceeded 0 packets; actions:
          drop
        conformed 1 pps, exceed 0 pps

    Class-map: class-default (match-any)
      410 packets, 47646 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Note the four packets dropped under the class-map for the Telnet traffic from R5. The ping operation was 100% successful because the packet rate across the Frame-Relay cloud is less than 2 packets per second.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

Note

Erase the running configurations of all devices and load the Security initial configurations.

11.35 Control Plane Protection (CPPr)

- Enhance R6 protection by dropping packets going to all closed ports.
- Ensure this does not affect connections made on ports 3020 and 4040.
- To reduce the effect of broadcast storms, limit the rate of input non-IP packets to 100 per second
- Set the queue-limit for input HTTP packets to 100 packets and limit the packet rate to 10 per second.
- Ensure that all fragmented packets transiting the router are limited to 1 Mpps.

Configuration

Note

Control Plane Protection (CPPr) is a Cisco IOS feature aimed at preventing infrastructure attacks, i.e. attacks targeting at the router itself. The Control Plane implements routing and management protocols, such as OSPF, BGP, RIP, SNMP, SSH, Telnet and so on. The most typical attacks against the control plane are designed to exhaust resources, i.e. they deplete a router's resources thus causing service denial. On most IOS platforms, control plane applications run on the central Route Processor (or CPU) in parallel with asynchronous packet switching. Packet routing is commonly implemented using a CEF switching path, during a hardware interrupt processing task. All packets directed to the control plane (e.g. routing updates, keepalives, SSH/SNMP sessions) are handled using process-switching, which is the most CPU intensive method of switching.

Since IOS 12.3T a feature known as Control Plane Policing allows for applying traffic policing for packets not handled during interrupt processing (fast/CEF paths). This feature provides basic protection against DoS attacks targeted at the router's CPU, such as fragmented ping flooding. A sample control plane policing configuration would look similar to the following example:

Copyright © 2010 Internetwork Expert

```
ip access-list extended BGP
 permit tcp any any eq bgp
!
ip access-list extended OSPF
 permit ospf any any
!
class-map BGP
 match access-group name BGP
!
class-map OSPF
 match access-group name OSPF
!
policy-map CONTROL_PLANE_POLICY
  class BGP
    police rate 2000 pps burst 100 packets
  class OSPF
    police rate 10 pps burst 3
  class class-default
    police rate 5000 pps burst 100 packets
!
control-plane
  service-policy input CONTROL_PLANE_POLICY
```

Notice that in the above configuration the police command specifies the rate in packets per second and the burst size in packets. This version of the police command is only applicable to the control-plane policy. In addition to the input policy, you can configure output policing as well, and limit the rate of the packets produced by the router's control plane.

Using just the demonstrated aggregate form of policing alone, you may already increase the level of your router's protections. However, CPPr extended CPP and made it even more granular. CPPr treats the Route Processor (RP) as a virtual interface attached to the router. All packets redirected to the RP are classified into three categories corresponding to the three subinterfaces of the virtual interface:

Copyright © 2010 Internetwork Expert

`Control-plane host subinterface`. This interface receives all control-plane TCP/UDP traffic that is directly destined for one of the router interfaces. Examples of control-plane host traffic include management traffic or routing protocols. Notice that non TCP/UDP control traffic will end up on the CEF-exception sub-interface. Most control plane protection features operate on this subinterface and thus this sub-interface provides most features, such as policing, port filtering and per-protocol queue thresholds.

Port-filtering allows for the automatic dropping of packets destined for TCP/UDP ports that are not currently open on the router. The operating system automatically detects all open ports, and you can manually configure some exceptions. The net effect is a reduced load on router's CPU during the times of flooding attacks.

Per-protocol queue thresholds allow you to set selective queue limits for packets of different types, for example BGP, OSPF, FTP, DNS, HTTP, IGMP, SNMP etc.

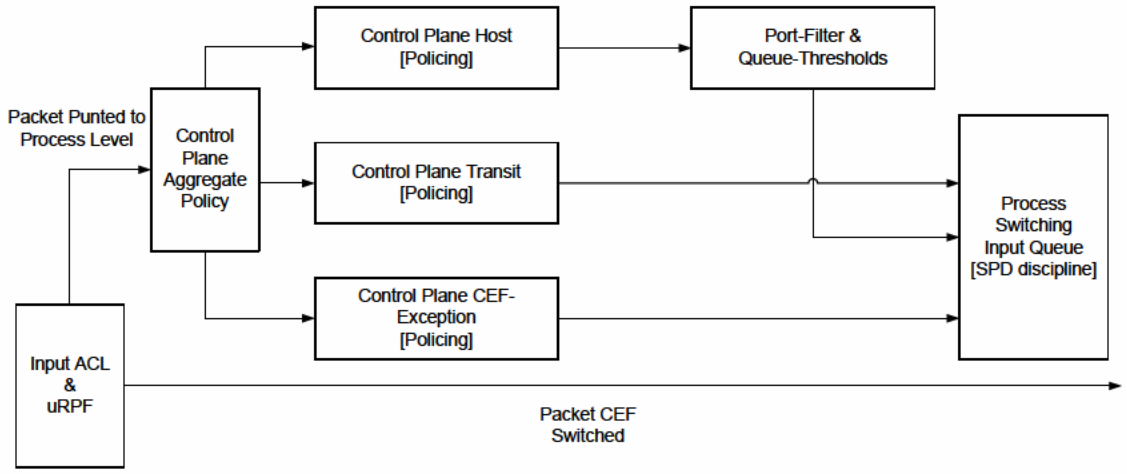
`Control-plane transit subinterface`. This subinterface is intended to handle *transit* IP packets not handled via the CEF mechanism that need to be switching in the processor context. This might happen, for instance, when a packet must be routed out an Ethernet interface and no ARP lookup has been made yet for the next-hop, making the CEF adjacency incomplete.

`Control-plane CEF exception subinterface`. This is where packets causing an exception in the CEF switching path land. So far, those include non-IP router-destined traffic, such as CDP, L2 keepalive messages (e.g. Frame-Relay LMI keepalives) or ARP packets and IP packets with options or TTL <=1. Additionally, non-UDP local multicast traffic destined to the router (e.g. OSPF updates) fall into this category as well.

You may apply a separate rate-limiting policy to any of the sub-interfaces or have a single aggregate policy embracing all subinterfaces (classic control plane policing). It is possible to configure both the subinterface and aggregate policy, but this feature appears to be unstable in many 12.4T images. So far, it seems to be better to configure either aggregate or subinterface specific policies.

Copyright © 2010 Internetwork Expert

Before any packet reaches any specific control plane subinterface, it is first processed via a series of ingress features including input access-list, uRPF checks, and aggregate control-plane policy. After passing the subinterface-specific policy, the packet is then enqueued onto the respective interface input queue and handled via the SPD (Selective Packet Discard) policy. Here is a visual outline of the process:



Let's look into the CPPr configuration steps. You start by defining class-maps to select control-plane traffic. Those are regular L3/L4 class-maps, matching access-lists/DSCP values etc. Please note that NBAR cannot be used for control-plane traffic classification. After the class-maps are defined, you define a regular policy map and assign classes to it, applying the desired actions. The policing command is the same as the one described above, based on packet rate per second. However, after this things start to differ. You may now attach the new policy map to any of the control plane sub-interfaces, like this:

```
policy-map CPP_HOST_POLICY
  class BGP
    police rate 256000
!
control-plane host
  service-policy input CPP_HOST_POLICY
```

Copyright © 2010 Internetwork Expert

You may use the command **control-plane {host| cef-exception| transit}** to select any of the control-plane subinterfaces. Notice that you cannot configure egress policing for any of the host subinterfaces. Now, the most important host subinterface also supports the port-filtering feature. To configure this one, you first need to create a class-map defining ports to be filtered. The command is **class-map type port-filter [match-all | match-any] <NAME>.** For example

```
class-map type port-filter match-any CLOSED_PORTS
 match closed-ports
 match port TCP 1024
 match port UDP 2048
```

The first **match** command auto-detects all open ports in the system and designates all other ports as "closed". To check the ports that are currently detected as being "open" use the command **show control-plane host open ports**. As you can see, you may add your own ports using the match port statement. You can also use the **match not port** command to unblock a port that was not automatically detected by the system as being open. For example, if you have a rotary-group 30 configured on a VTY line, you may want to make sure that TCP port 3030 is not blocked. You can use the following configuration:

```
class-map type port-filter match-all CLOSED_PORTS
  match closed-ports
  match not port 3020
```

To apply the port filtering settings, create a port-filtering policy-map and apply it to the host control-plane subinterface:

```
policy-map type port-filter FILTER_CLOSED_PORTS
  class CLOSED_PORTS
    drop
!
control-plane host
  service-policy type port-filter input FILTER_CLOSED_PORTS
```

This configuration will effectively drop all packets destined to closed ports before affecting router's CPU. The last feature to discuss is per-protocol input queue thresholds. As mentioned, these allow for differentiated per-protocol services on the control-plane host interface. In order to define a queue threshold, we must first create a special type of class class-map, called a queue-threshold class-map. This class-map should match one or many of the supported protocols:

Copyright © 2010 Internetwork Expert

```
class-map type queue-threshold [match-all | match-any]
<CLASS>
 match protocol [bgp | dns | ftp | http | igmp | snmp | ssh
| syslog | telnet | tftp]
```

As you can see, the list is pretty extensive. You can use the special command **match host-protocols** to select ALL TCP/UDP based protocols running on the router and then set a queue threshold for those. Packets are classified when they enter the router and are redirected to the host subinterface. After you have the special class-map defined you may configure a special queue-threshold policy-map and apply it to the host subinterface:

```
class-map type queue-threshold CMAP_BGP
  match protocol bgp
!
policy-map type queue-threshold BGP_THRESHOLD
  class CMAP_BGP
    queue-limit 50
!
control-plane host
 service-policy type queue-threshold input BGP_THRESHOLD
```

The queue limit is set in packets. This is the separate limit enforced for this particular protocol in the common input queue. There could be only one input queue-threshold policy-map, but you can configure every class with a separate limit.

Copyright © 2010 Internetwork Expert

R6:

```
!
! ACL to select fragmented IP packets
!
ip access-list extended FRAGMENTS
 permit ip any any fragment
!
class-map FRAGMENTS
 match access-group name FRAGMENTS

!
! HTTP traffic
!
ip access-list extended HTTP
 permit tcp any any eq www
!
class-map HTTP
 match access-group name HTTP

!
! Class-map to select closed ports. Notice that we
! need to unblock the RIP port explicitly.
!
class-map type port-filter match-all CLOSED_PORTS
 match closed-ports
 match not port TCP 3020
 match not port TCP 3040
 match not port UDP 520
!
policy-map type port-filter HOST_PORT_FILTER
 class CLOSED_PORTS
  drop

!
! To set the queue thresholds for HTTP
!
class-map type queue-threshold QUEUE_HTTP
 match protocol http
!
policy-map type queue-threshold HOST_QUEUE_THRESHOLD
  class QUEUE_HTTP
    queue-limit 100

!
! Control plane rate-limit policies
!
policy-map HOST_RATE_LIMIT
 class HTTP
  police rate 10 pps burst 2 packets

!
policy-map CEF_EXCEPTION_RATE_LIMIT
 class class-default
  police rate 100 pps burst 20 packets
```

Copyright © 2010 Internetwork Expert

```
!
policy-map TRANSIT_RATE_LIMIT
 class FRAGMENTS
  police rate 1000000 pps burst 200000 packets

!
! Applying the policies together
!
control-plane host
  service-policy input HOST_RATE_LIMIT
  service-policy type queue-threshold input HOST_QUEUE_THRESHOLD
  service-policy type port-filter input HOST_PORT_FILTER
!
control-plane transit
  service-policy input TRANSIT_RATE_LIMIT
!
control-plane cef-exception
 service-policy input CEF_EXCEPTION_RATE_LIMIT
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Configure an HTTP server in R3 and simulate an HTTP connection from R1 to R3.

```
Rack1R1#telnet 150.1.6.6 80
Trying 150.1.6.6, 80 ... Open
GET /
HTTP/1.1 400 Bad Request
Date: Fri, 07 May 2010 09:48:06 GMT
Server: cisco-IOS
Connection: close
Accept-Ranges: none

400 Bad Request

[Connection to 150.1.6.6 closed by foreign host]
Rack1R1#
```

> **Note**
>
> Check the policy-map statistics for the host subinterface.

```
Rack1R6#show policy-map control-plane host
 Control Plane Host

  Service-policy input: HOST_RATE_LIMIT

    Class-map: HTTP (match-all)
      12 packets, 720 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name HTTP
      police:
          rate 10 pps, burst 2 packets
        conformed 8 packets; actions:
          transmit
        exceeded 4 packets; actions:
          drop
        conformed 0 pps, exceed 0 pps

    Class-map: class-default (match-any)
      5 packets, 338 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

Note

Now check the port-filter policy map applied to the host subinterface. Configure R6 to listen on port 3020 by setting up a rotary group number on any VTY line. Notice that the list of auto-detected open ports does not include the ports manually added to the policy-map.

```
Rack1R6#show policy-map type port-filter control-plane host
 Control Plane Host

  Service-policy port-filter input: HOST_PORT_FILTER

    Class-map: CLOSED_PORTS (match-all)
      4 packets, 543 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:  closed-ports
      Match: not  port tcp 3020
      Match: not  port tcp 3040
      Match: not  port udp 520
      drop

    Class-map: class-default (match-any)
      12 packets, 816 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

R6:

```
line vty 0
 rotary 20
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#show control-plane host open-ports
Active internet connections (servers and established)
Prot        Local Address      Foreign Address                  Service
State
 tcp                *:23                 *:0                    Telnet
LISTEN
 tcp                *:80                 *:0                 HTTP CORE
LISTEN
 udp                *:67                 *:0             DHCPD Receive
LISTEN
 udp                *:68                 *:0              BootP client
LISTEN

Rack1R1#telnet 150.1.6.6 3020
Trying 150.1.6.6, 3020 ... Open


User Access Verification

Password:
```

> **Note**
>
> Check the queue thresholds for HTTP traffic class next. Notice that there are matches because we connected to the router using HTTP.

```
Rack1R6#show policy-map type queue-threshold control-plane host
      queue-limit 100
      queue-count 0     packets allowed/dropped 0/0
 Control Plane Host

  Service-policy queue-threshold input: HOST_QUEUE_THRESHOLD

    Class-map: QUEUE_HTTP (match-all)
      8 packets, 480 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:  protocol http

    Class-map: class-default (match-any)
      18 packets, 1224 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

> **Note**
>
> Next, check the CEF-exceptions subinterface statistics. There are a lot of packets matching this sub-interface as all non-IP keepalives, CDP packets, non-IP multicast hit this interface.

Copyright © 2010 Internetwork Expert

```
Rack1R6#show policy-map control-plane cef-exception
 Control Plane Cef-exception

  Service-policy input: CEF_EXCEPTION_RATE_LIMIT

    Class-map: class-default (match-any)
      537 packets, 38156 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      police:
          rate 100 pps, burst 20 packets
        conformed 537 packets; actions:
          transmit
        exceeded 0 packets; actions:
          drop
        conformed 1 pps, exceed 0 pps
```

Note

There is just one packet recorded for the transit control-plane subinterface. This packet was generated when we were pinging off R1 across R6 and the firewall attempted to resolve the IP to MAC mapping for BB1.

```
Rack1R6#show policy-map control-plane transit
 Control Plane Transit

  Service-policy input: TRANSIT_RATE_LIMIT

    Class-map: FRAGMENTS (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name FRAGMENTS
      police:
          rate 1000000 pps, burst 200000 packets
        conformed 0 packets; actions:
          transmit
        exceeded 0 packets; actions:
          drop
        conformed 0 pps, exceed 0 pps

    Class-map: class-default (match-any)
      1 packets, 114 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

11.36 IOS ACL Selective IP Option Drop

- Configure R3 to drop any packets destined to the router with IP source route option (either loose or strict).
- Apply this protection to the link connected to SW1.

Configuration

Note

IP options are special extensions to the standard IP header (which is 20 bytes in length) that signal special processing requirements for the datagram. For example, IP options may instruct the router to record the route in the IP header or specify a source-route at the origin. Packets with IP options are process-switched by Cisco IOS routers and put a significant load on the router CPU. Thus, it is important to filter packets with IP options unless they are really needed. You may configure the router to silently discard all packets with IP options using the command **ip options drop**. You may use access-lists to drop IP options selectively, using the ACL line syntax similar to the following:

```
permit ip any any option <Option Name>
```

You may find the list of IP options using the command-line interface help features ("?" sign). The special **any-option** keyword selects any option. The access-list may be used in either a policy-map or as an access-group. Notice that only named extended access-lists are supported for this feature.

R3:

```
ip access-list extended DROP_IP_SOURCE_ROUTE
 deny ip any any option lsr
 deny ip any any option ssr
 permit ip any any
!
interface FastEthernet 0/0
 ip access-group DROP_IP_SOURCE_ROUTE in
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> To verify IP option based filtering, generate ICMP packets with the Loose source route option from SW1. Notice the response packets from R3 – the ICMP unreachable messages have reason "Received Packet has Options".

```
Rack1SW1#ping
Protocol [ip]:
Target IP address: 155.1.37.3
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface:
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]: Loose
Source route: 155.1.37.3
Loose, Strict, Record, Timestamp, Verbose[LV]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.37.3, timeout is 2 seconds:
Packet has IP options:  Total option bytes= 7, padded length=8
 Loose source route: <*>
   (155.1.37.3)

Unreachable from 155.1.37.3.  Received packet has options
 Total option bytes= 7, padded length=8
 Loose source route: <*>
   (155.1.37.3)

<snip>
```

> **Note**
>
> Check the access-list statistics on R3 after this.

```
Rack1R3#show ip access-lists DROP_IP_SOURCE_ROUTE
Extended IP access list DROP_IP_SOURCE_ROUTE
    10 deny ip any any option lsr (5 matches)
    20 deny ip any any option ssr
    30 permit ip any any (164 matches)
```

Copyright © 2010 Internetwork Expert

11.37 BGP Generic TTL Security Mechanism

- Configure eBGP session between R5 and R6 and make both routers accept the peer if it's no more than one hop away.
- Use AS numbers 100 and 200 for R5 and R6 respectively.

Configuration

Note

General TTL Security Mechanism (GTSM) defined in RFC 3682 specifies a protection method against BGP session hijacking and resource exhaustion attacks. Generally, the BGP process listens on the TCP port 179 and accepts all TCP SYN packets destined to this port, unless they are filtered by an ACL. It is possible to generate a barrage of spoofed packets imitating a valid BGP session and inject false information (if the session is unauthenticated) or generate a TCP SYN-flooding attack.

GTSM utilizes the simple fact that every router on the path to the BGP speaker decrements the TTL field in IP packets by one. Based on this, it is possible to identify potentially spoofed packets by looking at their TTL field – the packets sent from "afar" will have the TTL field below some threshold. It is possible to define a "secure radius" in the number of hop counts to accept the incoming IP packets. For example, if all BGP peers are within 10 hops away from the local BGP speaker, then all incoming IP packets will have their TTL field set to at no less than 245. This is because all IP packets start with TTL=255 and the field is decremented by every hop on the path. Thus, by accepting the IP packets with TTL greater than or equal to 245 it is possible to minimize the risk of spoofed packets reaching the BGP process. Notice that the usefulness of GTSM feature decreases as the diameter of eBGP Multihop session grows.

Copyright © 2010 Internetwork Expert

In order to configure the TTL security checks for a BGP peer use the command **neighbor <IP> ttl-security hops <hop-count>**. This command applies to eBGP peering sessions only (either directly-connected or multihop) and specifies the number of hops the remote peer could be away from the local speaker. Keep in mind the internal BGP sessions are not protected, and therefore the internal network is assumed to be "trusted". All ncoming TCP packets targeted at the BGP port with the IP TTL value below (255 - <hop-count>) are silently discarded by the router. In addition, the feature sets the TTL value for outgoing TCP/IP packets to 255-<hop-count> to make sure the remote peer will accept the local packets. The GTSM feature is mutually exclusive with the **ebgp-multihop** BGP feature. This is because the eBGP session by default sets TTL=1 in the outgoing IP packets and with the **multihop <n>** session parameter, the TTL value is set to <n>, which is not compatible with GTSM Therefore, make sure you configured GTSM feature on both sides of the peering link.

R5:
```
router bgp 100
 neighbor 150.1.6.6 remote-as 200
 neighbor 150.1.6.6 ttl-security hops 2
 neighbor 150.1.6.6 update-source loopback 0
```

R6:
```
router bgp 200
 neighbor 150.1.5.5 remote-as 100
 neighbor 150.1.5.5 ttl-security hops 2
 neighbor 150.1.5.5 update-source loopback 0
```

Verification

Note

Check the TTL settings for every peer. Notice the minimum incoming TTL of 253 in the peer properties.

```
Rack1R5#show ip bgp neighbors 150.1.6.6 | inc TTL
Connection is ECN Disabled, Mininum incoming TTL 253, Outgoing TTL 255

Rack1R6#show ip bgp neighbors 150.1.5.5 | include TTL
Connection is ECN Disabled, Mininum incoming TTL 253, Outgoing TTL 255
```

Copyright © 2010 Internetwork Expert

11.38 Flexible Packet Matching

- Configure R4 to filter ICMP Echo packets with the string “AAA” in the payload. Look no deeper than 256 bytes in the packet.
- Ensure the filtering applies only to ICMP/IP packets received in Ethernet frames.

Configuration

Note

Flexible Packet Matching (FPM) is a new feature that allows for granular packet inspection in Cisco IOS routers. Using FPM you can match any string, byte or even bit at any position in an IP (or theoretically non-IP) packet. This may greatly aid in identifying and blocking network attacks using static patterns found in the attack traffic. This feature has some limitations though.

a) First, it is completely stateless, meaning that it does not track the state/history of the packet flow. Thus, FPM cannot discover dynamic protocol ports such as those used by H.323 or FTP nor can it detect patterns split across multiple packets. Essentially, you are allowed to apply inspection on a per-packet basis only.

b) Additionally, you cannot apply FPM to the control-plane traffic, as the feature is implemented purely in the CEF switching layer. Fragmented traffic is not assembled for matching, and the only inspected packet is the initial fragment of a given IP packet flow.

c) IP packets with IP options are not matched by FPM either because they are punted to the route processor.

d) Lastly, this feature inspects only unicast packets and does not apply to MPLS encapsulated packets.

Configuring an FPM filter consists of four steps.

(1) Loading protocol headers.

(2) Defining a protocol stack.

(3) Defining a traffic filter.

(4) Applying the policy & Verification

Let’s look at every one of these steps in detail.

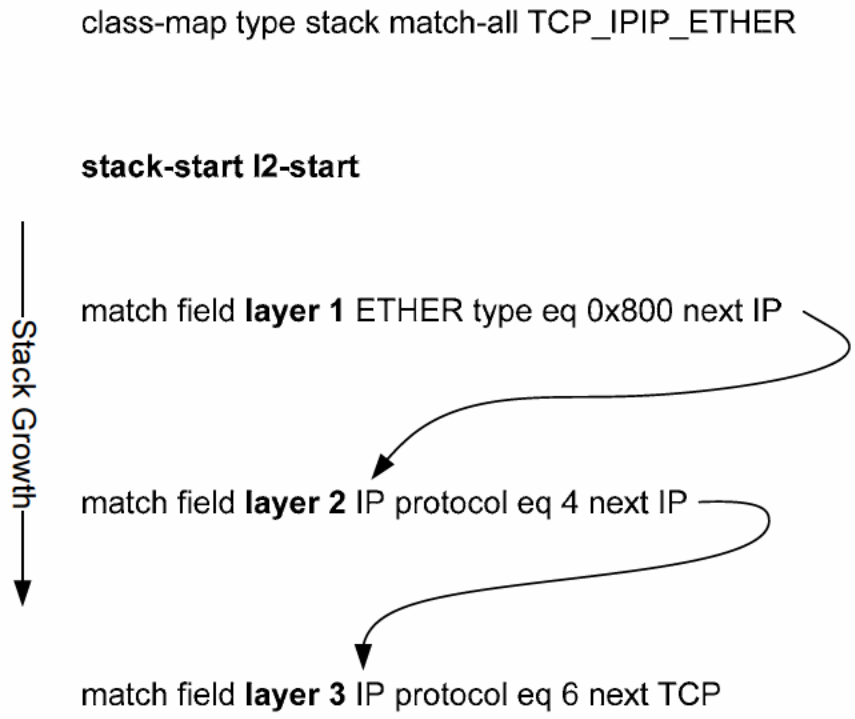
Copyright © 2010 Internetwork Expert

Load a PHDF (optional). PHDF stands for Packet Header Definition File. Those files use XML syntax and define the structure of various packet headers, such as Ethernet, IP, TCP, UDP and so on. They are very helpful in making filtering more structured, as with the PHDFs loaded you may filter based on the header field names and their values, instead of matching fixed offsets in the unstructured packet body. You may load the files into the router's memory using the command **load protocol <path>**. The PDHF files could be manually created using the simple XML syntax or downloaded from CCO. Since IOS version 12.4(15)T, four basic PHDFs are included in the IOS code and are located at the virtual path **system:fpm/phdfs**. You could load the files directly from there. Defining a custom PHDF requires an understanding of protocol header formats and field values along with basic XML formatting, which is beyond the scope of this document.

Define a protocol stack (optional). This step uses the PDHFs loaded previously and allows specifying the protocol headers found in the traffic you want to inspect. Using the protocol stack definition induces structure in the packets being inspected. This allows for filtering based on header field values and specifying offsets in the packet relative to the header fields. Additionally, you may define various protocol stacks (e.g. UDP in IP, UDP in GRE in IP) and reuse the same access control policy with various stacks.

You define the protocol stack using the command **class-map type-stack match-all <NAME>**. This class-map is always of type match-all and consists of a series of match entries. Every match entry should specify a protocol name defined in the loaded PHDFs, a header field value and the next protocol name found in stack. Look at the sample below – it defines the series of headers: TCP in IPIP tunnel headed (protocol 4) encapsulated in IP and in the Ethernet.

Copyright © 2010 Internetwork Expert



The order of the match statements is important – it defines the actual header stacking along with the "layer x" keywords. The layers are counted bottom up starting at the stack bottom. The other important thing is that every next **`match`** statement should define the protocol specified in the **`next`** clause of the previous match statement. This is a basic consistency check illustrated using the arrows in the diagram.

A few words about the **`match`** operator. This command uses the syntax **`match field <PROTO> <header-field> <operator> <value>.`** It is important that the protocol and header fields are learned dynamically from PHDFS and this allows for high flexibility. The operator could be "eq", "neq", "range", "string", "regex", "lt" and "gt". The operator names are self-explanatory, but the "eq" operator also supports a special "mask" parameter. The mask is interpreted as a bitmap, where "1" means the corresponding bit in the value could be any of "0" or "1" and "0" means the respective bit is "locked".

Copyright © 2010 Internetwork Expert

By default, the protocol stack is matched starting AFTER the data-link level header. That is, the first header defined in the stack is matched against the header going after the L2 header. In real life, this is commonly the IP header. However, sometime you may want to match the layer2 header fields as well, e.g. Ethernet addresses. To make the protocol stack match starting at L2 level, use the command **stack-start l2-start**.

At this point, having just stack class-maps you may already apply traffic filtering to the packets matching the configured stack. To accomplish this, create a policy-map of type "access-control" and assign a stack class to the policy-map. For example:

```
class-map type stack TCP_IN_IP_IN_ETHER
 statck-start l2-start
 match field ETHER type eq 0x800 next IP
 match field layer 2 IP protocol eq 0x6 next TCP
!
policy-map type access-control DROP_TCP_IN_IP_IN_ETHER
 class TCP_IN_IP_IN_ETHER
   drop
```

There is basically just one action available with the access-control policy-maps, and the action is drop. You may also use the send-response command under a class to make the router send an ICMP unreachable response. However, using just stack class-maps might be inflexible, as you cannot match the packets payload. You may however use this for filtering based on addresses, protocol flags and so on.

Define a traffic filter. Traffic filter is defined by means of a special class-map of the type "access-control" and configuring a respective policy-map of the same type. Using these types of class-maps you can match the protocol field values using the command **match field <PROTO>** if the respective protocol's PHDF has been loaded. If you want to match the protocol fields, then the policy-map using the newly defined access-control class-map must be nested under the stack-type class-map defining this protocol. We'll see an example later.

In addition to matching the protocol header fields, you can match the packet payload at a fixed offset against a fixed value, value range, string or regular expression. You may base the offset off the define protocol header field (e.g. +10 bytes from TCP flags) using the command **match start <PROTO> <FIELD-NAME> offset <OFFSET> size <SIZE>**. For example:

Copyright © 2010 Internetwork Expert

```
match start TCP checksum offset 100 size 1
match start IP version offset 0 size 1
```

Of course, this type of offset is only possible if the respective protocol definition has been loaded and the containing stack-type class map has this protocol defined.

Irrespective of the protocols loaded/defined, you may base the offset from the L2 or L3 packet starts (absolute offsets). For example if the packet is IP in Ethernet, then L2-start is the first bit of the Ethernet header and L3-start is the first bit of the IP header. The command syntax is as follows:

```
match start {l2-start|l3-start} offset <OFFSET-BYTES> size
<SIZE-BYTES> <operator> <value>
```

This command specifies the offset in bytes from the selected start. If the size is less than or equal to 4 bytes, you can use the “eq, neq, lt, gt” operators in addition to “regex” and “string”. This allows for per-bit matching using the “eq” and “mask” operators combination. For example:

```
match start l3-start offset 0 size 1 eq 0x2 mask 0x3
match start l2-start offset 36 size 5 string ABCDE
```

Create an access-control policy-map. There are two options here. You may create a simple access-control policy-map, which has just the basic access-control assigned, without nesting. For example:

```
class-map type access-control PASSWORD
 match start l3-start offset 0 size 100 regex
".*[pP][aA][sS][wW].*"
!
policy-map type access-control DROP_PASSWORD
 class PASSWORD
  drop
```

Copyright © 2010 Internetwork Expert

When using this simple non-nested syntax, you may only use the absolute offsets with the command **match {l2-start|l3-start}** and cannot reference any protocol header fields.

You may use a more flexible approach, by nesting the filtering policy under a stack-type class-map configured in containing policy-map. This allows for using protocol headers in the filtering policy or basing the offsets from the packet header fields. Notice that the nested filtering policy may only use the protocol headers defined in the containing stack class-map. Here is an example that looks for string "TEST" in TCP packets:

```
class-map type stack TCP_IN_IP
 match field IP protocol eq 0x6 next TCP
!
! Protocol fields matched in the filtering policy
! must be for the protocols defined in the stack
! class-map
!
class-map type access-control match-any FILTER_CLASS
 match start TCP payload offset 0 size 4 string TEST
!
policy-map type access-control FILTER_POLICY
 class FILTER_CLASS
  drop
!
policy-map type access-control STACK_POLICY
  class TCP_IN_IP
  service-policy FILTER_POLICY
```

Apply the traffic filtering policy. Finally, the access-control policy map should be applied to an interface either inbound or outbound using the interface-level command **service-policy type access-control {input|output} <NAME>**. This could be a simple non-nested policy using just stack-type class-maps or access-control class-maps or more advanced, stack and nested access-control policy-map.

Copyright © 2010 Internetwork Expert

R4:

```
!
! Load the protocol header definition
!
load protocol system:fpm/phdf/ip.phdf
load protocol system:fpm/phdf/icmp.phdf
load protocol system:fpm/phdf/tcp.phdf
load protocol system:fpm/phdf/udp.phdf
load protocol system:fpm/phdf/ether.phdf

!
! Define a protocol stack
!
class-map type stack ICMP_IN_IP_IN_ETHER
 stack-start l2-start
 match field ether type eq 0x800 next ip
 match field layer 2 ip protocol eq 1 next icmp

!
! Define access-control policy-map
!
class-map type access-control match-all ICMP_ECHO_STRING
 match field icmp type eq 8
 match start icmp payload offset 0 size 256 regex ".*AAAA.*"
!
policy-map type access ACCESS_CONTROL_POLICY
 class ICMP_ECHO_STRING
  log
!
policy-map type access-control STACK_POLICY
 class ICMP_IN_IP_IN_ETHER
   service-policy ACCESS_CONTROL_POLICY
!
interface FastEthernet 0/1
 service-policy type access-control input STACK_POLICY
```

Copyright © 2010 Internetwork Expert

Verification

Note

Generate ICMP packets with the payload “AAAA” from R6 to R1. To accomplish this, use any ASCII codes table to find that the ASCII code for “A” is 0x41. After this, issue the following command from R6:

```
Rack1R6#ping 150.1.4.4 source loopback 0 data 4141

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
Packet sent with a source address of 150.1.6.6
Packet has data pattern 0x4141
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/4 ms
```

Note

Now check the access-control policy-map statistics on R4. Notice that both the stack-map and the access-control map have been matched. Additionally, you should get the log message that appears after the debugging output:

Copyright © 2010 Internetwork Expert

```
Rack1R4#show policy-map type access-control interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy access-control input: STACK_POLICY

    Class-map: ICMP_IN_IP_IN_ETHER (match-all)
      5 packets, 570 bytes
      5 minute offered rate 0 bps
      Match: field ETHER type eq 0x800 next IP
      Match: field IP protocol eq 1 next ICMP

      Service-policy access-control : ACCESS_CONTROL_POLICY

        Class-map: ICMP_ECHO_STRING (match-all)
          5 packets, 570 bytes
          5 minute offered rate 0 bps
          Match: field ICMP type eq 8
          Match: start ICMP payload-start offset 0 size 256 regex
".*AAAA.*"
      log

        Class-map: class-default (match-any)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: any

    Class-map: class-default (match-any)
      1 packets, 86 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

%SEC-6-IPACCESSLOGDP: list ICMP_ECHO_STRING permitted icmp 150.1.6.6
(FastEthernet0/1 ) -> 150.1.4.4 (8/0), 5 packets
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.39 Zone Based Firewall

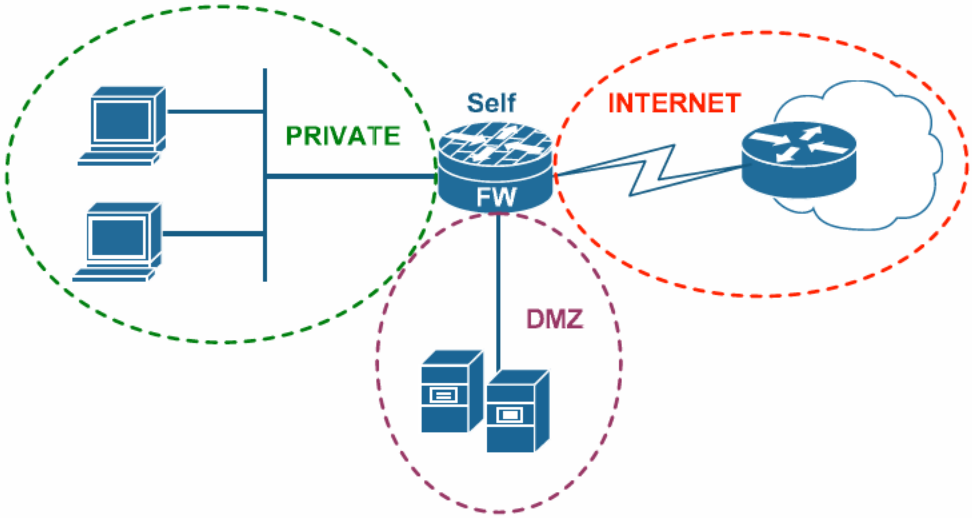
- Configure three security zones on R5: INSIDE, OUTSIDE and DMZ for the interfaces facing R4, SW2 and SW3 respectively.
- Create interface VLAN 5 on SW3 and assign the address 155.1.5.9/24.
- Only allow R5 to be accessed using SSH and HTTPs from zones OUTSIDE and DMZ.
- Allow the INSIDE users to use the following protocols when accessing OUTSIDE: HTTP, FTP, ICMP, DNS, SSH, Telnet, AOL Instant Messenger.
- Account for the AOL Messenger using the non-standard port 80 and users connecting to HTTP proxies on ports 3128 and 8080.
- Allow the OUTSIDE users to access the servers in DMZ using the HTTP, FTP, DNS and TACACS+ protocols.
- The INSIDE users should have the same set of protocols allowed to zone DMZ with the addition of SSH and HTTPS.
- Preserve IP routing with your configuration and allow the users logged into the router to ping/telnet to OUTSIDE and DMZ zones.
- Limit the inside users to subnet 150.X.1.0/24 and the DMZ server to the subnet 155.X.37.0/24.

Configuration

Note

Zone based Firewall of ZFW is a new approach to configuring access control in the IOS firewall. Prior to this feature, traffic filtering was accomplished using access-lists and stateful traffic inspection rules (CBAC). Both the access-lists and inspection rules apply directly to the physical interfaces, which may poorly reflect organization's security policy, as the policy deals with more high-level objects than interfaces. A new core concept of ZFW is zone, which groups different interfaces sharing the same security attributes, the same level of trust. For example, you may have security zones that reflect your enterprise security levels partitioning. On the figure below, you can see three security zones assigned to three router interfaces:

Copyright © 2010 Internetwork Expert



Permissions for traffic forwarding is made between the zones, not physical interfaces. By default, traffic is permitted between the interfaces within the same security zone, and blocked between different zones. Traffic between the interface configured in a security zone and interface not in any zone is blocked. In addition, you cannot apply classic firewall rules (e.g. access-lists or inspection rules) to the interface configured in a security zone.

There is one default zone in every router, known as **`self`**, which encompasses the router's own IP addresses. Traffic to and from this zone to any other zone is *permitted* by default, to allow for control plane and management plane traffic. However, you may apply an explicit policy between the **`self`** zone and any other configured zone to control the router-originated traffic. If you apply a policy from any configured zone to the **`self`** zone, the traffic from **`self`** zone to the other zone will be permitted, but the returning traffic from the configured zone may be blocked. To resolve this, you may need to assign a policy from zone self to the configured zone and inspect the router-generated traffic.

In order to permit certain traffic flows between two different security zones, you need to define a zone pair and apply an inspection policy to this pair. Zone pairs are ordered, i.e. {A, B} is not the same as {B, A}. The first zone in the pair is source zone and the other one is destination zone. When you assign the inspection policy to a zone pair, it applies to the traffic flowing from source to destination zone. The inspection policy is configured as a set of traffic-classes along with associated actions, such as inspect, pass or drop. Traffic classes are generally defined using access-lists and/or protocol matching.

Copyright © 2010 Internetwork Expert

Zone based firewall implements the same set of features as the classic IOS firewall. It allows defining advanced inspection options, such as deep protocol inspection and parameter tuning. To accomplish all listed features, ZFW uses a new configuration framework called Cisco Policy Language (CPL). The syntax for the new configuration language was heavily borrowed from the well-known MQC (Modular QoS CLI) and resembles it in many ways, by using policy-maps, class-maps and actions association. Here is a brief overview of ZFW configuration steps:

1) Define zones. After initial planning, you define security regions and come with the zone names. The command to accomplish this is **zone security <NAME>** and every zone may have a meaningful description assigned to it.

2) Define zone-pairs. Zone pairs should be defined prior to policy configuration. As mentioned before, zone pairs are ordered sets and specify the direction of traffic flow. The command to define a zone pair is: **zone-pair security <NAME> source <A> destination <B>**, e.g. **zone-pair security SELF_TO_OUTSIDE source self destination OUTSIDE**.

3) Define class-maps that describe traffic that must have policy applied as it crosses a zone-pair. There is a new special type of class-maps used at this step – **class-map type inspect**, which specifies inspection scope.

4) Define policy-maps to apply action to your class-maps' traffic. A new special type of policy-map called **policy-map type inspect** has been introduced by CPL to specify inter-zone policies. You assign the previously defined class-maps to the policy-map and specify associated actions.

5) Define other advanced policy attributes. Advanced options include inspection parameter tuning, TCP/UDP session limiting and application inspection rules. We'll discuss that in separate scenario.

6) Apply policy-maps to zone-pairs. The command to apply the security policy to a zone pair is the same as to define the pair. For example:

```
zone-pair security INSIDE_TO_SELF source INSIDE destination
self
  service-policy type inspect PMAP_INSIDE_TO_SELF
```

Notice that policy applies just in one direction, and in some cases you may need to apply a policy in the opposite direction as well.

Copyright © 2010 Internetwork Expert

7) Assign interfaces to zones. This is last step, accomplished using the interface command **zone-member security <ZONE>**.

Defining Traffic Classes.

Generic inspection type class-maps could be of type **match-all** or **match-any**, just like the regular MQC class-maps. In the first case, all match conditions must be met in order for the class-map to match the traffic. You can match an access-list using the command **match access-group [number] [name <NAME>]**, which is the only way to select traffic sources or destination.

In addition to matching access-lists, you can match protocols supported by the inspection engine. The list of the protocols is the same as supported by CBAC. Unlike the classic class-map, when you configure the **match protocol** command, you do not engage NBAR engine – rather, the protocol will be selected for inspection once the containing policy-map will be assigned to a zone pair. It is common to combine access-list matching with protocol matching, for example:

```
class-map match-all CMAP_HTTP_INSIDE_USERS
 match access-group name ACL_INSIDE_USERS
 match protocol http
```

would match HTTP traffic for inside users. You can use the command **show ip port-map** to list all supported protocols available for matching.

You may apply class-map nesting using the command **match class-map**. This may be vefy helpful in situations where you need to apply complex AND/OR logic. For example, if you want to select a set of protocols used (e.g. HTTP, DNS, FTP, ICMP) used by a group of users. You may not combine multiple protocols match and ACL matching within a single class-map, but you may use class-map nesting as follows:

```
class-map match-any CMAP_INSIDE_PROTOCOLS
 match protocol http
 match protocol ftp
 match protocol dns
 match protocol icmp
```

Copyright © 2010 Internetwork Expert

```
!
ip access-list extended ACL_INSIDE_USERS
 perm ip 192.168.0.0 0.0.0.255 any
!
class-map match all CMAP_INSIDE_USERS
 match class-map CMAP_INSIDE_PROTOCOLS
 match access-group name INSIDE_USERS
```

Applying Policy Actions.

There are three general policy actions applicable under inspect-type policy-maps: "inspect", "drop" and "pass". The first action is similar to the CBAC inspection rule. It allows for stateful inspection of traffic flowing from source to destination zone, and automatically permits returning traffic flows even for complex protocols, such as H.323. The "drop" action simple discards matching traffic, and the "pass" action permits packet flow without stateful inspection, similar to "permit" action in access-lists. Here is an example:

```
policy-map PMAP_OUTSIDE_TO_DMZ
 class-map CMAP_LEGIT_TRAFFIC
    inspect
```

Notice that if you are using the "inspect" action, then the class-map must hav at least once "match protocol" statement, to specify the protocols for inspection. Otherwise, all protocols will be inspected. If you are using the "pass" action, make sure the returning traffic is also permitted, as the pinholes will not be opened dynamically.

Every inspection policy-map has an implicit class-default configured with action "drop". This simulates the implicit "deny any" action found in traditional access-lists.

Port-Mapping.

As mentioned previously, ZFW supports the same protocol set as CBAC. You may see the port-numbers used by various applications by issuing the command **show ip port-map**. If you need to define a custom protocol port, you may use the command **ip port-map <protocol>** to assign new port numbers to the protocol. If you need to re-use the well-know port for a different application, e.g. assign telnet application to port 80, you need to use an access-list using the command **ip port-map telnet port <PORT> list <ACL>.** These are the same requirement as with the classic CBAC firewall. For example:

Copyright © 2010 Internetwork Expert

```
access-list 99 permit 192.168.1.0 0.0.0.255
!
ip port-map telnet port tcp 80 list 99
```

Self-Zone Considerations.

Self-zone policy has limited functionality as compared to the policies available for transit-traffic zone-pairs. First, for the stateful inspection, router-generated traffic is limited to TCP, UDP, ICMP, and complex-protocol inspection for H.323. Application Inspection (HTTP, FTP, Telnet) is not available for self-zone policies, i.e. you cannot apply any advanced protocol filtering for traffic destined to/from the firewall. Additionally, session and rate limiting cannot be configured on self-zone policies.

You may want to configure stateful inspection for protocol terminated/originated at the router if you apply a policy between the zone self and any other configured zone. The policy-map by default drops all non-matched traffic, and this may cause issues. For example, consider that you configured to pass only ICMP and SSH traffic to the firewall using the following configuration

```
ip access-list extended ACL_OUTSIDE_TO_SELF
 permit tcp any any eq 22
 permit icmp any any
!
class-map type inspect match-any CMAP_OUTSIDE_TO_SELF
 match access-group name ACL_OUTSIDE_TO_SELF
!
policy-map type inspect PMAP_OUTSIDE_TO_SELF
 class CMAP_OUTSIDE_TO_SELF
   pass
!
zone-pair security ZP_OUTSIDE_TO_SELF source OUTSIDE
destination self
 service-policy type inspect PMAP_OUTSIDE_TO_SELF
```

This will effectively block any ICMP/TCP/UDP sessions originated off the router, by the virtue of implicit class-default found in the new policy-map. In order to resolve this issue, you may configure the following inspection (remember that only UDP/TCP/ICMP/H.323 are supported, everything else should be permitted explicitly).

Copyright © 2010 Internetwork Expert

```
class-map type inspect match-any  CMAP_ROUTER_PROTOCOLS
 match protocol icmp
 match protocol udp
 match protocol tcp
!
policy-map type inspect PMAP_SELF_TO_OUTSIDE
 class CMAP_ROUTER_PROTOCOLS
  inspect
!
zone-pair security ZP_SELF_TO_OUTSIDE source self
destination OUTSIDE
 service-policy type inspect PMAP_SELF_TO_OUTSIDE
```

If you are using OSPF for routing and have a policy to control traffic from outside to self, you may need to configure something like the following:

```
ip access-list extended OSPF
 permit ospf any any
!
class-map type inspect CMAP_OSPF
 match access-group name OSPF
!
policy-map type inspect PMAP_OUTSIE_TO_SELF
 class CMAP_OSPF
   pass
```

If you want to have at least some inspection for protocols not supported with the zone self, you may still use TCP/UDP inspection. For example, if you want added security with HTTPS and SSH you may use the following configuration to apply basic traffic inspection:

```
ip access-list extended ACL_SSH_HTTPS
 permit tcp any any eq 22
 permit tcp any any eq 443
!
class-map type inspect match-all CMAP_SSH_HTTPS
 match access-group name ACL_SSH_HTTPS
 match protocol tcp
!
policy-map type inspect PMAP_OUTSIDE_TO_SELF
 class CMAP_SSH_HTTPS
   inspect
```

Copyright © 2010 Internetwork Expert

Note the above configuration is only really needed if you have a policy-map applied to the {self, OUTSIDE} zone pair and thus may block traffic returning from the router.

Finally, here is a short lists of the ZFW configuration steps in the recommended order.

1) Define access-lists for traffic scopes
2) Define class-maps
3) Define policy-maps
4) Configure zones and zone-pairs, apply the policies
5) Assign zones to the interfaces

Try using meaningful names for all object, and use prefixes ACL_, CMAP_, PMAP_ to allow you easily distinguishing object's purpose.

Copyright © 2010 Internetwork Expert

R5:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Access-Lists
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!
! Traffic sourced FROM INSIDE IPs
!
ip access-list standard ACL_INSIDE_USERS
 permit 150.30.4.0 0.0.0.255

!
! Traffic TO DMZ IPs
!
ip access-list extended ACL_DMZ_HOSTS
 permit ip any 155.30.5.0 0.0.0.255

!
! Telnet sessions (for zone self inspection)
!
ip access-list extended ACL_TELNET
 permit tcp any any eq 23

!
! SSH and HTTPs connections
!
ip access-list extended ACL_SSH_HTTPS
 permit tcp any any eq 22
 permit tcp any any eq 443

!
! RIP updates
!
ip access-list extended ACL_RIP
 permit udp any any eq 520

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Class-maps of type inspect
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!
! RIP updates
!
class-map type inspect match-all CMAP_RIP
 match access-group name ACL_RIP

!
! HTTPs and SSH traffic
!
class-map type inspect match-all CMAP_HTTPS_SSH
 match access-group name ACL_SSH_HTTPS
 match protocol tcp

!
! Telnet Traffic
!
class-map type inspect match-all CMAP_TELNET
```

Copyright © 2010 Internetwork Expert

```
 match access-group name ACL_TELNET
 match protocol tcp

!
! ICMP Traffic
!
class-map type inspect match-all CMAP_ICMP
 match protocol icmp

!
! Protocols allowed from INSIDE to OUTSIDE
!
class-map type inspect match-any CMAP_INSIDE_TO_OUTSIDE_PROTOCOLS
 match protocol http
 match protocol ftp
 match protocol icmp
 match protocol dns
 match protocol ssh
 match protocol telnet
 match protocol aol

!
! Protocols allowed from OUTSIDE to DMZ
!
class-map type inspect match-any CMAP_OUTSIDE_TO_DMZ_PROTOCOLS
 match protocol http
 match protocol ftp
 match protocol dns
 match protocol tacacs

!
! Protocols allowed from INSIDE to DMZ
!
class-map type inspect match-any CMAP_INSIDE_TO_DMZ_PROTOCOLS
 match protocol http
 match protocol ftp
 match protocol dns
 match protocol tacacs
 match protocol ssh
 match protocol https

!
! Traffic from INSIDE to OUTSIDE
!
class-map type inspect match-all CMAP_INSIDE_TO_OUTSIDE_ACCESS
 match access-group name ACL_INSIDE_USERS
 match class-map CMAP_INSIDE_TO_OUTSIDE_PROTOCOLS

!
! Traffic from OUTSIDE to DMZ
!
class-map type inspect match-all CMAP_OUTSIDE_TO_DMZ_ACCESS
 match class-map CMAP_OUTSIDE_TO_DMZ_PROTOCOLS
 match access-group name ACL_DMZ_HOSTS
!
! Traffic from INSIDE to DMZ (note the two ACLs used)
!
```

```
class-map type inspect match-all CMAP_INSIDE_TO_DMZ_ACCESS
 match class-map CMAP_INSIDE_TO_DMZ_PROTOCOLS
 match access-group name ACL_DMZ_HOSTS

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Policy-Maps for Configured Zones
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!
! Policy-maps for INSIDE, OUTSIDE, DMZ
!
policy-map type inspect PMAP_INSIDE_TO_OUTSIDE
 class CMAP_INSIDE_TO_OUTSIDE_ACCESS
  inspect
!
policy-map type inspect PMAP_INSIDE_TO_DMZ
 class CMAP_INSIDE_TO_DMZ_ACCESS
  inspect
!
policy-map type inspect PMAP_OUTSIDE_TO_DMZ
 class CMAP_OUTSIDE_TO_DMZ_ACCESS
  inspect

!
! Policy-maps dealing with the zone self.
!
policy-map type inspect PMAP_OUTSIDE_TO_SELF
 class CMAP_HTTPS_SSH
  inspect
 class CMAP_RIP
  pass
!
policy-map type inspect PMAP_DMZ_TO_SELF
 class CMAP_HTTPS_SSH
  inspect
!
policy-map type inspect PMAP_SELF_TO_ANY
 class CMAP_TELNET
  no pass
  inspect
 class CMAP_ICMP
  no pass
  inspect
class CMAP_RIP
  pass
```

Copyright © 2010 Internetwork Expert

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Zones and Zone-Pairs
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

zone security INSIDE
zone security OUTSIDE
zone security DMZ
!
zone-pair security ZP_INSIDE_TO_OUTSIDE source INSIDE destination
OUTSIDE
 service-policy type inspect PMAP_INSIDE_TO_OUTSIDE
!
zone-pair security ZP_INSIDE_TO_DMZ source INSIDE destination DMZ
 service-policy type inspect  PMAP_INSIDE_TO_DMZ
!
zone-pair security ZP_OUTSIDE_TO_DMZ source OUTSIDE destination DMZ
 service-policy type inspect  PMAP_OUTSIDE_TO_DMZ
!
zone-pair security ZP_OUTSIDE_TO_SELF source OUTSIDE destination self
 service-policy type inspect  PMAP_OUTSIDE_TO_SELF
!
zone-pair security ZP_DMZ_TO_SELF source DMZ destination self
 service-policy type inspect PMAP_DMZ_TO_SELF
!
zone-pair security ZP_SELF_TO_OUTSIDE source self destination OUTSIDE
 service-policy type inspect PMAP_SELF_TO_ANY
!
zone-pair security ZP_SELF_TO_DMZ source self destination DMZ
 service-policy type inspect PMAP_SELF_TO_ANY
!
! Apply zones to interfaces
!
interface Serial 0/1/0
 zone-member security INSIDE
!
interface FastEthernet 0/0
 zone-member security OUTSIDE
!
interface FastEthernet 0/1
 zone-member security DMZ
```

SW3:

```
interface vlan 5
 ip address 155.1.5.9 255.255.255.0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> To test the Zone Based Firewall configuration, generate traffic flows between different zones. Start with the INSIDE to OUTSIDE direction. Use the **show policy-map** command on R5 to display statistics for the respective policy and look for packet matches. There are other protocols inspected by the policy, but we don't test them in this verification.

```
Rack1R4#ping 150.1.8.8

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms
Rack1R1#

Rack1R5#show policy-map type inspect zone-pair ZP_INSIDE_TO_OUTSIDE
 Zone-pair: ZP_INSIDE_TO_OUTSIDE

  Service-policy inspect : PMAP_INSIDE_TO_OUTSIDE

    Class-map: CMAP_INSIDE_TO_OUTSIDE_ACCESS (match-all)
      Match: access-group name ACL_INSIDE_USERS
      Match: class-map match-any CMAP_INSIDE_TO_OUTSIDE_PROTOCOLS
        Match: protocol http
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol ftp
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol icmp
          1 packets, 80 bytes
          30 second rate 0 bps
        Match: protocol dns
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol ssh
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol telnet
          1 packets, 24 bytes
          30 second rate 0 bps
        Match: protocol aol
          0 packets, 0 bytes
          30 second rate 0 bps
      Inspect
        Packet inspection statistics [process switch:fast switch]
        tcp packets: [0:41]
        icmp packets: [0:10]

        Session creations since subsystem startup or last reset 2
```

Copyright © 2010 Internetwork Expert

```
        Current session counts (estab/half-open/terminating) [1:0:0]
        Maxever session counts (estab/half-open/terminating) [1:1:1]
        Last session created 00:00:08
        Last statistic reset never
        Last session creation rate 2
        Maxever session creation rate 2
        Last half-open session total 0

    Class-map: class-default (match-any)
      Match: any
      Drop (default action)
        6 packets, 368 bytes
```

> **Note**
>
> While telnet session is still connected, check the sessions inspected by the ZFW. Notice the session corresponding to your telnet connection.

```
Rack1R5#show policy-map type inspect zone-pair ZP_INSIDE_TO_OUTSIDE
sessions
 Zone-pair: ZP_INSIDE_TO_OUTSIDE

  Service-policy inspect : PMAP_INSIDE_TO_OUTSIDE

    Class-map: CMAP_INSIDE_TO_OUTSIDE_ACCESS (match-all)
      Match: access-group name ACL_INSIDE_USERS
      Match: class-map match-any CMAP_INSIDE_TO_OUTSIDE_PROTOCOLS
        Match: protocol http
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol ftp
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol icmp
          1 packets, 80 bytes
          30 second rate 0 bps
        Match: protocol dns
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol ssh
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol telnet
          2 packets, 48 bytes
          30 second rate 0 bps
        Match: protocol aol
          0 packets, 0 bytes
          30 second rate 0 bps
      Inspect
        Established Sessions
         Session 844E2620 (155.1.13.1:17513)=>(150.1.2.2:23) telnet
SIS_OPEN
          Created 00:01:02, Last heard 00:01:00
```

```
          Bytes sent (initiator:responder) [37:79]

    Class-map: class-default (match-any)
      Match: any
      Drop (default action)
        6 packets, 368 bytes
```

> **Note**
>
> Now perform the similar tests for connections from OUTSIDE and INSIDE to the DMZ zone. Generate some traffic first...

```
Rack1SW2#telnet 155.1.45.4 80
Trying 155.1.45.4, 80 ... Open
GET /

HTTP/1.1 200 OK
<snip>

Rack1SW4#telnet 155.1.45.4 80
Trying 155.1.45.4, 80 ... Open
GET /
HTTP/1.1 200 OK
<snip>
```

Copyright © 2010 Internetwork Expert

Note

Check the policy-map statistics after this:

```
Rack1R5#show policy-map type inspect zone-pair ZP_INSIDE_TO_DMZ
 Zone-pair: ZP_INSIDE_TO_DMZ

  Service-policy inspect : PMAP_INSIDE_TO_DMZ

    Class-map: CMAP_INSIDE_TO_DMZ_ACCESS (match-all)
      Match: access-group name ACL_DMZ_HOSTS
      Match: class-map match-any CMAP_INSIDE_TO_DMZ_PROTOCOLS
        Match: protocol http
          1 packets, 24 bytes
          30 second rate 0 bps
        Match: protocol ftp
          3 packets, 72 bytes
          30 second rate 0 bps
        Match: protocol dns
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol tacacs
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol ssh
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol https
          0 packets, 0 bytes
          30 second rate 0 bps
      Inspect
        Packet inspection statistics [process switch:fast switch]
        tcp packets: [6:38]
        ftp packets: [0:1]

        Session creations since subsystem startup or last reset 2
        Current session counts (estab/half-open/terminating) [0:0:0]
        Maxever session counts (estab/half-open/terminating) [1:1:1]
        Last session created 00:01:35
        Last statistic reset never
        Last session creation rate 0
        Maxever session creation rate 1
        Last half-open session total 0

    Class-map: class-default (match-any)
      Match: any
      Drop (default action)
        10 packets, 688 bytes
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show policy-map type inspect zone-pair ZP_OUTSIDE_TO_DMZ
 Zone-pair: ZP_OUTSIDE_TO_DMZ

  Service-policy inspect : PMAP_OUTSIDE_TO_DMZ

    Class-map: CMAP_OUTSIDE_TO_DMZ_ACCESS (match-all)
      Match: class-map match-any CMAP_OUTSIDE_TO_DMZ_PROTOCOLS
        Match: protocol http
          1 packets, 24 bytes
          30 second rate 0 bps
        Match: protocol ftp
          3 packets, 72 bytes
          30 second rate 0 bps
        Match: protocol dns
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol tacacs
          0 packets, 0 bytes
          30 second rate 0 bps
      Match: access-group name ACL_DMZ_HOSTS
      Inspect
        Packet inspection statistics [process switch:fast switch]
        tcp packets: [3:32]
        ftp packets: [0:1]

        Session creations since subsystem startup or last reset 2
        Current session counts (estab/half-open/terminating) [0:0:0]
        Maxever session counts (estab/half-open/terminating) [1:1:1]
        Last session created 00:01:07
        Last statistic reset never
        Last session creation rate 0
        Maxever session creation rate 2
        Last half-open session total 0

    Class-map: class-default (match-any)
      Match: any
      Drop (default action)
        0 packets, 0 bytes
```

Copyright © 2010 Internetwork Expert

Note

Finally, we check the inspection of the router-generated traffic. Per the configuration, users should be able to ping and telnet off the router. We test this assumption by generating traffic to the OUTSIDE zone.

```
Rack1R5#ping 150.1.8.8

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.8.8, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/4/8 ms

Rack1R5#telnet 150.1.8.8
Trying 150.1.8.8 ... Open


User Access Verification

Password:
Rack1SW2>exit

[Connection to 150.1.8.8 closed by foreign host]
```

Note

Check the matches for the policy-map applied to traffic from zone self to the zone OUTSIDE. Notice that there was one session created per every class.

```
Rack1R5#show policy-map type inspect zone-pair ZP_SELF_TO_OUTSIDE
 Zone-pair: ZP_SELF_TO_OUTSIDE

  Service-policy inspect : PMAP_SELF_TO_ANY

    Class-map: CMAP_TELNET (match-all)
      Match: access-group name ACL_TELNET
      Match: protocol tcp
      Inspect
        Packet inspection statistics [process switch:fast switch]
        tcp packets: [40:0]

        Session creations since subsystem startup or last reset 1
        Current session counts (estab/half-open/terminating) [0:0:0]
        Maxever session counts (estab/half-open/terminating) [1:1:1]
        Last session created 00:00:07
        Last statistic reset never
        Last session creation rate 1
        Maxever session creation rate 1
        Last half-open session total 0
```

Copyright © 2010 Internetwork Expert

```
    Class-map: CMAP_ICMP (match-all)
      Match: protocol icmp
      Inspect
        Packet inspection statistics [process switch:fast switch]
        icmp packets: [20:0]

        Session creations since subsystem startup or last reset 2
        Current session counts (estab/half-open/terminating) [0:0:0]
        Maxever session counts (estab/half-open/terminating) [1:1:0]
        Last session created 00:00:10
        Last statistic reset never
        Last session creation rate 1
        Maxever session creation rate 1
        Last half-open session total 0

    Class-map: class-default (match-any)
      Match: any
      Drop (default action)
        0 packets, 0 bytes
```

Copyright © 2010 Internetwork Expert

Note

Now, make sure that the users on the OUTSIDE zone can access the router using SSH but not telnet. Additionally, make sure that the OUTSIDE users cannot ping the firewall.

```
Rack1SW2#ping 155.1.58.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.58.5, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)

Rack1SW2#telnet 155.1.58.5
Trying 155.1.58.5 ...
% Connection timed out; remote host not responding

Rack1SW2#telnet 155.1.58.5 22
Trying 155.1.58.5, 22 ... Open
SSH-1.99-Cisco-1.25

Rack1R5#show policy-map type inspect zone-pair ZP_OUTSIDE_TO_SELF
 Zone-pair: ZP_OUTSIDE_TO_SELF

  Service-policy inspect : PMAP_OUTSIDE_TO_SELF

    Class-map: CMAP_HTTPS_SSH (match-all)
      Match: access-group name ACL_SSH_HTTPS
      Match: protocol tcp
      Inspect
        Packet inspection statistics [process switch:fast switch]
        tcp packets: [11:0]

        Session creations since subsystem startup or last reset 1
        Current session counts (estab/half-open/terminating) [0:0:0]
        Maxever session counts (estab/half-open/terminating) [1:1:1]
        Last session created 00:00:10
        Last statistic reset never
        Last session creation rate 1
        Maxever session creation rate 1
        Last half-open session total 0

    Class-map: CMAP_RIP (match-all)
      Match: access-group name ACL_RIP
      Pass
        0 packets, 0 bytes

    Class-map: class-default (match-any)
      Match: any
      Drop (default action)
        20 packets, 1144 bytes
```

Copyright © 2010 Internetwork Expert

11.40 ZFW Rate Limiting

- In order to prevent DoS attacks, limit the OUTSIDE to DMZ traffic flow to 512Kbps.
- Enable half-open session limiting, so that no more than 2000 half-open sessions are allowed and no more than 100 new half-open sessions generated per minute are permitted.
- Stop dropping the half-open session states once the absolute amount of half-open connections falls below 1000 and the rate falls below 10.

Configuration

Note

Zone based firewall supports two types of rate-limiting:

1) Limiting aggregate packet rate for the flows between security zones.
2) Limiting the maximum number and/or rate of the half-open connections for TCP/UDP sessions.

The first feature resembles QoS traffic policing, but applies to traffic matching the inspect class maps and essentially limits the aggregated flows between zones, and not out/in of a single interface. Thus if you have multiple interfaces in a single zone, the configuration will apply aggregate limiting across all interfaces. The command to apply policing is **`police rate <RATE Bps> burst <Burst in bytes>`** configured under the respective class of a policy-map. Any traffic bursts exceeding the configured rate are dropped by the firewall, i.e. you cannot apply advanced actions such as remarking. The smaller is the burst, the less traffic could be sent instantly after an idle period. By setting larger burst value, you ensure smoother traffic flows but risk getting sudden spikes of heavy traffic bursts. There is no optimal value for the burst value – they are usually picked up based on experimental data. Here is an example of ZFW policing configuration:

```
policy-map type inspect PMAP_OUTSIDE_TO_INSIDE
 class ICMP
   inspect
   police 256000 burst 8000
```

Notice that you need the inspect action configured prior to enabling the policing. One limitation of the traffic-rate policing is that it is not supported with the zone self – i.e. for any pair that includes this zone.

Copyright © 2010 Internetwork Expert

The other rate-limiting feature applies to the TCP/UDP based connections. It helps preventing resource exhaustion DoS attacks, such as SYN flooding and behaves similarly to TCP Intercept feature. However, the only mode available is "watch" mode where firewall passively observes connections and tears them apart when thresholds are crossed. The UDP sessions are emulated as bidirectional UDP packets exchange that time out after some configurable timer value.

In order to apply connection-based limits to a particular class applied under a policy map you need to define an inspect parameter-map first. This object defines various connection attributes including TCP-intercept like parameters and various timeouts. The values defined in the parameter-map are similar to the CBAC parameter values familiar to you from the classic firewall configuration. Here is a full-list of the inspect parameter-map configurable options.

```
parameter-map type inspect parameter-map-name
 alert {on|off}
 audit-trail {on|off}
 dns-timeout <seconds>
 icmp idle-timeout <seconds>
 max-incomplete {low <num-of-conn> | high <num-of-conn>}
 one-minute {low <num-of-conn> | high <num-of-conn>}
 tcp max-incomplete host <threshold> [block-time <minutes>]
 sessions maximum <sessions>
 tcp finwait-time <seconds>
 tcp idle-time <seconds>
 tcp synwait-time <seconds>
 udp idle-time <seconds>
```

The **alert** and **audit-trail** options define whether to produce an alert for particular event (e.g. security violation, improper TCP stream structure, etc) and generate connection statistics information, e.g. number of bytes send/received. The various timeout values (e.g. **dns-timeout, tcp finwait-time, tcp idle-time**) specify the amount of time to expire a particular type of connection.

The setting for TCP-intercept feature are:

max-incomplete low/high – specify the low (stop dropping half-open sessions) and high (start dropping half-open sessions) absolute thresholds for the half-open TCP/UDP connections. A half-open UDP connection is the connection that sent but not received any packet.
one-minute low/high – specify the per-minute thresholds for the number of session. This effectively controls the rate of the half-open connections.

Copyright © 2010 Internetwork Expert

tcp max-incomplete host – sets the number of half-open sessions allowed per host and the time to block this host after the limit has been exceeded.

In order to apply the parameter-map to a zone pair, use the option to the **inspect** command configured under a particular class. The traffic flows matching this class will inherit the parameters specified by the parameter map. Other classes will use the default parameter-map that could be viewed using the command **show parameter-map type inspect default**. Here is an example that limits the number of session to one hundred for the internal users:

```
parameter-map type inspect PMAP_PARAMS
  sessions maximum 100
!
policy-map type inspect PMAP_INSIDE_TO_OUTSIDE
  cass INSIDE_TRAFFIC
    inspect PMAP_PARAMS
```

R5:

```
parameter-map type inspect PMAP_PARAMS
 max-incomplete low 1000
 max-incomplete high 2000
 one-minute low 10
 one-minute high 100
 tcp max-incomplete host 200 block-time 1
 sessions maximum 5000
 dns-timeout 15

!
! The burst rate does not matter much, unless you have to
! tune the application performance in real-life scenarios.
!
policy-map type inspect PMAP_OUTSIDE_TO_DMZ
 class CMAP_OUTSIDE_TO_DMZ_ACCESS
  police rate 512000 burst 32000
  inspect PMAP_PARAMS
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> First, check the default values configured in the system. Next, review the configured map and compare it with the default settings. Notice that the default values for the session interception feature are unlimited, meaning no session control is used.

```
Rack1R5#show parameter-map type inspect default
 parameter-map type inspect default values
  audit-trail on
  alert off
  max-incomplete low  unlimited
  max-incomplete high unlimited
  one-minute low  unlimited
  one-minute high unlimited
  udp idle-time 10
  icmp idle-time 5
  dns-timeout 0
  tcp idle-time 3600
  tcp finwait-time 5
  tcp synwait-time 30
  tcp max-incomplete host unlimited block-time 0
  sessions maximum 0

Rack1R5#show parameter-map type inspect
 parameter-map type inspect PMAP_PARAMS
  audit-trail on
  alert off
  max-incomplete low  1000
  max-incomplete high 2000
  one-minute low  10
  one-minute high 100
  udp idle-time 10
  icmp idle-time 5
  dns-timeout 15
  tcp idle-time 3600
  tcp finwait-time 5
  tcp synwait-time 30
  tcp max-incomplete host 200 block-time 1
  sessions maximum 5000

Rack1R5#show policy-map type inspect PMAP_OUTSIDE_TO_DMZ
  Policy Map type inspect PMAP_OUTSIDE_TO_DMZ
    Class CMAP_OUTSIDE_TO_DMZ_ACCESS
      Inspect PMAP_PARAMS
      Police rate 512000 burst 32000
    Class class-default
```

Copyright © 2010 Internetwork Expert

> **Note**
>
> Now add ICMP to the list of protocols inspected from OUTSIDE to DMZ and generate a flood of ICMP packets from OUTSIDE to the AAA server.

```
Rack1R5(config)#class-map type inspect CMAP_OUTSIDE_TO_DMZ_PROTOCOLS
Rack1R5(config-cmap)#match protocol icmp

Rack1SW2#ping 155.1.5.9 size 1000 timeout 0 repeat 1000

Type escape sequence to abort.
Sending 1000, 1000-byte ICMP Echos to 155.1.5.9, timeout is 0 seconds:
 ...............
 ...............
 ...............
 ...............
 ...............
 ...............
 ...............
 ...............
 ...............
 ...............
 ...............
 ...............
 ...............
 ...............
 ...............
Success rate is 0 percent (0/1000)
Rack1R2#
```

> **Note**
>
> Now check the statistics for the policy applied to the respective zone pair. Notice that the policing statistics appears on top of the show command output. Approximately 10% of the ICMP packets exceeded the configured settings.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show policy-map type inspect zone-pair ZP_OUTSIDE_TO_DMZ
 Zone-pair: ZP_OUTSIDE_TO_DMZ
      Police
      rate 512000 bps,32000 limit
      conformed 118 packets, 120124 bytes; actions: transmit
      exceeded 983 packets, 1000694 bytes; actions: drop
      conformed 0 bps, exceed 0 bps

  Service-policy inspect : PMAP_OUTSIDE_TO_DMZ

    Class-map: CMAP_OUTSIDE_TO_DMZ_ACCESS (match-all)
      Match: class-map match-any CMAP_OUTSIDE_TO_DMZ_PROTOCOLS
        Match: protocol http
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol ftp
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol dns
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol tacacs
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol icmp
          1 packets, 980 bytes
          30 second rate 0 bps
      Match: access-group name ACL_DMZ_HOSTS
      Inspect
        Packet inspection statistics [process switch:fast switch]
        tcp packets: [3:32]
        icmp packets: [0:118]
        ftp packets: [0:1]

        Session creations since subsystem startup or last reset 3
        Current session counts (estab/half-open/terminating) [1:0:0]
        Maxever session counts (estab/half-open/terminating) [1:1:1]
        Last session created 00:00:05
        Last statistic reset never
        Last session creation rate 1
        Maxever session creation rate 2
        Last half-open session total 0

    Class-map: class-default (match-any)
      Match: any
      Drop (default action)
        2000 packets, 1960000 bytes
```

Copyright © 2010 Internetwork Expert

11.41 ZFW Application Inspection

- Configure the firewall so that the internal users are not allowed to use insecure POP3/IMAP4 logins to the outside servers.
- Block image downloads from digg.com for internal users and log any attempts to use AOL messenger for non text-chat service (e.g. file transfer).
- Image files have filename extensions .jpg, .png, .gif
- Log all violations.

Configuration

Note

Application inspection and control (AIC) applies at Layer 7 of the OSI model, allowing for policy configuration based on application protocol parameters. Applications are the most vulnerable part of your network, and thus enforcing strict per-application policy is important to increase the overall security. For example, you may want your firewall to block certain SMTP commands that open vulnerabilities in the servers. Alternatively, you may want to make sure that HTTP connections are not used to tunnel instant-messenger services or used to connect to other non-HTTP services. This is only possible with application-level inspection. Of course, there are numerous limitations for IOS AIC, if you compare this feature with ASA's MPF capabilities.

Cisco IOS Software ZFW offers application inspection and control on these application services:

```
HTTP
SMTP/ESMTP
POP3/IMAP
P2P Application Traffic (eDonkey, Kazaa2, GNUtella etc)
IM Applications
Sun RPC (MSN/AOL/Y!M)
```

AIC uses special type of class-maps and policy-maps to configure application protocol inspection. Those are *application specific* inspection class-maps and policy-maps for example **`class-map type inspect http`**, **`policy-map type inspect http`** and so on, for every application protocol supported. The special class-map defines protocol parameters and should be used inside the special policy map as usual. The special policy-map should be later applied as a *nested (child)* service-policy under the generic inspection policy-map. For example:

Copyright © 2010 Internetwork Expert

```
class-map type inspect http match-any CMAP_HTTP_VIOLATION
  match req-resp protocol-violation
!
policy-map type inspect http PMAP_HTTP_VIOLATION_RESET
  class type inspect http CMAP_HTTP_VIOLATION
   reset
!
class-map type inspect match-any CMAP_HTTP_TRAFFIC
 match protocol http
!
policy-map type inspect PMAP_INSIDE_OUTSIDE
 class type inspect CMAP_HTTP_TRAFFIC
  inspect
  service-policy http PMAP_HTTP_VIOLATION_RESET
```

It is important that the parent L3/L4 class-map (CMAP_HTTP_TRAFFIC in our case) define only the protocols supported by AIC. For example, if the parent class-map defines FTP protocol (**match protocol ftp**), you cannot apply the nested service-policy for any AIC supported protocol, as FTP is not supported by AIC.

Application inspection and control (AIC) has different set of features for every of the above-mentioned protocols.

1) HTTP inspection offers granular filtering on several types of application activity, including different methods (GET/POST), HTTP header contents and URIs. Additionally HTTP inspection allows you limiting document transfer size, web address lengths, and the flow of browser requests/server replies to enforce compliance with application-behavior standards. Finally, you may limit types of content (e.g. block java apps, images, videos or compressed content) that are transferred over HTTP connections.

2) SMTP inspection can limit content length and enforce protocol compliance. It checks for command syntax and permits only the legal SMTP/ESMTP commands.

3) POP3 and IMAP inspection can help ensure that users are using secure authentication mechanisms to prevent compromise of user credentials. This is illustrated in this particular scenario.

One special type of objects that is often used with the application inspection are regular-expressions parameter maps. It allows you defining sets of regular expressions to be later used with application inspection, for example to match URL strings or command parameters, such as FTP GET filename or SMTP recipient. The syntax to define a regular-expression parameter map is:

Copyright © 2010 Internetwork Expert

```
parameter-map type regex REGEX_CISCO
  pattern  [cC][iI][sS][cC][oO]
!
class-map type inspect http PMAP_CISCO
 match request header host regex REGEX_CISCO
```

When matching HTTP header values , don't forget the leading ".*" as Cisco IOS compiles the resulting regex with the header name prepended, e.g. "[hH][oO][sS][tT]:<pattern>" when you match the "host" header field against "<pattern>". Additionally, be careful when using the "$" anchor, as every HTTP line ends with \CR\LF and you would need to match those as well. Try not to use regex anchors at all, when matching various protocol header fields.

You may find the detailed information on the application inspection parameters in Cisco's documentation. From the lab perspective, make sure you have a good understanding of application filtering and basic application-specific feature, and refer to the documentation for additional information.

R5:

```
!
! Define regular expression first
!
parameter-map type regex REGEX_IMAGES
 pattern .*\.([jJ][pP][gG]|[pP][nN][gG]|[gG][iI][fF])
!
parameter-map type regex REGEX_DIGG
 pattern .*[dD][iI][gG][gG]\.[cC][oO][mM]

!
! The class that matches image URLs on Digg
!
class-map type inspect http match-all CMAP_DIGG_IMAGES
 match request header host regex REGEX_DIGG
 match request uri regex REGEX_IMAGES
 no match request arg regex REGEX_IMAGES

!
! Insecure POP3/IMAP4 logins
!
class-map type inspect pop3 CMAP_POP3_INSECURE
 match login clear-text
!
class-map type inspect imap CMAP_IMAP_INSECURE
 match login clear-text
```

Copyright © 2010 Internetwork Expert

```
!
! AOL Chat service
!
class-map type inspect aol CMAP_AOL_NON_CHAT
 match no service text-chat

!
! Policy-map type inspect IM could use MSN, Y!M and AOL
! type class-maps. They are all unified under a single policy
! of type IM
!
policy-map type inspect im PMAP_AOL_POLICY
 class CMAP_AOL_NON_CHAT
  log

policy-map type inspect http PMAP_HTTP_POLICY
 class CMAP_DIGG_IMAGES
  reset
  log
!
policy-map type inspect pop3 PMAP_POP3_POLICY
 class CMAP_POP3_INSECURE
  reset
  log
!
policy-map type inspect imap PMAP_IMAP_POLICY
 class CMAP_IMAP_INSECURE
  reset
  log

!
! We need to de-configure grouped protocol matching
! and create separate class-maps for every protocol
! that is to be inspected in-depth. This is to avoid
! possible collisions with protocols not supported by AIC
!
class-map type inspect CMAP_INSIDE_TO_OUTSIDE_PROTOCOLS
 no match protocol http
 no match protocol aol
 no match protocol pop3
 no match protocol imap
```

Copyright © 2010 Internetwork Expert

```
!
! Protocol-specific class-maps
!
class-map type inspect match-all CMAP_INSIDE_TO_OUTSIDE_HTTP
 match protocol http
 match access-group name ACL_INSIDE_USERS
!
class-map type inspect match-all CMAP_INSIDE_TO_OUTSIDE_AOL
 match protocol aol
 match access-group name ACL_INSIDE_USERS
!
class-map type inspect match-all CMAP_INSIDE_TO_OUTSIDE_POP3
 match protocol pop3
 match access-group name ACL_INSIDE_USERS
!
class-map type inspect match-all CMAP_INSIDE_TO_OUTSIDE_IMAP
 match protocol imap
 match access-group name ACL_INSIDE_USERS

!
! Modify the zone-pair policy-maps
!
policy-map type inspect PMAP_INSIDE_TO_OUTSIDE
 class CMAP_INSIDE_TO_OUTSIDE_HTTP
  inspect
  service-policy http PMAP_HTTP_POLICY
class CMAP_INSIDE_TO_OUTSIDE_AOL
  inspect
  service-policy im PMAP_AOL_POLICY
class CMAP_INSIDE_TO_OUTSIDE_POP3
  inspect
  service-policy pop3 PMAP_POP3_POLICY
class CMAP_INSIDE_TO_OUTSIDE_IMAP
  inspect
  service-policy imap PMAP_IMAP_POLICY
```

Copyright © 2010 Internetwork Expert

Verification

Note

We are going to test some of the application firewall capabilities. First, we configure R2 as HTTP server. Next, we emulate an HTTP GET request from R1 requesting the file “image.jpg” from the virtual host “digg.com”. Notice the message logged in R3:

```
Rack1R1#telnet 150.1.2.2 80
Trying 150.1.2.2, 80 ... Open
GET /image.jpg HTTP/1.1
Host: digg.com

[Connection to 150.1.2.2 closed by foreign host]

%APPFW-4-HTTP_URI_REGEX_MATCHED: URI regex
(.*\.([jJ][pP][gG]|[pP][nN][gG]|[gG][iI][fF])) matched - resetting
session 155.1.13.1:60627 150.1.2.2:80 on zone-pair ZP_INSIDE_TO_OUTSIDE
class CMAP_INSIDE_TO_OUTSIDE_HTTP appl-class CMAP_DIGG_IMAGES

%APPFW-4-HTTP_HDR_FIELD_REGEX_MATCHED: Header field
(^[Hh][Oo][Ss][Tt]:.*[dD][iI][gG][gG]\.[cC][oO][mM]) matched -
resetting session 155.1.13.1:60627 150.1.2.2:80 on zone-pair
ZP_INSIDE_TO_OUTSIDE class CMAP_INSIDE_TO_OUTSIDE_HTTP appl-class
CMAP_DIGG_IMAGES
```

Copyright © 2010 Internetwork Expert

Note

Now let's try checking POP3 inspection. Since we don't have a POP3 server running in R2, let's use a trick. Enable TCP "echo" service in R2 and configure R3 to inspect POP3 on the TCP/7 port for the IP address 155.X.23.2. After this, connect from R1 to R2 on port 7 and issue POP3 "USER" command. Notice the debugging output in the firewall, showing that inspection encountered invalid command.

R3:

```
access-list 99 permit 155.1.23.2
!
ip port-map pop3 port tcp 7 list 99

Rack1R1#telnet 155.1.23.2 7
Trying 155.1.23.2, 7 ... Open
UUSSEERR  ppeetrtr

[Connection to 155.1.23.2 closed by foreign host]

%FW-5-POP3_INVALID_COMMAND: (target:class)-
(ZP_INSIDE_TO_OUTSIDE:CMAP_INSIDE_TO_OUTSIDE_POP3):Invalid POP3 command
from initiator (155.1.13.1:18292): Invalid verb
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

11.42 Classic IOS Transparent Firewall

- Configure R6 as a transparent firewall deployed on the subnet 10.0.0.0/24.
- Change the IP addresses for VLAN67 and VLAN146 interface of SW1, R1 and R4 to reflect this.
- VLAN67 users are allowed to originate HTTP, FTP, ICMP, DNS and SSH traffic across the firewall to VLAN146.
- VLAN146 users are only allowed initiating FTP (passive), HTTP and DNS connections to the serves on VLAN 67
- Do not permit any IPv6 traffic across the firewall.

Configuration

Note

IOS Transparent firewall was introduced in IOS version 12.3(7)T. It allows for traffic filtering and stateful inspection when the router acts as Layer 2 bridge. The benefit of using this operational mode is the ease of deployment – the end users are not required to change any networking settings – the firewall acts as the bump on a wire.

In transparent firewall mode, the router does not attempt any routing and simple learns Ethernet MAC addresses on all interfaces, performing Ethernet frame switching based on destination MAC addresses. In order to configure the IOS firewall as a bridge, you should first define a type of bridging configured in the firewall. This could be either CRB (Concurrent Routing and Bridging) or IRB (Integrated Routing and Bridging). Both use the concept of a bridge group, which is essentially a virtual bridge inside the router, with its own MAC address table.

1) CRB mode is enabled using the command **`bridge crb`**. Allows the router to bridge frames between interfaces configured as members of a single bridge group. Interfaces not in any bridge group with IP addresses assigned may accept and forward IP packets. Essentially, the router acts as a bridge for one set of interfaces and L3 router for the rest of the interfaces.

Copyright © 2010 Internetwork Expert

2) IRB mode is enabled using the command **bridge irb**. allows configuring a special BVI (Bridge Virtual Interface) for every bridge group. This interface represents the router as L3 device within the respective bridge group. All devices in the group aware of this BVI's MAC address may send IP packets to the router and communicate with other L2 domains. This mode allows the router to bridge between the interfaces and route packets for the members of the bridge-group that are aware of this service. You may compare BVI with an SVI (Switch Virtual Interface, e.g. VLAN 31) in a Layer 3 switch. In addition to the **bridge irb** command you need to add the command **bridge <number> route ip** where <number> is the bridge-group number. Otherwise, the BVI interface will not route the IP packets.

In order to configure an interface/subinterface as member of a bridge group, use the command **bridge-group <number>** in interface configuration mode. By default, the new group starts with Spanning-Tree protocol disabled. You may need the command **bridge <n> protocol IEEE** to enable the IEEE-compatible spanning tree for the bridge-group. Notice that for 802.1Q subinterfaces the firewall will run the PVST+ variant of STP.

You may apply the classic firewall inspect rules and access-lists to the interface configured with the bridge-group numbers after this. For example:

```
bridge irb
bridge-group 1 protocol ieee
bridge 1 route ip
!
ip inspect name CBAC tcp
ip inspect name CBAC udp
ip inspect name CBAC icmp
!
ip acess-list extended OUTSIDE_IN
 deny ip any any log
!
interface FastEthernet 0/0
 bridge-group 1
 ip access-group OUTSIDE_IN in
!
interface FastEthernet 0/1
 bridge-group 1
 ip inspect CBAC in
```

Copyright © 2010 Internetwork Expert

The inspection rules apply to any support IP-based protocol. Only TCP/UDP/ICMP flows are supported by L2 transparent firewall. All non-IP packets are permitted by default across the firewall, including ARP, STP, IPv6 and so on. In order to block a non-IP protocol, you need to apply a protocol-type access-list to the bridged interface. Those lists have their numbers in range 201-299 and permit traffic based on Ethertype of SNAP PID value. For example, the following list will permit IP and ARP but block IPX:

```
access-list 201 permit 0x800
access-list 201 permit 0x806
access-list 201 deny 0x8137
access-list 201 permit 0x0 0xFFFF
```

The Ethertype for now becoming popular IPv6 is 0x86DD, and you can use this value to filter this non-IP protocol. In order to associate a protocol-type list with an interface, use the interface-level command **`bridge-group <n> input-type-list <ACL#>`**.

Transparent firewall does not inspect or process multicast Ethernet frames – they are permitted regardless of any input acces-lists. However, broadcast frames (e.g. NetBIOS over TCP/UDP or DHCP) are subject to transparent ACL checks. Transparent firewall provides special handling for DHCP packets as those are important for host auto-configuration. The command to permit DHCP packets across the firewall regardless of the IP access-lists configuration is **`ip inspect l2-transparent dhcp-passthrough`**.

Lastly, a few words on the IP address assigned to the BVI interface. It must be on the same subnet as the protected segments. You may treat is as equivalent to the ASA firewall's management IP. The traffic originated from the router across the bridged interface is not subject to the checks by the ACL applied on the bridged interfaces. Instead, to control the management access, you may need a separate access-list attached to the BVI.

The solution for this task configures inspection in two directions: inside to outside,dmz and outside to DMZ. Notice that the access-list applied to the outside interface needs to permit the RIP protocol updates and the access to the DMZ subnet. The access-list applied ingress to the DMZ interface prevents any unauthorized traffic from returning. The configuration does not look elegant, and this is a direct result of the inspection rules being applied per-interface, not per security-zone.

Copyright © 2010 Internetwork Expert

SW1:

```
interface Vlan 67
 ip address 10.0.0.7 255.255.255.0
```

R1:

```
interface FastEthernet 0/0
 ip address 10.0.0.1 255.255.255.0
```

R4:

```
interface FastEthernet 0/1
 ip address 10.0.0.4 255.255.255.0
!
! Enable SSH for testing
!
ip domain-name cisco.com
crypto key generate rsa general modulus 768
```

R6:

```
bridge irb
bridge 1 protocol ieee
bridge 1 route ip
!
interface BVI1
 ip address 10.0.0.6 255.255.255.0
!
ip inspect name INSIDE_PROTOCOLS http
ip inspect name INSIDE_PROTOCOLS ftp
ip inspect name INSIDE_PROTOCOLS dns
ip inspect name INSIDE_PROTOCOLS ssh
ip inspect name INSIDE_PROTOCOLS icmp
!
ip inspect name OUTSIDE_PROTOCOLS ftp
ip inspect name OUTSIDE_PROTOCOLS http
ip inspect name OUTSIDE_PROTOCOLS dns


!
! Outside ACL, need to permit access to INSIDE
!
no ip access-list extended OUTSIDE_IN
ip access-list extended OUTSIDE_IN
 permit tcp any any eq 80
 permit udp any any eq 43
 permit udp any any eq 21
 deny ip any any log
```

Copyright © 2010 Internetwork Expert

```
!
! Block IPv6
!
access-list 201 deny 0x86dd
access-list 201 permit 0x0 0xFFFF

!
! Inside Interface
!
interface FastEthernet 0/0.67
 encapsulation dot1q 67
 no ip address
 bridge-group 1
 ip inspect INSIDE_PROTOCOLS in
 bridge-group 1 input-type-list 201

!
! Outside Interface
!
interface FastEthernet 0/1.146
 encapsulation dot1q 146
 no ip address
 bridge-group 1
 ip access-group OUTSIDE_IN in
 bridge-group 1 input-type-list 201
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Start verifications by checking the bridge-group status. Make sure you see MAC addresses learned at all interfaces configured for bridging and check the TX counters. It may take some time to learn MAC addresses, as this process requires traffic to be generated by devices.

```
Rack1R6#show bridge

Total of 300 station blocks, 297 free
Codes: P - permanent, S - self

Bridge Group 1:

    Address       Action   Interface       Age   RX count   TX count
0013.c451.f240   forward   Fa0/0.146         0        123         23
0013.c440.3980   forward   Fa0/0.67          0        297         23
```

> **Note**
>
> Check CBAC configuration next. We have two rules configured, one to inspect the internal sessions, and another to protect the DMZ servers. Notice that the DMZ rules do not allow ICMP traffic, but the inside rules do. Thus, the outside users are not allowed to ping the servers, while inside users can do this.

```
Rack1R6#show ip inspect config
Session audit trail is disabled
Session alert is enabled
one-minute (sampling period) thresholds are [unlimited : unlimited]
connections
max-incomplete sessions thresholds are [unlimited : unlimited]
max-incomplete tcp connections per host is unlimited. Block-time 0
minute.
tcp synwait-time is 30 sec -- tcp finwait-time is 5 sec
tcp idle-time is 3600 sec -- udp idle-time is 30 sec
tcp reassembly queue length 16; timeout 5 sec; memory-limit 1024 kilo
bytes
dns-timeout is 5 sec
Inspection Rule Configuration
 Inspection name INSIDE_PROTOCOLS
    http alert is on audit-trail is off timeout 3600
    ftp alert is on audit-trail is off timeout 3600
    dns alert is on audit-trail is off timeout 30
    ssh alert is on audit-trail is off timeout 30
    icmp alert is on audit-trail is off timeout 10
```

Copyright © 2010 Internetwork Expert

Note

Now test the firewall rules. Telnet from R4 to SW1 and confirm that the session is not established. At the same time, ensure that you can connect to R4 on SSH port and ping it R4 from SW1.

```
Rack1R4#telnet 10.0.0.7
Trying 10.0.0.7 ...
% Connection timed out; remote host not responding

R3:
%SEC-6-IPACCESSLOGP: list OUTSIDE_IN denied tcp 10.0.0.4(23) ->
10.0.0.7(19184), 1 packet

Rack1SW1#telnet 10.0.0.4 22
Trying 10.0.0.2, 22 ... Open

Rack1R6#show ip inspect sessions
Established Sessions
 Session 845FBEE8 (10.0.0.7:48036)=>(10.0.0.4:22) ssh SIS_OPEN

Rack1SW1#ping 10.0.0.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/12/48 ms
```

Note

Now check the traffic from outside to inside. Traffic to the inside should be permitted to certain ports.

```
Rack1R4#telnet 10.0.0.7 80
Trying 10.0.0.7, 80 ... Open

Rack1R4#ping 10.0.0.7

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.7, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)
```

Copyright © 2010 Internetwork Expert

> **Note**
>
> Make sure that RIP updates are allowed across the firewall as they are multicast.

```
Rack1SW1#show ip route rip
<snip>
     150.1.0.0/24 is subnetted, 2 subnets
R       150.1.4.0 [120/1] via 10.0.0.4, 00:00:09, Vlan67
<snip>
```

```
Rack1R6#show ip inspect statistics
Packet inspection statistics [process switch:fast switch]
  tcp packets: [0:11]
   packets: [2:36]
Interfaces configured for inspection 2
Session creations since subsystem startup or last reset 5
Current session counts (estab/half-open/terminating) [0:0:0]
Maxever session counts (estab/half-open/terminating) [1:1:1]
Last session created 00:03:45
Last statistic reset never
Last session creation rate 0
Maxever session creation rate 1
Last half-open session total 0
<snip>
```

> **Note**
>
> Make sure the firewall can ping the connected devices thanks to the BVI configuration. This is because this traffic is not subject to the checks by the ACL applied to the firewall bridged interfaces.

```
Rack1R6#ping 10.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

Copyright © 2010 Internetwork Expert

Note

You may visualize L2 inspection process using the below debugging command. The output below shows the inspection process for the ICMP echo and echo-reply packets sent across the firewall.

```
Rack1R6#debug ip inspect l2-transparent packets
INSPECT L2 firewall debugging is on

Rack1SW1#ping 10.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 8/8/12 ms

RTBAP: Check AuthProxy is configured on idb=FastEthernet0/1.23 path=1
linktype=38
L2FW:Input ACL not configured or the ACL is bypassed
L2FW:Output ACL is not configured or ACL is bypassed
L2FW*:insp_l2_fast_inspection returning INSP_L2_OK
TBAP: Check if AP return traffic (fast path) foroutput
idb=FastEthernet0/0.67 IP->prot=1

L2FW*:insp_l2_fast_inspection: pak 83A7BB7C, input-interface
FastEthernet0/0.67, output-interface FastEthernet0/1.146
L2FW*:Src 10.0.0.7 dst 10.0.0.1 protocol icmp
TBAP: Check AuthProxy is configured on idb=FastEthernet0/0.67 path=1
linktype=38
```

Copyright © 2010 Internetwork Expert

11.43 ZFW-Based IOS Transparent Firewall

- Modify the previous scenario to use CPL for the firewall configuration.

Configuration

Note

It is possible to use CPL (Cisco Policy Language) and define transparent firewall rules using ZFW syntax. In fact, this approach has many advantages, as it operates with a better abstraction (zone) instead of interfaces and allows for easy to understand and concise configurations.

To configure ZFW, simply apply zone names to the interfaces configured for Ethernet bridging. It is possible, though not recommended to configure L3 and L2 interfaces in the same zone. However, in most cases you may want to keep those domains separate.

Any BVI interfaces on the router automatically belong to the "self" zone, and the policy should be configured appropriately. You may still use access-list on the BVIs, if there is no security zone assigned. If you want to treat the whole L2 domain behind the BVI as a single L3 security zone, you may simply assign a zone to the BVI, but not the bridging interfaces.

As for multicast and broadcast traffic types – both are not inspected by ZFW. Moreover, in transparent mode both traffic types are permitted across the firewall configured with zones. Keep this in mind when deploying ZFW configurations.

```
R6:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Traffic-classes definition
!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!
! Inside protocols
!
class-map type inspect match-any CMAP_PROTOCOLS_FROM_INSIDE
 match protocol http
 match protocol ftp
 match protocol dns
 match protocol ssh
 match protocol icmp
```

Copyright © 2010 Internetwork Expert

```
!
! Outside protocols
!
class-map type inspect match-any CMAP_PROTOCOLS_TO_INSIDE
 match protocol ftp
 match protocol http
 match protocol dns
!
!
class-map type inspect CMAP_RIP_TRAFFIC
 match access-group name ACL_RIP_TRAFFIC

!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Policies definition
!!!!!!!!!!!!!!!!!!!!!!!!!!!!

policy-map type inspect PMAP_INSIDE_TO_OUTSIDE
 class CMAP_PROTOCOLS_FROM_INSIDE
  inspect
!
policy-map type inspect PMAP_OUTSIDE_TO_INSIDE
 class CMAP_PROTOCOLS_TO_INSIDE
  inspect

!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Security zones and zone-pairs
!!!!!!!!!!!!!!!!!!!!!!!!!!!!

zone security INSIDE
zone security OUTSIDE

zone-pair security ZP_INSIDE_TO_OUTSIDE source INSIDE destination
OUTSIDE
 service-policy type inspect PMAP_INSIDE_TO_OUTSIDE
!
zone-pair security ZP_OUTSIDE_TO_INSIDE source OUTSIDE destination
INSIDE
 service-policy type inspect PMAP_OUTSIDE_TO_INSIDE
```

Copyright © 2010 Internetwork Expert

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Remove the classic configs
! and apply the zone-based
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!
! Inside Interface
!
interface FastEthernet 0/0.67
 bridge-group 1
 no ip inspect INSIDE_PROTOCOLS in
 zone-member security INSIDE
 bridge-group 1 input-type-list 201

!
! Outside Interface
!
interface FastEthernet 0/1.146
 bridge-group 1
 no ip access-group OUTSIDE_IN in
 bridge-group 1 input-type-list 201
 zone-member security OUTSIDE
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> You may apply the same verification techniques as were used in the previous task. First, make sure SW1 sees RIP updates from R4 as multicast packets are not subject to Transparent ZFW checks. Next, initiate a session from SW1 to R4 and check that it's being inspected by the firewall. Use the same approach you used in the previous scenario.

```
Rack1SW1#show ip route rip
<snip>
     150.1.0.0/24 is subnetted, 2 subnets
R       150.1.4.0 [120/1] via 10.0.0.2, 00:00:10, Vlan67

Rack1SW1#telnet 10.0.0.4 22
Trying 10.0.0.4, 22 ... Open
SSH-1.99-Cisco-1.25


Rack1R5#show policy-map type inspect zone-pair ZP_INSIDE_TO_OUTSIDE
sessions
 Zone-pair: ZP_INSIDE_TO_OUTSIDE

  Service-policy inspect : PMAP_INSIDE_TO_OUTSIDE

    Class-map: CMAP_PROTOCOLS_FROM_INSIDE (match-any)
      Match: protocol http
        0 packets, 0 bytes
        30 second rate 0 bps
      Match: protocol ftp
        0 packets, 0 bytes
        30 second rate 0 bps
      Match: protocol dns
        0 packets, 0 bytes
        30 second rate 0 bps
      Match: protocol ssh
        1 packets, 24 bytes
        30 second rate 0 bps
      Match: protocol icmp
        0 packets, 0 bytes
        30 second rate 0 bps
      Inspect
        Established Sessions
         Session 845FBC20 (10.0.0.7:30775)=>(10.0.0.4:22) ssh SIS_OPEN
          Created 00:00:03, Last heard 00:00:03
          Bytes sent (initiator:responder) [0:20]

<snip>
```

Copyright © 2010 Internetwork Expert

11.44 IOS IPS

- Configure R4 to deny inline the attacker that attempts to use IP loose source route option.
- For testing purposes, enable the ICMP echo signature and configure the IPS to notify syslog.

Configuration

Note

Historically, IOS IPS started its evolution since IOS 12.0(5)T, when a limited subset of Cisco IDS v3.x was introduced in IOS. The configuration syntax was based on the **`ip audit`** command, and IOS implemented a fixed number of IDS signatures (initially 59, but then 42 more signatures were added in IOS 12.2T). Cisco IPS used proprietary Post Office Protocol (POP) for event communication with IDS Director (part of Cisco IDS system). The configuration only supported two signature categories: "attack" and "info". The same feature is still available in PIX and ASA platforms, though supporting different signature sets.

With IOS release 12.3(8)T Cisco introduced IDS code 4.x into IOS and named the feature as IPS, replacing **`ip audit`** command with **`ip ips`** syntax. The IPS featured support for SDEE (Security Device Event Exchange) protocol instead of legacy POP and implemented IDS 4.x signature engines, such as ATOMIC.IP, ATOMIC.ICMP, SERVICE.HTTP, SERVICE.FTP, OTHER and so on. A total of 132 built-in signatures were supported in Cisco IPS. However, now in addition to the built-in signatures, it was possible to load an external SDF (Signature definition file, and XML file used IDS 4.x format) with additional patterns that were dynamically compiled in the IPS core. This allowed for some extensibility, though the number of extensions was limited. Theoretically, it was possible to create a custom signature using XML syntax, though Cisco never openly documented it. The total number of signatures available to IOS was significantly less compared to the list available in the hardware appliances.

Recently, with the IOS 12.4(11)T the support for IPS 5.x signatures has been added to IOS. More than that, there are no more hardcoded signatures in the IOS code itself. New v5.x signature packages (in format different from the old 4.x) are now required to provide IOS with signature for the IPS engine. The packages contain the same list of signatures that a normal v5.x hardware appliance uses, and which is really large. A router is supposed to load and compile the new signatures dynamically. The configuration for the IPS subsystem is now stored separately, in the flash memory of the router, and the whole configuration process is now very similar to the hardware IPS CLI-based configuration. However, there are serious limitations in functionality, such as inability to create custom signatures.

Copyright © 2010 Internetwork Expert

Here are the configuration steps to enable Cisco IPS functionality using v5.x signatures:

Step 1. Downloading IOS IPS files
Step 2. Creating IOS IPS configuration directory on flash
Step 3. Configuring IOS IPS crypto key
Step 4. Enabling IOS IPS
Step 5. Loading IOS IPS signature package to router
Step 6: Tuning Signatures

Downloading IOS IPS files.

Although this step is already performed for you in the lab environment, in real life scenarios you need to access the Security/IPS section of CCO and download the following files:

IOS-Sxxx-CLI.pkg: Signature package – make sure you downloaded the latest signature package. Here xxx is the package version.

realm-cisco.pub.key.txt: This is Cisco's public-key, used to sign the signature package. The router will use this key to verify the package integrity.

Creating IOS IPS configuration directory on flash.

As mentioned previously, the new IPS code store all configuration in special files located in the flash memory. It is recommended creating a separate directory in the flash memory to store the files. Use the **mkdir** command to accomplish this. The files stored in the flash memory are compressed and not editable. Here is a quick overview of the files (replace "router" with your router's hostname):

router-sigdef-default.xml - contains all the factory default signature definitions.

router-sigdef-delta.xml - contains signature definitions that have been changed from the default.

router-sigdef-typedef.xml - is a file that has all the signature parameter definitions.

router-sigdef-category.xml - has all the signature category information, such as category ios_ips basic and advanced.

Copyright © 2010 Internetwork Expert

router-seap-delta.xml - contains changes made to the default SEAP parameters. SEAP is the signature event processing engine, similar to the one found in the hardware appliance.

router-seap-typedef.xml - contains all the SEAP parameter definitions

Configuring IOS IPS crypto key.

Open the file **realm-cisco.pub.key.txt** in a text editor or list it content using the **more** command if the file is located in the flash memory. Copy the contents of the file, enter the configuration mode using the command **configure terminal** and paste the file content. This will import the public key into the routers configuration. Don't forget to save the running configuration after this.

Enabling IOS IPS.

This step comprises IPS initialization, creating an inspection rule, specifying the configuration file location and retiring unneeded signature categories. The first thing you need to do is to configure basic IPS settings:

```
ip ips name <NAME> [list <ACL_NAME>]
ip ips config location flash:<directory>
ip ips notify sdee
ip ips notify log
```

The first two lines create the IPS rule – you may associate and access-list with the IPS rule, to limit the scope of inspection traffic. The second line specifies the flash directory you created previously, to store the IPS configuration. The last two commands instruct the IPS to notify listeners connecting via SDEE for IPS monitoring and send IPS events into system log.

The next important step consists of retiring most IPS signatures. The issue is that the IPS package is so big, that the IOS will not be able to fit all signatures into memory, unless there is plenty of it (above a gigabyte). Thus, you need to enter the IPS configuration mode and retire most, if not all signatures initially. Retired signatures are not compiled in the memory; disabled signatures are compiled but not triggered. Thus it is important to retire and not simply disable the signatures. Use the following syntax:

Copyright © 2010 Internetwork Expert

```
ip ips signature-category
 category all
  retired true
  exit
 category ios_ips basic
  retired false
  exit
 exit
```

The above configuration will retire signatures bases on categories (logical grouping). Optionally, you may retire the all category and then selectively enable some signatures. Notice the use of the **exit** command – this syntax is the same as when configuring an IPS 5.x appliance via CLI.

The last thing you need to do in order to activate the IPS is apply the IPS rule to an interface. This will make the configuration active, even though there are no signatures loaded yet.

```
interface FastEthernet 0/0
  ip ips <NAME> in
```

You may choose either in or out directions to inspect ingress or egress traffic. Commonly, you need to inspect in both directions on the same interface, but you may apply the same rule to different interface in the same direction to have the same effect.

```
Loading IOS IPS signature package to router.
```

This step completes the IPS initialization. The signatures are loaded using special destination know as IDCONF. In order to complete the signature load, use the command **copy <URL> idconf**, e.g.

copy ftp://cisco:cisco@10.0.0.100/IOS-S310-CLI.pkg idconf

or

copy flash://IOS-S310-CLI.pkg idconf

This will initiate the compile process in the router memory. All retired signatures are ignored during the process. When the process is finished, use the command **show ip ips signature count** to check the statistics on the loaded signatures.

Copyright © 2010 Internetwork Expert

Tuning Signatures.

Basically, what you could do with the signatures is enabling or disabling them, changing event-action, alarm-severity and fidelity-rating attributes. You cannot tune the engine settings, not create custom signatures. Here is a sample list of steps that you need to do in order to tune a signature:

```
ip ips signature-definition
  signature 2004 0
    status
      enabled true
      retired false
    exit
    engine
      event-action produce-alert
    exit
    alert-severity high
    fidelity-rating 100
  exit
exit
```

The above sample shows all the basic settings you can change for a signature. Additionally, you may operate with signature categories, like we did when first loading the signature file. You may enable/disable/retire/unretired groups of signatures without having to deal with every one of them individually. Additionally, you may change the action for all signatures in a group at once, or change severify/fidelity.

```
ip ips signature-category
  category attack
    retired false
    event-action deny-attacker-inline
    enabled true
    exit
  exit
```

The above configuration will unretire/enable all “attack” signatures and specify a deny action for all of them. If you having hard time finding the right signature, use the IDM interface to lookup signature and/or signature categories based on name, ID, engine and so, as it’s hard to navigate through the IOS IPS configuration.

Copyright © 2010 Internetwork Expert

The last feature supported by IOS IPS, is a limited SEAP (Signature Event Action Processing) feature. Specifically, you may assign Target Value Rating to IP subnets using the following syntax:

```
ip ips event-action-rules
  target-value medium target-address 150.1.2.0/24
  exit
```

This will affect the risk-rating computation formula for attacks towards this IP subnet. However, there are no event-filter or event-overrides implemented in the IOS IPS yet, so those setting are only informational.

R4:

Note

Perform the steps required to initialize the IPS prior to starting the configuration part.

```
ip ips name MYIPS
ip ips config location flash:ips
ip ips notify sdee
ip ips notify log
!
interface FastEthernet 0/1
 ip ips MYIPS in
!
interface FastEthernet 0/0
 ip ips MYIPS in
!
ip ips signature-category
  category all
    retired true
    exit
  exit
!
```

Copyright © 2010 Internetwork Expert

```
ip ips signature-definition
  signature 2004 0
    status
      enabled true
      retired false
    exit
  exit
  signature 1004 0
   status
    enabled true
    retired false
    exit
   engine
     event-action deny-attacker-inline
     exit
  exit
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Start by checking the overall IPS configuration. After this, initiate ping across R6 and check R4's syslog for events:

```
Rack1R4#show ip ips all

IPS Signature File Configuration Status
    Configured Config Locations: flash:ips/
    Last signature default load time: 23:14:30 UTC Jun 27 2009
    Last signature delta load time: 23:21:59 UTC Jun 27 2009
    Last event action (SEAP) load time: -none-

    General SEAP Config:
    Global Deny Timeout: 3 seconds
    Global Overrides Status: Enabled
    Global Filters Status: Enabled

IPS Auto Update is not currently configured

IPS Syslog and SDEE Notification Status
    Event notification through syslog is enabled
    Event notification through SDEE is enabled

IPS Signature Status
    Total Active Signatures: 2
    Total Inactive Signatures: 2318

IPS Packet Scanning and Interface Status
    IPS Rule Configuration
      IPS name MYIPS
    IPS fail closed is disabled
    IPS deny-action ips-interface is false
    Interface Configuration
      Interface FastEthernet0/0
        Inbound IPS rule is MYIPS
        Outgoing IPS rule is not set
      Interface FastEthernet0/1
        Inbound IPS rule is MYIPS
        Outgoing IPS rule is not set

IPS Category CLI Configuration:
    Category all:
        Retire: True
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#ping 204.12.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/4 ms

%IPS-4-SIGNATURE: Sig:2004 Subsig:0 Sev:25 ICMP Echo Request
[155.1.146.1:8 -> 204.12.1.254:0] VRF:NONE RiskRating:25
%IPS-4-SIGNATURE: Sig:2004 Subsig:0 Sev:25 ICMP Echo Request
[155.1.146.1:8 -> 204.12.1.254:0] VRF:NONE RiskRating:25
%IPS-4-SIGNATURE: Sig:2004 Subsig:0 Sev:25 ICMP Echo Request
[155.1.146.1:8 -> 204.12.1.254:0] VRF:NONE RiskRating:25
%IPS-4-SIGNATURE: Sig:2004 Subsig:0 Sev:25 ICMP Echo Request
[155.1.146.1:8 -> 204.12.1.254:0] VRF:NONE RiskRating:25
%IPS-4-SIGNATURE: Sig:2004 Subsig:0 Sev:25 ICMP Echo Request
[155.1.146.1:8 -> 204.12.1.254:0] VRF:NONE RiskRating:25
```

System Management Solutions

> **Note**
>
> Erase and reload all devices and load the *System Management* initial configurations prior to starting.

12.1 Exec Aliases

- Configure aliases on R1 to accomplish this following:
  - Issuing the command **ri** in exec mode should view the routing table contents, while **rb** views the BGP table.
  - Issuing **i Fa0/0** in global configuration should switch to interface FastEthernet 0/0 configuration mode.
  - Issuing **iea ACL** in global configuration should create the extended access-list named ACL.
  - Issuing **so** and **si** should apply a policy-map to an interface either outbound or inbound respectively.

Configuration

```
R1:
alias interface si service-policy input
alias interface so service-policy output
alias configure i interface
alias configure iae ip access-list extended
alias exec ri show ip route
alias exec rb show ip bgp
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Command aliases in IOS can be used as shortcuts to speed up and simplify repetitive configurations. Aliases can be configured for other parser modes, such as route-map mode or class-map mode, but the most common ones are exec, global configuration, and interface mode. These configurations can be verified as follows:

```
Rack1R1(config)#i fastEthernet 0/0
Rack1R1(config-if)#exit
Rack1R1(config)#iae ACL
Rack1R1(config-ext-nacl)#exit
Rack1R1(config)#iae 100
Rack1R1(config-ext-nacl)#exit
Rack1R1(config)#i fastEthernet 0/0
Rack1R1(config-if)#so TEST
% policy map TEST not configured
Rack1R1(config-if)#si TEST
% policy map TEST not configured

Rack1R1#ri
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

R    222.22.2.0/24 [120/9] via 155.1.13.3, 00:00:03, Serial0/1
                   [120/9] via 155.1.0.2, 00:00:03, Serial0/0
R    204.12.1.0/24 [120/1] via 155.1.146.4, 00:00:00, FastEthernet0/0
     155.1.0.0/24 is subnetted, 15 subnets
C       155.1.146.0 is directly connected, FastEthernet0/0
R       155.1.23.0 [120/1] via 155.1.13.3, 00:00:03, Serial0/1
R       155.1.10.0 [120/3] via 155.1.0.5, 00:00:03, Serial0/0
R       155.1.8.0 [120/2] via 155.1.0.5, 00:00:03, Serial0/0
R       155.1.9.0 [120/3] via 155.1.146.6, 00:00:04, FastEthernet0/0
C       155.1.13.0 is directly connected, Serial0/1
C       155.1.0.0 is directly connected, Serial0/0
R       155.1.7.0 [120/2] via 155.1.146.6, 00:00:04, FastEthernet0/0
R       155.1.5.0 [120/1] via 155.1.0.5, 00:00:04, Serial0/0
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#ri 222.22.2.0
Routing entry for 222.22.2.0/24
  Known via "rip", distance 120, metric 9
  Redistributing via rip
  Last update from 155.1.13.3 on Serial0/1, 00:00:03 ago
  Routing Descriptor Blocks:
  * 155.1.13.3, from 155.1.13.3, 00:00:03 ago, via Serial0/1
      Route metric is 9, traffic share count is 1
    155.1.0.2, from 155.1.0.5, 00:00:05 ago, via Serial0/0
      Route metric is 9, traffic share count is 1

Rack1R1#rb
% BGP not active
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.2 System Message Logging

- Enable system message logging on R4 and R6 as follows:
  - Both routers should save debugging messages to their internal buffers up to 8192 bytes.
  - Debugging messages should be sent to the router consoles, but limited to 1 message per second.
  - Console log messages should not interrupt other command output.
  - Users logged via telnet should only see informational level messages and above.

Configuration

```
R4:
logging on
logging buffered 8192 debugging
logging console debugging
logging rate-limit console all 1
logging monitor informational
!
line console 0
 logging synchronous

R6:
logging on
logging buffered 8192 debugging
logging console debugging
logging rate-limit console all 1
logging monitor informational
!
line console 0
 logging synchronous
```

Verification

Note

Logging messages can be sent to multiple destinations simultaneously, including the console/terminal lines, the internal buffer, and a remote syslog server. Some platforms also allow storing logging messages in the local flash memory.

For each destination you can configure the message severity level, which controls which messages are logged. Logging at severity 7 (debugging) includes all messages 0 through 7. Logging at severity 4 includes all messages 0 through 4.

Copyright © 2010 Internetwork Expert

For syslog the logging facility is also available, which controls the format of the log message. Using different facilities for different devices, for example local5 for all routers and local6 for all switches, can make sorting through log files on the syslog server itself more manageable.

Synchronous logging on a terminal line makes the logging process wait for the terminal to finish printing a line before outputting its own message.

Logging rate-limiting shrinks the number of messages (actually message lines) sent per second either to all destinations or to the console line specifically. Note that the RIP debugging output below only prints a limited amount of messages to the console (due to rate limiting enabled), while the logging buffer contains full messages.

```
Rack1R4#debug ip rip
RIP protocol debugging is on
RIP: received v2 update from 155.1.45.5 on Serial0/1
RIP: sending v2 update to 224.0.0.9 via Ethernet0/1 (155.1.146.4)
RIP: received v2 update from 155.1.0.5 on Serial0/0
RIP: sending v2 update to 224.0.0.9 via Serial0/0 (155.1.0.4)
RIP: received v2 update from 155.1.45.5 on Serial0/1
RIP: sending v2 update to 224.0.0.9 via Loopback0 (150.1.4.4)
RIP: sending v2 update to 224.0.0.9 via Serial0/1 (155.1.45.4)
RIP: sending v2 update to 224.0.0.9 via Ethernet0/0 (204.12.1.4)
RIP: received v2 update from 155.1.146.6 on Ethernet0/1
RIP: received v2 update from 155.1.45.5 on Serial0/1
RIP: sending v2 update to 224.0.0.9 via Loopback0 (150.1.4.4)
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show logging
Syslog logging: enabled (11 messages dropped, 1148 messages rate-limited,
                0 flushes, 0 overruns, xml disabled, filtering disabled)
    Console logging: level debugging, 63 messages logged, xml disabled,
                     filtering disabled
    Monitor logging: level informational, 0 messages logged, xml disabled,
                     filtering disabled
    Buffer logging: level debugging, 1177 messages logged, xml disabled,
                    filtering disabled
    Logging Exception size (4096 bytes)
    Count and timestamp logging messages: disabled

No active filter modules.

    Trap logging: level informational, 63 message lines logged

Log Buffer (8192 bytes):
0, metric 4, tag 0
       155.1.0.0/24 via 0.0.0.0, metric 1, tag 0
       155.1.5.0/24 via 0.0.0.0, metric 2, tag 0
       155.1.7.0/24 via 0.0.0.0, metric 3, tag 0
       155.1.8.0/24 via 0.0.0.0, metric 3, tag 0
       155.1.9.0/24 via 0.0.0.0, metric 4, tag 0
       155.1.10.0/24 via 0.0.0.0, metric 4, tag 0
       155.1.13.0/24 via 0.0.0.0, metric 2, tag 0
       155.1.23.0/24 via 0.0.0.0, metric 3, tag 0
       155.1.37.0/24 via 0.0.0.0, metric 3, tag 0
       155.1.45.0/24 via 0.0.0.0, metric 1, tag 0
       155.1.58.0/24 via 0.0.0.0, metric 2, tag 0
       155.1.67.0/24 via 0.0.0.0, metric 2, tag 0
       155.1.79.0/24 via 0.0.0.0, metric 3, tag 0
       155.1.108.0/24 via 0.0.0.0, metric 3, tag 0
RIP: build update entries
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.3 Syslog Logging

- Configure R4 and R6 to log messages to syslog as follows:
  - Both R4 and R6 should log to the server 155.X.146.100.
  - Log all messages up to notifications.
  - R4 should use reliable transport at port 5000.
  - R6 should use the default transport.
  - Use IDs ROUTER4 and ROUTER6 respectively and the UNIX facility LOCAL1.
  - Messages should be sourced off of the routers' Loopback0 interfaces.
  - Set the message queue depth to 256.

Configuration

```
R4:
logging queue-limit trap 256
logging origin-id string ROUTER4
logging facility local1
logging trap notifications
logging source-interface Loopback0
logging host 155.1.146.100 transport tcp port 5000

R6:
logging queue-limit trap 256
logging origin-id string ROUTER6
logging facility local1
logging trap notifications
logging source-interface Loopback0
logging host 155.1.146.100
```

Verification

Note

IOS supports both TCP and UDP to transport syslog messages. UDP is the default, but does not have reliable delivery like TCP does. The UNIX facility LOCAL1 allows the syslog server to multiplex incoming messages. Multiple logging destinations can be configured using their own transport, but all servers share the same facility and trap level.

Copyright © 2010 Internetwork Expert

```
Rack1R6#show logging
Syslog logging: enabled (11 messages dropped, 0 messages rate-limited,
                0 flushes, 0 overruns, xml disabled, filtering disabled)
    Console logging: level notifications, 24 messages logged, xml disabled,
                     filtering disabled
    Monitor logging: level debugging, 0 messages logged, xml disabled,
                     filtering disabled
    Buffer logging: disabled, xml disabled,
                    filtering disabled
    Logging Exception size (4096 bytes)
    Count and timestamp logging messages: disabled

No active filter modules.

    Trap logging: level notifications, 59 message lines logged
        Logging to 155.1.146.100(global) (udp port 514, audit disabled,  link
up), 2 message lines logged, xml disabled,
             filtering disabled
```

To trace the packets sent to the syslog server configure IP packet debugging for an access-list matching the syslog UDP packets. To generate a notification log go to global configuration and then type **exit**.

```
Rack1R6(config)#access-list 100 permit udp any any eq 514
Rack1R6#debug ip packet detail 100
IP packet debugging is on (detailed) for access list 100

Rack1R6#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#exit
Rack1R6#
%SYS-5-CONFIG_I: Configured from console by console
IP: tableid=0, s=150.1.6.6 (local), d=155.1.146.100
(FastEthernet0/0.146), routed via RIB
```

Copyright © 2010 Internetwork Expert

12.4 Logging Counting and Timestamps

- Configure R4 and R6 to count error messages.
- R4 should insert uptime-based timestamps in log messages, while R6 should insert date-time based timestamps.
- R6 should add milliseconds based timestamps to debugging messages only.
- Include year information in logging messages on R6 but not in debugging messages.
- Provide a solution to prevent against tampering with stored syslog information.

Configuration

```
R4:
service timestamp debug uptime
service timestamps log uptime
service sequence-numbers
logging count

R6:
service timestamps debug datetime msec
service timestamps log datetime year
service sequence-numbers
logging count
```

Verification

Note

Timestamps can be configured separately for debugging and logging messages. They can be based on either router's uptime, or the current date and time. In the latter case, millisecond resolution can also be configured.

In the output below note that debugging message and logging output (e.g. system events) use different timestamps (logging messages have year information).

Copyright © 2010 Internetwork Expert

```
Rack1R6#debug ip rip
RIP protocol debugging is on
Rack1R6#
000081: Jul XX 23:25:54.920: RIP: ignored v2 update from bad source
54.1.3.254 on Serial0/0
000136: Jul XX 23:25:56.535: RIP: sending v2 update to 224.0.0.9 via
FastEthernet0/0.67 (155.1.67.6)

Rack1R6#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#exit
Rack1R6#
001127: Jul XX 2008 23:32:36: %SYS-5-CONFIG_I: Configured from console
by console
```

Compare the above debugging output with the debugging output on R4, which has timestamps based on uptime.

```
Rack1R4#debug ip rip

002601: 02:55:20: RIP: sending v2 update to 224.0.0.9 via Ethernet0/1
(155.1.146.4)
002629: 02:55:22: RIP: received v2 update from 204.12.1.254 on
Ethernet0/0
002669: 02:55:23: RIP: sending v2 update to 224.0.0.9 via Loopback0
(150.1.4.4)
002736: 02:55:24: RIP: sending v2 update to 224.0.0.9 via Serial0/0
(155.1.0.4)
```

Note that all above debugging/logging lines have sequence numbers. This helps preventing tampering with the stored syslog message. Error messages counting allows to count all “notification” and above system message to provide statistics for quick analysis of the system history.

Copyright © 2010 Internetwork Expert

```
Rack1R6#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface fastEthernet 0/0
Rack1R6(config-if)#shutdown
Rack1R6(config-if)#no shut
Rack1R6(config-if)#
Rack1R6(config-if)#^Z

Rack1R6#show logging count

Facility       Message Name                    Sev Occur      Last Time
================================================================================
SYS            CONFIG_I                          5     4 Jul XX 2008 23:39:00
-------------  ------------------------------  ------------------------------
SYS TOTAL                                              4

LINEPROTO      UPDOWN                            5     2 Jul XX 2008 23:38:37
-------------  ------------------------------  ------------------------------
LINEPROTO TOTAL                                        2

LINK           UPDOWN                            3     1 Jul XX 2008 23:38:36
LINK           CHANGED                           5     1 Jul XX 2008 23:38:28
-------------  ------------------------------  ------------------------------
LINK TOTAL                                             2
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.5 Logging to Flash Memory

- Configure SW1 to log notification messages in its internal flash memory.
- Create the directory /var/log and store messages in the file named "syslog".
- Set the maximum file size to 32768 bytes.

Configuration

```
Rack1SW1#mkdir flash:/var
Create directory filename [var]?
Created dir flash:/var

Rack1SW1#mkdir flash:/var/log
Create directory filename [/var/log]?
Created dir flash:/var/log

SW1:
logging file flash:/var/log/syslog 32768 notifications
logging on
```

Verification

> **Note**
>
> From the equipment currently available in the lab exam only the Catalyst switches support the storing of syslog messages in flash memory.

```
Rack1SW1#show logging
Syslog logging: enabled (0 messages dropped, 1 messages rate-limited, 0
flushes, 0 overruns, xml disabled, filtering disabled)
    Console logging: level notifications, 74 messages logged, xml disabled,
                     filtering disabled
    Monitor logging: level debugging, 0 messages logged, xml disabled,
                     filtering disabled
    Buffer logging: level debugging, 157 messages logged, xml disabled,
                    filtering disabled
    Exception Logging: size (4096 bytes)
    Count and timestamp logging messages: disabled
    File logging: file flash:/var/log/syslog,
        max size 32768, min size 0,
        level notifications, 2 messages logged
    Trap logging: level informational, 160 message lines logged

Log Buffer (4096 bytes):
<snip>

Rack1SW1#more flash:/var/log/syslog
03:54:30: %SYS-5-CONFIG_I: Configured from console by console
03:55:56: %SYS-5-CONFIG_I: Configured from console by console
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.6 Lonfiguration Change Notification and Logging

- Configure R4 to locally track changes made to its running configuration.
- Log these changes via syslog, but ensure passwords in the configuration will not be sent across this communication channel.
- Set the queue size to 1000.

Configuration

```
R4:
archive
 log config
  logging enable
  logging size 1000
  notify syslog
  hidekeys
```

Verification

> **Note**
>
> Change notification and logging is simple alternative to AAA accounting. This feature allows you to track configuration changes along with the user who made them. The primary purpose is to log the changes to a remote syslog server for further archiving and analysis, but it's also possible to review the local changes queue.

```
Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface FastEthernet 0/0
004299: 02:36:49: %PARSER-5-CFGLOG_LOGGEDCMD: User:console  logged
command:interface FastEthernet0/0
Rack1R4(config-if)#shutdown
004300: 02:36:53: %PARSER-5-CFGLOG_LOGGEDCMD: User:console  logged
command:shutdown
Rack1R4(config-if)#no shutdown
004301: 02:36:56: %PARSER-5-CFGLOG_LOGGEDCMD: User:console  logged
command:no shutdown
004302: 02:36:58: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed
state to up
004303: 02:37:00: %LINEPROTO-5-UPDOWN: Line protocol on Interface
FastEthernet0/0, changed state to up
```

Copyright © 2010 Internetwork Expert

It's possible to review the changes made by all users, or just by a specified user. The statistics option displays the number of user sessions tracked and the memory usage.

```
Rack1R4#show archive log config all
 idx    sess           user@line      Logged command
    1       1        console@console  | logging enable
    2       1        console@console  |do sh run | beg arch
    3       2        console@console  |interface FastEthernet0/0
    4       2        console@console  | shutdown
    5       2        console@console  | no shutdown

Rack1R4#show archive log config statistics
Config Log Session Info:
       Number of sessions being tracked: 1
       Memory being held: 3910 bytes
       Total memory allocated for session tracking: 3910 bytes
       Total memory freed from session tracking: 0 bytes

Config Log log-queue Info:
       Number of entries in the log-queue: 5
       Memory being held by the log-queue: 1089 bytes
       Total memory allocated for log entries: 1089 bytes
       Total memory freed from log entries: 0 bytes
```

Additionally it's possible to display the changes in a format suitable to direct application to a running IOS router (e.g. to replay the changes).

```
Rack1R4#show archive log config all provisioning
logging enable
do sh run | beg arch
interface FastEthernet0/0
 shutdown
 no shutdown
```

Copyright © 2010 Internetwork Expert

12.7 Configuration Archive & Rollback

- Configure SW1 to store its configuration archive on the TFTP server 155.X.58.100 using “sw1-config” as the prefix.
- Archive the configuration every 24 hours.
- Additionally the configuration should be archived every time the running config is saved to NVRAM.

Configuration

```
SW1:
archive
 path tftp://155.1.58.100/sw1-config
 write-memory
 time-period 1440
```

Verification

> **Note**
>
> A router can save its configuration snapshots to a network path or to a secondary non-volatile storage plugged into router (e.g. disk0), but not to its primary flash memory storage. When saving the configuration to a network path the maximum number of copies can not be specified.

```
Rack1SW1#show archive
The next archive file will be named tftp://155.1.58.100/sw1-config -1
 Archive #  Name
   0
   1
   2
   3
   4
   5
   6
   7
   8
   9
   10
   11
   12
   13
   14
```

Copyright © 2010 Internetwork Expert

Configuration archiving is very useful in combination with two other features, configuration rollback and contextual diff utility. To see how they work manually create a copy of the running-configuration in flash, then make some changes to the running configuration.

```
Rack1SW1#copy running-config flash:/saved-config
Destination filename [saved-config]?

2582 bytes copied in 1.241 secs (2081 bytes/sec)

Rack1SW1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#int vlan 67
Rack1SW1(config-if)#shutdown
Rack1SW1(config-if)#ip address 155.1.76.7 255.255.255.0
```

Now list the differences between the two configurations, the saved one and the running one. The “+” sign means that the configuration is present in the second configuration but missing in the first file. The “-“ sign means the configuration line is present in the first file but is missing in the second one.

```
Rack1SW1#show archive config differences flash:/saved-config
system:/running-config
Contextual Config Diffs:
interface Vlan67
 +ip address 155.1.76.7 255.255.255.0
 +shutdown
interface Vlan67
 -ip address 155.1.67.7 255.255.255.0
```

The below output displays the lines that need to be added to the running configuration in order to make it look like the saved configuration.

```
Rack1SW1#show archive config incremental-diffs flash:/saved-config
!List of Commands:
interface Vlan67
 ip address 155.1.67.7 255.255.255.0
end
```

Copyright © 2010 Internetwork Expert

Use the **configure replace** command to roll-back the running configuration to the configuration stored in flash. This procedure will ensure the configurations are identical, and will even un-shutdown the interfaces.

```
Rack1SW1#configure replace flash:/saved-config list force
!Pass 1

!List of Commands:
interface Vlan67
 no ip address 155.1.76.7 255.255.255.0
 no shutdown
interface Vlan67
 ip address 155.1.67.7 255.255.255.0
end


Total number of passes: 1
Rollback Done
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.8 Logging with Access-Lists

- Configure R4 to log every RIP packet entering the interface connected to VLAN 146.
- Generate a cumulative log entry for every 2 matched packets.
- Send a logging message no more than once per 10 seconds.
- Log the MAC address of the device sending the packet.

Configuration

```
R4:
ip access-list extended LOGGING
 permit udp any any eq rip log-input
 permit ip any any
!
interface FastEthernet 0/1
 ip access-group LOGGING in
!
ip access-list logging interval 10
ip access-list log-update threshold 2
```

Verification

Note

The log-input option logs the MAC address of the device that sent the packet in addition to the packet information. The update threshold set the minimal number of hits that are needed to generate a logging message, and effectively aggregates the hits. The logging interval specifies the minimal interval of log messages generation, essentially providing a rate-limiting capability.

In the below output R1's connection to VLAN 146 is disabled to show only the updates that R4 receives from R6. Note that a log message is generated approximately every 20 seconds, as R6 is configured to send updates every 10 seconds, and the logging configuration aggregates the ACL hits in pairs.

Copyright © 2010 Internetwork Expert

```
Rack1R1#conf t
Rack1R1(config)#inteface FastEthernet 0/0
Rack1R1(config-if)#shutdown

Rack1R4
004318: 03:57:24: %SEC-6-IPACCESSLOGP: list LOGGING permitted udp
155.1.146.6(520) (FastEthernet0/1 0011.9283.0400) -> 224.0.0.9(520), 2
packets

004319: 03:57:43: %SEC-6-IPACCESSLOGP: list LOGGING permitted udp
155.1.146.6(520) (FastEthernet0/1 0011.9283.0400) -> 224.0.0.9(520), 2
packets

004320: 03:58:03: %SEC-6-IPACCESSLOGP: list LOGGING permitted udp
155.1.146.6(520) (FastEthernet0/1 0011.9283.0400) -> 224.0.0.9(520), 2
packets

004321: 03:58:21: %SEC-6-IPACCESSLOGP: list LOGGING permitted udp
155.1.146.6(520) (FastEthernet0/1 0011.9283.0400) -> 224.0.0.9(520), 2
packets
```

Now change the update count to 1 so that every packet hit generates a message. Note that logging messages are generated every 10 seconds now.

```
Rack1R4(config)#ip access-list log-update threshold 1

004330: 03:59:43: %SEC-6-IPACCESSLOGP: list LOGGING permitted udp
155.1.146.6(520) (FastEthernet0/1 0011.9283.0400) -> 224.0.0.9(520), 1
packet

004331: 03:59:52: %SEC-6-IPACCESSLOGP: list LOGGING permitted udp
155.1.146.6(520) (FastEthernet0/1 0011.9283.0400) -> 224.0.0.9(520), 1
packet

004332: 04:00:01: %SEC-6-IPACCESSLOGP: list LOGGING permitted udp
155.1.146.6(520) (FastEthernet0/1 0011.9283.0400) -> 224.0.0.9(520), 1
packet

004333: 04:00:11: %SEC-6-IPACCESSLOGP: list LOGGING permitted udp
155.1.146.6(520) (FastEthernet0/1 0011.9283.0400) -> 224.0.0.9(520), 1
packet
```

Copyright © 2010 Internetwork Expert

12.9 TCP Keepalives

- Configure R1 so that it's capable of detecting dead TCP connections and closing them prematurely.

Configuration

```
R1:
service tcp-keepalives-out
service tcp-keepalive-in
```

Verification

> **Note**
>
> The TCP keepalive feature is useful for probing idle connection to see if they are still active. In some cases, such as when the exec-timeout feature is disabled on VTY lines, this helps to prevent resources starvation by freeing connections that are not active anymore.
>
> To test this, initiate a connection from R6 to R1 and verify its parameters. Note the keepalive fields in the TCP Connection Block output.

```
Rack1R6#telnet 155.1.146.1
Trying 155.1.146.1 ... Open

User Access Verification

Password:
Rack1R1>

Rack1R1#show tcp brief all
TCB        Local Address               Foreign Address         (state)
849905B4   155.1.146.1.23              155.1.146.6.17316       ESTAB
8551BA24   *.80                        *.*                     LISTEN

Rack1R1#show tcp tcb 849905B4
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled, Mininum incoming TTL 0, Outgoing TTL 255
Local host: 155.1.146.1, Local port: 23
Foreign host: 155.1.146.6, Foreign port: 17316

Enqueued packets for retransmit: 0, input: 0  mis-ordered: 0 (0 bytes)

Event Timers (current time is 0x1BE3ED2):
Timer          Starts    Wakeups            Next
Retrans             4          0             0x0
TimeWait            0          0             0x0
AckHold             5          2             0x0
SendWnd             0          0             0x0
KeepAlive           7          0       0x1BECF6D
```

```
GiveUp              0          0             0x0
PmtuAger            0          0             0x0
DeadWait            0          0             0x0

iss: 3817118842  snduna: 3817118922  sndnxt: 3817118922     sndwnd:
4049
irs: 2400837743  rcvnxt: 2400837781  rcvwnd:       4091  delrcvwnd:
37

SRTT: 124 ms, RTTO: 1405 ms, RTV: 1281 ms, KRTT: 0 ms
minRTT: 0 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: passive open, active open, retransmission timeout,
  keepalive running
IP Precedence value : 0

Datagrams (max data segment is 1460 bytes):
Rcvd: 11 (out of order: 0), with data: 7, total data bytes: 37
Sent: 10 (retransmit: 0, fastretransmit: 0, partialack: 0, Second
Congestion: 0), with data: 7, total data bytes: 79
Rack1R1#
```

Debug TCP transactions on R1, then shut down the switch-port connected to R6 to simulate a dead connection. Note that after four keepalives have been missed R1 forcefully closes the idle connection.

```
Rack1R1#debug ip tcp transactions

Rack1SW2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW2(config)#int fastEthernet 0/6
Rack1SW2(config-if)#

Rack1R1#
TCP66: keepalive timeout (0/4)
Rack1R1#
TCP66: keepalive timeout (1/4)
Rack1R1#
TCP66: keepalive timeout (2/4)
Rack1R1#
TCP66: keepalive timeout (3/4)
Rack1R1#
TCP66: keepalive timeout (4/4)
TCP66: state was ESTAB -> CLOSED [23 -> 155.1.146.6(17316)]
TCB 0x849905B4 destroyed
```

Copyright © 2010 Internetwork Expert

12.10 Generating Exception Core Dumps

- Configure R3 to save core dumps to an FTP server at 155.X.146.100 under the name “r3-core”.
- Use active FTP and the username/password values cisco/cisco.
- Configure the router to generate a memory dump and reload as soon as free memory falls below 1Mbyte.
- The router should also reload in the case that memory fragmentation prohibits a process from allocating more than 64Kbytes of memory.
- Disable the local crash information collection.

Configuration

```
R3:
exception core-file r3-core
exception protocol ftp
exception dump 155.1.146.100
exception memory fragment 64000 reboot
exception memory minimum 1000000 reboot
!
no ip ftp passive
ip ftp username cisco
ip ftp password cisco
!
no exception crashinfo
```

Copyright © 2010 Internetwork Expert

Verification

Note

The exception feature allows for the generation of memory dumps and collecting crash information in case of emergency conditions. Memory dumps are usually stored at an outside server and delivered using FTP, TFTP, or the RCP protocol. By default, crash information is stored in local the flash memory. In this case it is explicitly disabled with the **`no exception crashinfo`** command. Special conditions may also be configured to generate exceptions and reload the router, such as when memory fragmentation occurs. By default, reaching the threshold would force a software crash. Using the "reboot" keyword you can force the router to simply reload without generating any crashinfo.

When using FTP to deliver the crash dump, configure the username and password using the IOS FTP client parameters. To verify this configuration issue the **`show exception`** and **`write core`** commands.

```
Rack1R3#show exception
155.1.146.100
```

Copyright © 2010 Internetwork Expert

12.11 Conditional Debugging

- Configure R6 to debug RIP updates entering and leaving the connection to VLAN 67 only.

Configuration

```
Rack1R6#debug condition interface fastEthernet 0/0.67
Rack1R6#debug ip rip
```

Verification

Note

Conditional debugging is an extremely useful tool to limit the amount of information on busy routers. Not only is it possible to match interface names, but calling numbers (e.g. for ISDN access-servers) or usernames (e.g. for PPP sessions) are also supported. Note that the **`undebug all`** command does not remove the debug conditions. It is required to disable the conditions using explicit command syntax.

Observe the debugging output on R6 with the condition set to match the VLAN 67 interface.

```
Rack1R6#
001525: Jul XX 00:00:11.026: RIP: received v2 update from 155.1.67.7 on
FastEthernet0/0.67
Rack1R6#
001531: Jul XX 00:00:15.061: RIP: sending v2 update to 224.0.0.9 via
FastEthernet0/0.67 (155.1.67.6)
Rack1R6#
001571: Jul XX 00:00:20.582: RIP: received v2 update from 155.1.67.7 on
FastEthernet0/0.67
Rack1R6#
001577: Jul XX 00:00:24.168: RIP: sending v2 update to 224.0.0.9 via
FastEthernet0/0.67 (155.1.67.6)
```

Disable the condition on R6 and see how the number of debugging messages magnifies.

Copyright © 2010 Internetwork Expert

```
Rack1R6#undebug condition interface fastEthernet 0/0.67
This condition is the last interface condition set.
Removing all conditions may cause a flood of debugging
messages to result, unless specific debugging flags
are first removed.

Proceed with removal? [yes/no]: yes
Condition 1 has been removed

Rack1R6#
004607: Jul 17 00:10:29.857: RIP: sending v2 update to 224.0.0.9 via
Serial0/0 (54.1.1.6)

004653: Jul 17 00:10:30.462: RIP: sending v2 update to 224.0.0.9 via
Loopback0 (150.1.6.6)

004700: Jul 17 00:10:32.445: RIP: received v2 update from 155.1.67.7 on
FastEthernet0/0.67

004706: Jul 17 00:10:33.479: RIP: sending v2 update to 224.0.0.9 via
FastEthernet0/0.146 (155.1.146.6)
```

Copyright © 2010 Internetwork Expert

12.12 Telnet Service Options

- Configure R3 to source all telnet sessions from its Loopback0 interface, and to use a marking of IP Precedence 3 for these packets.
- Do not display any telnet informational messages.
- Create a hostname entry “R4” that pointing to the IP address of R4.
- Ensure that a user typing “R4” in exec mode is connected without seeing the IP address of R4.
- Idle outgoing telnet sessions should signal the remote host to pause output.
- Display the message “Sorry, your connection failed” when a telnet connection to the above host fails.

Configuration

```
R3:
service telnet-zeroidle
ip telnet source-interface Loopback0
ip telnet tos 60
ip telnet quiet
ip telnet hidden addresses
!
no ip domain-lookup
ip host R4 155.1.146.4
!
busy-message R4 # Sorry, your connection failed #
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> The telnet client in IOS can be tuned to provide additional “obfuscation” settings, which hides information from the user initiating the session. Note how the below changes once the “quiet” and “hidden address” features are disabled. With both features on, telnet never displays any progress or informational messages, and does not report the IP address of the host it connects to.

```
Rack1R3#R4
Translating "R4"

User Access Verification

Password: cisco
Rack1R4>exit

Rack1R3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#no ip telnet quiet
Rack1R3(config)#do telnet R4
Trying R4 address #1 ... Open


User Access Verification

Password: cisco
Rack1R4>exit

Rack1R3(config)#no ip telnet hidden address
Rack1R3(config)#do telnet R4
Trying R4 (155.1.146.4)... Open


User Access Verification

Password: cisco
Rack1R4>
```

Copyright © 2010 Internetwork Expert

To verify the TOS byte settings are correct connect to R4 from R3, then telnet back and verify the TCP connection properties. Note that the TOS value is entered in HEX format in the configuration.

```
Rack1R3#R4
Translating "R4"
Trying R4 (155.1.146.4)... Open


User Access Verification

Password: cisco
Rack1R4>telnet 150.1.3.3
Trying 150.1.3.3 ... Open


User Access Verification

Password: cisco
Rack1R3>en
Password: cisco

Rack1R3#show tcp brief all
TCB        Local Address               Foreign Address         (state)
64FEBBC0   150.1.3.3.22596             R4.23                   ESTAB
64FEC074   150.1.3.3.23                R4.52354                ESTAB
64F300E0   *.80                        *.*                     LISTEN

Rack1R3#show tcp tcb 64FEBBC0
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled, Mininum incoming TTL 0, Outgoing TTL 255
Local host: 150.1.3.3, Local port: 22596
Foreign host: 155.1.146.4, Foreign port: 23

<snip>

SRTT: 300 ms, RTTO: 304 ms, RTV: 4 ms, KRTT: 0 ms
minRTT: 16 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: active open, retransmission timeout
IP Precedence value : 3

Datagrams (max data segment is 536 bytes):
Rcvd: 92 (out of order: 0), with data: 86, total data bytes: 673
Sent: 95 (retransmit: 0, fastretransmit: 0, partialack: 0, Second
Congestion: 0), with data: 54, total data bytes: 125
```

Copyright © 2010 Internetwork Expert

To verify the busy message settings shutdown the interface connecting to VLAN 146 on R4 and then try connecting to R4 from R3. Note that the busy-message can be set globally as well as per VTY line, and is always bound to a target host-name.

```
Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface FastEthernet 0/1
Rack1R4(config-if)#shutdown

Rack1R3#R4
Translating "R4"
 Sorry, your connection failed
```

With the zero-idle feature an idle session will advertise a windows size of zero to the remote peer. The result of this is that all output collected at the remote side will be paused and buffered until the user resumes the session. To test this generate logging output on R6 as follows.

```
Rack1R3#telnet 150.1.6.6
Trying 150.1.6.6 ... Open

User Access Verification

Password:
Rack1R6>en
Password: cisco

Rack1R6#debug ip rip
RIP protocol debugging is on
Rack1R6#terminal monitor
Rack1R6#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#logging monitor debugging
Rack1R6(config)#
005493: Jul XX 07:13:30.435: RIP: sending v2 update to 224.0.0.9 via
FastEthernet0/0.146 (155.1.146.6)
005494: Jul XX 07:13:30.435: RIP: build update entries
```

At this moment go back to R6's console and disable all debugging. Next, resume the initiated from R3 and you should see all the output collected displayed at once on the screen.

```
Rack1R6#
006485: Jul XX 07:14:17.175:       212.18.1.0/24 via 0.0.0.0, metric 2, tag 0
006486: Jul XX 07:14:17.175:       212.18.2.0/24 via 0.0.0.0, metric 2, tag 0
006487: Jul XX 07:14:17.175:       212.18.3.0/24 via 0.0.0.0, metric 2, tag 0
<snip>
```

Copyright © 2010 Internetwork Expert

12.13 Tuning Packet Buffers

- Enable automatic buffer tuning on R3.
- Configure R4 to double the permanent number of all public (not interface specific) buffers.
- Ensure there are at least 10 Large and Huge buffers always available.

Configuration

```
R3:
buffers tune automatic

R4:
buffers small permanent 100
buffers middle permanent 50
buffers big permanent 100
buffers verybig permanent 20
buffers large permanent 10
buffers huge permanent 10
```

Verification

Note

Buffers represent chunks of space in the router's I/O memory. Each interface may have its own private buffer pool, and all interfaces have access to public buffer pools. Buffers are used to store incoming packets, but since packets may vary in size, different group of buffer sizes exist. Having dynamically sized buffers is inefficient in terms of speed and memory fragmentation. Having pre-allocated buffers may speed-up incoming packets processing, but consumes additional memory. Therefore tuning the buffer space is a trade-off between speed and resource usage.

Sometimes it is necessary to tune buffers manually based on their historical demand. However with recent IOS versions it is possible to enable automatic buffer tuning, which changes the number of pre-allocated buffers based on an adaptive learning algorithm.

Note that for each buffer group it is possible to specify the upper and lower bounds of the buffer count, as well as the permanent and initial buffer allocation. Use the **`show buffer`** command before applying any new settings to observe the historical statistics and the current settings. A buffer "hit" mean a buffer was available for use when a packet arrived, while a "miss" means the IOS had to allocate a new buffer on demand for the packet.

Copyright © 2010 Internetwork Expert

```
Rack1R4#show buffers
Buffer elements:
     1118 in free list (1119 max allowed)
     79500 hits, 0 misses, 619 created

Public buffer pools:
Small buffers, 104 bytes (total 50, permanent 50):
     49 in free list (20 min, 150 max allowed)
     28487 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Middle buffers, 600 bytes (total 25, permanent 25, peak 31 @ 09:48:39):
     25 in free list (10 min, 150 max allowed)
     84985 hits, 2 misses, 6 trims, 6 created
     0 failures (0 no memory)
Big buffers, 1536 bytes (total 50, permanent 50):
     50 in free list (5 min, 150 max allowed)
     1595 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
VeryBig buffers, 4520 bytes (total 10, permanent 10):
     10 in free list (0 min, 100 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Large buffers, 5024 bytes (total 0, permanent 0):
     0 in free list (0 min, 10 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Huge buffers, 18024 bytes (total 0, permanent 0):
     0 in free list (0 min, 4 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
```

The below output shows the buffer sizes after they been changed on R4.

```
Rack1R4#show buffers
Buffer elements:
     1118 in free list (1119 max allowed)
     80566 hits, 0 misses, 619 created

Public buffer pools:
Small buffers, 104 bytes (total 100, permanent 100):
     99 in free list (20 min, 150 max allowed)
     29016 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Middle buffers, 600 bytes (total 50, permanent 50):
     50 in free list (10 min, 150 max allowed)
     86380 hits, 2 misses, 6 trims, 6 created
     0 failures (0 no memory)
Big buffers, 1536 bytes (total 100, permanent 100):
     100 in free list (5 min, 150 max allowed)
     1620 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
VeryBig buffers, 4520 bytes (total 20, permanent 20):
     20 in free list (0 min, 100 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
<snip>
```

Copyright © 2010 Internetwork Expert

12.14 Terminal Line Settings

- Configure R3 to allow only telnet connections to VTY lines 0 through 4 without using an access-list.
- When a user is telnetted into R3 and mistypes a command in exec mode, R3 should not try to open a telnet session to the command as if it was a hostname.
- Configure VTY line 0 to listen for telnet at port 3001.
- When the virtual terminal line is busy issue the output “Sorry, the line is already in use” to the connecting user.
- Exec sessions on a VTY line should timeout after 2 minutes of inactivity; additionally a user should not be able to hold the line busy for more than 5 minutes.
- Sessions initiated from a VTY line should time out in 1 minute.
- When the console line is idle the user should see the output “Welcome to IOS”
- Allow no more than 1 session to be initiated from the console line.
- The terminal length should be no more than 20 lines.
- IP netmasks should be displayed using hex numbers.
- Allow a user to lock VTY terminal lines.

Configuration

```
R3:
line vty 0 4
 session-timeout 1
 exec-timeout 2 0
 lockable
 absolute-timeout 5
 ip netmask-format hexadecimal
 refuse-message #
Sorry, the line is already in use
#
 length 20
 transport preferred none
 transport input telnet
!
line vty 0
 rotary 1
!
line console 0
 session-limit 1
 vacant-message #
Welcome to IOS
#
```

Copyright © 2010 Internetwork Expert

Verification

Note

IOS supports multiple timeout settings for terminal sessions, which can be divided as follows. Absolute – limits the maximum amount of time the user could spend on the line. Exec – limits the time an exec shell can be idle. Session – the maximum amount of time a session opened from terminal (e.g. telnet to other router) could be idle.

A refuse message is displayed when someone tries to connect to a line already in use. The vacant message is displayed when the current line is idle (not in use, e.g. no exec shell started).

The transport input/output option specifies which protocols can be used to connect to/from the terminal line. The preferred transport protocol is used when no session protocol is specified at exec prompt and a station name is typed. By default the preferred transport is telnet, which is why the router tries to telnet when a command is mistyped.

The lock feature allows a user to lock the current terminal session and require a password to unlock it.

Copyright © 2010 Internetwork Expert

```
Rack1R3#show line vty 0
   Tty Typ     Tx/Rx    A Modem  Roty AccO AccI   Uses   Noise  Overruns   Int
    130 VTY              -    -      -    -    -     10       0     0/0       -

Line 130, Location: "", Type: ""
Length: 20 lines, Width: 80 columns
Baud rate (TX/RX) is 9600/9600
Status: Ready, No Exit Banner
Capabilities: Lockable
Modem state: Ready
Group codes:    0
Special Chars: Escape  Hold  Stop  Start  Disconnect  Activation
                ^^x    none   -     -       none
Timeouts:      Idle EXEC    Idle Session   Modem Answer  Session   Dispatch
               00:02:00       00:01:00                  00:05:00    not set
                            Idle Session Disconnect Warning
                              never
                            Login-sequence User Response
                             00:00:30
                            Autoselect Initial Wait
                              not set
Modem type is unknown.
Session limit is not set.
Time since activation: never
Editing is enabled.
History is enabled, history size is 20.
DNS resolution in show commands is enabled
Full user help is disabled
Allowed input transports are telnet.
Allowed output transports are lat pad v120 lapb-ta mop telnet rlogin ssh.
Preferred transport is none.
No output characters are padded
No special data dispatching characters
```

The netmask format can be verified as below.

Copyright © 2010 Internetwork Expert

```
Rack1R3#telnet 150.1.3.3
Trying 150.1.3.3 ... Open


User Access Verification

Password: cisco

Rack1R3#show interface serial 1/0
Serial1/0 is up, line protocol is up
  Hardware is CD2430 in sync mode
  Internet address is 155.1.0.3 0xFFFFFF00
  MTU 1500 bytes, BW 128 Kbit, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation FRAME-RELAY, loopback not set
```

Rotary groups allow bundling multiple lines into a pool and to give the option to access the pool using the dedicated TCP port number 3000+N, where N is the rotary group number. Those special port numbers can also be used as "backdoors" for telnet access. Note that the refuse message is displayed when a user attempts to connect to the busy line.

```
Rack1R3#telnet 150.1.3.3 3001
Trying 150.1.3.3, 3001 ... Open


User Access Verification

Password: cisco

Rack1R3>telnet 150.1.3.3 3001
Trying 150.1.3.3, 3001 ... Open

Sorry the line is already in use

[Connection to 150.1.3.3 closed by foreign host]
```

Copyright © 2010 Internetwork Expert

The console session limit can be verified as follows.

```
Rack1R3#telnet 150.1.2.2
Trying 150.1.2.2 ... Open


User Access Verification

Password: cisco
Rack1R2>Ctrl-Shift-6-x
Rack1R3#telnet 150.1.5.5
Session limit exceeded

Rack1R3#w
Conn Host                Address            Byte  Idle Conn Name
*  1 150.1.2.2           150.1.2.2             0     0 150.1.2.2
```

To verify session locking login in via telnet and issuing the **lock** command.

```
Rack1R3#telnet 150.1.3.3
Trying 150.1.3.3 ... Open

User Access Verification

Password: cisco
Rack1R3>lock
Password: cisco
Again: cisco

<snip>

      Locked

Rack1R3#telnet 150.1.3.3
Trying 150.1.3.3 ... Open
```

Copyright © 2010 Internetwork Expert

Verify that setting the preferred transport to “none” disables automatic telnet sessions opening for hostnames entered in the CLI as follows.

```
User Access Verification

Password: cisco

Rack1R3>hostname
         ^
% Invalid input detected at '^' marker.
```

Check the vacant message for the console line as follows.

```
Rack1R3#exit

<snip>

Welcome to IOS
```

12.15 SNMPv2 Server

- Configure SNMP on R4 with the SNMP contact set to “Default Contact” and the SNMP Location to “Default Location”.
- Enable the SNMP server service using the community string value of CISCO so that remote hosts can modify the local MIB database.
- Allow the system to be reloaded via SNMP.
- Ensure that interface index numbers persist between reloads.

  Only allow configuration transfers via TFTP to/from the host 155.X.146.100.

Configuration

```
R4:
snmp-server community CISCO RW
snmp-server location Default Location
snmp-server contact Default Contact
snmp-server ifindex persist
snmp-server system-shutdown
!
access-list 98 permit 155.1.146.100
snmp-server tftp-server-list 98
```

Verification

Note

By default when you configure the SNMP agent with a community string, SNMPv2c is used. This version uses only the community string for authentication. Issuing the **`snmp-server community`** command is the minimal configuration needed to enable the devices to be polled via SNMP. The read-write string allows the management station to make changes to the managed device, as opposed to the read-only string.

SNMP interface index (IfIndex) values are not being the same between reloads by default. This may confuse some network management systems (NMS), as interfaces are identified by the IfIndex value in the SNMP Management Information Base (MIB). To resolve this the SNMP IfIndex persistence feature may be helpful.

To allow the NMS to reload the managed device, read-write access must be allowed, and the **`snmp-server system-shutdown`** feature must be enabled.

Copyright © 2010 Internetwork Expert

The TFTP server list specifies the acceptable addresses to download/upload the router's configuration when instructed via SNMP. In recent IOS versions, the command has been replaced with the command **snmp-server file-transfer access-group**. The IOS still accepts the old command though, but converts it to the new format.

```
Rack1R4#show snmp
Chassis: 26388555
Contact: Default Contact
Location: Default Location
0 SNMP packets input
    0 Bad SNMP version errors
    0 Unknown community name
    0 Illegal operation for community name supplied
    0 Encoding errors
    0 Number of requested variables
    0 Number of altered variables
    0 Get-request PDUs
    0 Get-next PDUs
    0 Set-request PDUs
    0 Input queue packet drops (Maximum queue size 1000)
0 SNMP packets output
    0 Too big errors (Maximum packet size 1500)
    0 No such name errors
    0 Bad values errors
    0 General errors
    0 Response PDUs
    0 Trap PDUs

SNMP logging: disabled

Rack1R4#show snmp community

Community name: ILMI
Community Index: cisco0
Community SecurityName: ILMI
storage-type: read-only   active


Community name: CISCO
Community Index: cisco3
Community SecurityName: CISCO
storage-type: nonvolatile active

Rack1R4#show snmp mib ifmib ifindex
Ethernet0/0: Ifindex = 1
Loopback0: Ifindex = 7
Null0: Ifindex = 6
Serial0/0: Ifindex = 3
VoIP-Null0: Ifindex = 5
Ethernet0/1: Ifindex = 2
Serial0/1: Ifindex = 4
```

Copyright © 2010 Internetwork Expert

12.16 SNMPv2c Access Control

- Configure R4 to restrict read-write access to only hosts in VLAN 146.
- Log any other SNMP access attempts using the community string "CISCO".
- Allow any host to access R4's MIBs in read-only mode using the community string "PUBLIC".
- Read-only access should be permitted to the "cisco" MIB subtree only.

Configuration

```
R4:
access-list 99 permit 155.1.146.0 0.0.0.255
access-list 99 deny any log
!
snmp-server view ROVIEW cisco included
snmp-server community CISCO RW 99
snmp-server community PUBLIC view ROVIEW ro
```

Verification

Note

It is highly advisable to limit access to managed SNMP devices by associating an access-list with the SNMP server community. By using the log keyword it is possible to expose other hosts that attempt to access the router via SNMP.

Additionally SNMPv2 allows the restricting of access to certain MIB values using the concept of views. A view is partition of the whole MIB tree built by including or excluding various subtrees. Note that view definition may have multiple include/exclude statement, and the latter ones take preference.

Copyright © 2010 Internetwork Expert

```
Rack1R4#show snmp community

Community name: ILMI
Community Index: cisco0
Community SecurityName: ILMI
storage-type: read-only   active


Community name: CISCO
Community Index: cisco4
Community SecurityName: CISCO
storage-type: nonvolatile active      access-list: 99


Community name: PUBLIC
Community Index: cisco5
Community SecurityName: PUBLIC
storage-type: nonvolatile active

Rack1R4#show snmp view
*ilmi system - included permanent active
*ilmi atmForumUni - included permanent active
ROVIEW cisco - included nonvolatile active
v1default iso - included permanent active
v1default internet.6.3.15 - excluded permanent active
v1default internet.6.3.16 - excluded permanent active
v1default internet.6.3.18 - excluded permanent active
v1default ciscoMgmt.394 - excluded permanent active
v1default ciscoMgmt.395 - excluded permanent active
v1default ciscoMgmt.399 - excluded permanent active
v1default ciscoMgmt.400 - excluded permanent active
```

Copyright © 2010 Internetwork Expert

12.17 SNMP Traps and Informs

- Configure R4 to send SNMP traps to the management host 155.X.146.100 and informs to the host 155.X.146.101.
- Only send notifications about interfaces going up or down.
- Ensure no informs are being sent about Serial interfaces changing their status.
- Use the community value of "CISCO" with the above traps and informs.

Configuration

```
R4:
snmp-server enable traps snmp linkdown linkup
snmp-server host 155.1.146.101 inform version 2c CISCO
snmp-server host 155.1.146.100 CISCO
!
interface Serial0/0
 no snmp trap link-status
!
interface Serial0/1
 no snmp trap link-status
```

Verification

Note

SNMP traps are part of SNMPv1 and SNMPv2 specifications. Additionally SNMPv2 allows sending notifications as informs, which differ from traps in that they require acknowledgement from the NMS. Informs are kept in router local queue until they are acknowledged or timeout has expired. Informs make SNMP reliable even though the transport protocol is still UDP.

The traps which are to be sent can be configured either globally or on a per-host basis. In the latter case the host settings override the global settings. Some traps however can only enabled globally, such as link up and down notifications. If the single command **`snmp-server enable traps`** is entered, all traps will be sent to all configured destinations, unless they are overridden on a per-host basis. Note that *it is required* to configure both the **`snmp-server enable traps`** and the **`snmp-server host`** commands in order to send notifications to a particular host.

Copyright © 2010 Internetwork Expert

Informs sending is enabled by default, and the same notifications configured with the global **snmp-server enable traps** command are sent to NMS hosts, provided that they are configured for informs. The same host may receive informs and traps in parallel, though this configuration is uncommon.

```
Rack1R4#show snmp host
Notification host: 155.1.146.101      udp-port: 162      type: inform
user: CISCO  security model: v2c

Notification host: 155.1.146.100      udp-port: 162      type: trap
user: CISCO  security model: v1
```

To verify SNMP trapping, enable SNMP packet debugging and create a link down event.

```
Rack1R4#debug snmp packet
SNMP packet debugging is on
Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface Ethernet 0/0
Rack1R4(config-if)#shutdown
Rack1R4(config-if)#
SNMP: Inform request, reqid 17, errstat 0, erridx 0
 sysUpTime.0 = 2019043
 snmpTrapOID.0 = snmpTraps.3
 ifIndex.1 = 1
 ifDescr.1 = Ethernet0/0
 ifType.1 = 6
 lifEntry.20.1 = administratively down
Packet sent via UDP to 155.1.146.101.162
SNMP: Queuing packet to 155.1.146.100
SNMP: V1 Trap, ent snmpTraps, addr 155.1.146.4, gentrap 2, spectrap 0
 ifIndex.1 = 1
 ifDescr.1 = Ethernet0/0
 ifType.1 = 6
 lifEntry.20.1 = administratively down
Jul 18 02:24:36.590: SNMP: Packet sent via UDP to 155.1.146.100
```

Copyright © 2010 Internetwork Expert

12.18 CPU and Memory Thresholds

- Configure R4 to monitor CPU usage every 5 seconds using the **process cpu** command, and to generate a rising threshold event every time the CPU usage hits 50%.
- Set up the free memory low threshold to 1000Kbytes, and reserve 512Kbytes of memory for the notification process using the **memory** command.

Configuration

```
R4:
snmp-server enable traps cpu threshold
!
memory free low-watermark processor 1000
process cpu threshold type total rising 50 interval 5
memory reserve critical 512
```

Verification

Note

CPU threshold notification allows sending SNMP traps and/or informs when the CPU usage crosses defined boundaries. A trap could be generated on the CPU value crossing the rising threshold (going above) or the falling threshold (going below). Additionally the memory threshold notification feature will post a syslog message when the amount of free memory shrinks below a defined limit.

To verify CPU threshold tracking enable SNMP packet debugging and set the CPU rising threshold to a very low value. After that, execute the **show run command** to cause a CPU spike.

Copyright © 2010 Internetwork Expert

```
Rack1R4#debug snmp packet
Rack1r4#conf t
Rack1R4(config)#process cpu threshold type total rising 5 interval 5
Rack1R4(config)#^Z
Rack1R4#sh run
Building configuration...

Current configuration : 2493 bytes
 ..

%SYS-1-CPURISINGTHRESHOLD: Threshold: Total CPU Utilization(Total/
Intr): 32%/0%, Top 3 processes(Pid/Util):  3/31%, 91/0%, 2/0%

SNMP: Inform request, reqid 21, errstat 0, erridx 0
 sysUpTime.0 = 2146590
 snmpTrapOID.0 = ciscoProcessMIB.2.0.1
 cpmCPUThresholdTable.1.2.1.1 = 5
 cpmCPUTotalTable.1.10.1 = 32
 cpmCPUTotalTable.1.11.1 = 0
 ciscoProcessMIB.1.2.3.1.5.1.3 = 31
 cpmProcessTable.1.5.1.3 = 2146497
SNMP: Packet sent via UDP to 155.1.146.101.162
SNMP: Queuing packet to 155.1.146.100
SNMP: V1 Trap, ent ciscoProcessMIB.2, addr 155.1.146.4, gentrap 6,
spectrap 1
 cpmCPUThresholdTable.1.2.1.1 = 5
 cpmCPUTotalTable.1.10.1 = 32
 cpmCPUTotalTable.1.11.1 = 0
 ciscoProcessMIB.1.2.3.1.5.1.3 = 31
Rack1R4#
 cpmProcessTable.1.5.1.3 = 2146497
SNMP: Packet sent via UDP to 155.1.146.100
```

To verify free memory threshold notifications find out the current amount of free processor memory, set the threshold to a high value, and then again to a lower value.

```
Rack1R4(config)#do show memory summmary
                Head    Total(b)     Used(b)     Free(b)   Lowest(b)  Largest(b)
Processor    6420A860    59725728    14638656    45087072    44403156    43632028
      I/O     7B00000     5242880     1743784     3499096     3499096     3499068
Critical     650EDF74      481976          52      481924      481924      481924
Critical      7C9F640       42312          52       42260       42260       42260

Rack1R4(config)#memory free low-watermark processor 50000

%SYS-4-FREEMEMLOW: Free Memory has dropped below low watermark
Pool: Processor  Free: 45087396  Threshold: 51200000

Rack1R4(config)#memory free low-watermark processor 5000

%SYS-5-FREEMEMRECOVER: Free Memory has recovered above low watermark
Pool: Processor  Free: 45087660 Threshold: 5120000
```

Copyright © 2010 Internetwork Expert

12.19 SNMPv3

- Create two SNMP views named NORMAL and RESTRICTED on R6.
- The NORMAL view should include the branch “iso”.
- The RESTRICTED view should include the branch “ifEntry.*.*n*”, where *n* is the ifIndex of R6’s serial interface.
- Create an SNMP group named NORMAL with the read/write view assigned to NORMAL, and a security model of “priv”.
- Assign the user named NORMAL to this group, set the SHA1 password to CISCO, and the encryption key to CISCO.
- Create an SNMP group named RESTRICTED with the read view assigned to RESTRICTED, and a security model of “auth”.
- Only users in VLAN 146 should be allowed in the SNMP group RESTRICTED.
- Assign the user named RESTRICTED to this group using the security model which only requires authentication using the password CISCO.
- Create an SNMP group named TRAP with the security model “priv”.
- Assign the user named TRAP to this group using the password and encryption key CISCO.
- Enable SNMP traps for LinkUp and LinkDown events only, and send them to the destination host 155.X.146.100 using the security model “priv” and the username TRAP.

Configuration

```
R6:
access-list 99 permit 155.1.146.0 0.0.0.255
!
snmp-server ifindex persist
snmp-server view NORMAL iso included
snmp-server view RESTRICTED ifEntry.*.3 included
!
snmp-server group NORMAL v3 priv read NORMAL write NORMAL
snmp-server group RESTRICTED v3 auth read RESTRICTED access 99
snmp-server group TRAP v3 priv
!
snmp-server user NORMAL NORMAL v3 auth sha CISCO priv des56 CISCO
snmp-server user RESTRICTED RESTRICTED v3 auth sha CISCO
snmp-server user TRAP TRAP v3 auth sha CISCO priv des56 CISCO
!
snmp-server enable traps snmp linkup linkdown
snmp-server host 155.1.146.100 traps version 3 priv TRAP
```

Copyright © 2010 Internetwork Expert

Verification

Note

SNMPv3 extends the previous versions of SNMP by introducing a new security model that replaces the old community-based authentication system. SMNPv3 also provides for communication privacy by means of encryption. The new concepts for SNMPv3 are the user, group, and security level.

A group defines what access rights a set of users have. This access policy controls which SNMP objects (MIBs) can be accessed for reading and writing, or which SNMP objects can generate notifications to the members of a group. The policy is defined by associating a read, write, or notify view with the group. By using a notify view, a group determines the list of notifications its users can receive. The group also defines the security *model* (SNMP version) and the security *level* (authentication and/or encryption) for its users.

If a group is defined without a *read* view, all objects are available to be read (implicit permit). Contrary to that, if a *write* or *notify* view is not defined, no write access is granted, and no objects can send notifications to members of the group (implicit deny). The notify view is usually not configured manually, and is auto-generated by the **`snmp-server host`** command when users in a group are bound to a notification target host.

The security *models* are defined as SNMPv1, SNMPv2, SNMPv3, while the security *levels* are defined as noAuthNoPriv, AuthNoPriv, and AuthPriv. noAuthNoPriv, the **`noauth`** keyword in the IOS, means no authentication and no encryption. AuthNoPriv, the **`auth`** keyword in the IOS, means authentication but no encryption. AuthPriv, the **`priv`** keyword in IOS, means authentication and encryption.

SNMPv3 can implement any of the three above security levels. SNMPv1 and SNMPv2 only support noAuthNoPriv. In the case that SNMPv3 uses noAuthNoPriv, the username serves as a replacement for the community string. All users sharing a group utilize the same security model, however the specific model settings (password and encryption key) are set per-user. Note that SNMPv3 does not send passwords in clear-text, but instead uses MD5 or SHA1 hash-based authentication. For encryption, statically configured keys are used along with a single-DES (56-bit) symmetric cipher. This means that the same key should be configured on NMS for the particular user.

Copyright © 2010 Internetwork Expert

Note that SNMPv3 users do not appear in the running configuration for security reasons. The basic configuration of SNMPv3 can be verified as follows.

```
Rack1R6#show snmp user

User name: TRAP
Engine ID: 800000090300000119221DA80
storage-type: nonvolatile active
Authentication Protocol: SHA
Privacy Protocol: DES
Group-name: TRAP

User name: NORMAL
Engine ID: 800000090300000119221DA80
storage-type: nonvolatile active
Authentication Protocol: SHA
Privacy Protocol: DES
Group-name: NORMAL

User name: RESTRICTED
Engine ID: 800000090300000119221DA80
storage-type: nonvolatile active
Authentication Protocol: SHA
Privacy Protocol: None
Group-name: RESTRICTED
```

For the TRAP group the notify view is auto-generated by the **snmp-server host** command, which bound the user (TRAP) and the group it belongs to (TRAP) to the list of notifications which are to be sent to the host.

```
Rack1R6#show snmp group
groupname: ILMI                           security model:v1
readview : *ilmi                          writeview: *ilmi
notifyview: <no notifyview specified>
row status: active

groupname: ILMI                           security model:v2c
readview : *ilmi                          writeview: *ilmi
notifyview: <no notifyview specified>
row status: active

groupname: TRAP                           security model:v3 noauth
readview : <no readview specified>        writeview: <no writeview
specified>
notifyview: *tv.FFFFFFFF.FFFFFFFF.FFFFFFFF0F
row status: active

groupname: TRAP                           security model:v3 priv
readview : v1default                      writeview: <no writeview
specified>
notifyview: <no notifyview specified>
row status: active
```

```
groupname: NORMAL                              security model:v3 priv
readview : NORMAL                              writeview: NORMAL
notifyview: <no notifyview specified>
row status: active

groupname: RESTRICTED                          security model:v3 auth
readview : RESTRICTED                          writeview: <no writeview
specified>
notifyview: <no notifyview specified>
row status: active access-list: 99

Rack1R6#show snmp view
*ilmi system - included permanent active
*ilmi atmForumUni - included permanent active
NORMAL iso - included nonvolatile active
v1default iso - included permanent active
v1default internet.6.3.15 - excluded permanent active
v1default internet.6.3.16 - excluded permanent active
v1default internet.6.3.18 - excluded permanent active
v1default ciscoMgmt.394 - excluded permanent active
v1default ciscoMgmt.395 - excluded permanent active
v1default ciscoMgmt.399 - excluded permanent active
v1default ciscoMgmt.400 - excluded permanent active
RESTRICTED ifEntry.0.3 FF:EF included nonvolatile active
*tv.FFFFFFFF.FFFFFFFF.FFFFFFF0F iso.2.840.10036 - included volatile
active
*tv.FFFFFFFF.FFFFFFFF.FFFFFFF0F internet - included volatile active
```

To ensure that SNMP traps are being sent in encrypted and authenticated, enable SNMP packet debugging along with a low-level packet dump for outgoing SNMP traps as follows.

```
Rack1R6(config)#access-list 100 permit udp any any eq 162
Rack1R6#debug ip packet detail 100 dump
IP packet debugging is on (detailed) (dump) for access list 100

Rack1R6#debug snmp packet
SNMP packet debugging is on

Rack1R6#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface loopback 0
Rack1R6(config-if)#shutdown
Rack1R6(config-if)#

SNMP Queuing packet to 155.1.146.100:
SNMP: V2 Trap, reqid 23, errstat 0, erridx 0
 sysUpTime.0 = 2893642
 snmpTrapOID.0 = snmpTraps.3
 ifIndex.11 = 11
 ifDescr.11 = Loopback0
 ifType.11 = 24
 lifEntry.20.11 = administratively down
```

Copyright © 2010 Internetwork Expert

```
SNMP: Packet sent via UDP to 155.1.146.100
IP: tableid=0, s=155.1.146.6 (local), d=155.1.146.100
(FastEthernet0/0.146), routed via RIB
IP: s=155.1.146.6 (local), d=155.1.146.100 (FastEthernet0/0.146), len
283, sending
    UDP src=58135, dst=162
07802010:                   4500011B 00060000          E. ....
07802020: FF11605E 9B019206 9B019264 E31700A2  ..`^. .....dc.."
07802030: 01077D80 3081FC02 0103300D 02011202  ..}.0.|...0. ..
07802040: 0205DC04 01030201 03043530 33040C80  ..\. .....503...
07802050: 00000903 00001192 21DA8002 01180202  . ......!Z......
07802060: 190C0404 54524150 040C3972 C05E1654  ....TRAP..9r@^.T
07802070: 7D2249D0 11F10408 00000018 F1F81DCD  }"IP.q. ....qx.M
07802080: 0481B021 3372C60A 394B07E8 93FD35A3  ..0!3rF.9K.h.}5#
07802090: 584291FA 887D96B8 6894CCE6 58AAD5DD  XB.z.}.8h.LfX*U]
078020A0: 5F4EACE4 4B30FB5C 25B58A78 09A78EF0  _N,dK0{\%5.x.'.p
078020B0: 863CEE49 3745902A FEE6530C BAD247DC  .<nI7E.*~fS.:RG\
078020C0: DF94530F E5ED993C FC955C8C 3B42FC15  _.S.em.<|.\.;B|.
078020D0: D4C2B05B 8CF2869A FED2EC8E 38570FBB  TB0[.r..~Rl.8W.;
078020E0: C1454016 599591C2 D3FA07B1 0D048B26  AE@.Y..BSz.1...&
078020F0: EF4CBD1F 34B5E63A 0C05AA56 385B32D2  oL=.45f:..*V8[2R
07802100: E5AB5AB6 F9F722F2 13C5610E 4ADE3FC3  e+Z6yw"r.Ea.J^?C
07802110: 73F78791 74AED319 5F86D648 74A04BE9  sw..t.S._.VHt Ki
07802120: 1CCDAB2E 0D022589 6BFBE813 E613B58C  .M+...%.k{h.f.5.
07802130: 739C66                               s.f
IP: s=155.1.146.6 (local), d=155.1.146.100 (FastEthernet0/0.146), len
283, encapsulation failed
    UDP src=58135, dst=162
07802010:                   4500011B 00060000          E. ....
07802020: FF11605E 9B019206 9B019264 E31700A2  ..`^. .....dc.."
07802030: 01077D80 3081FC02 0103300D 02011202  ..}.0.|...0. ..
07802040: 0205DC04 01030201 03043530 33040C80  ..\. .....503...
07802050: 00000903 00001192 21DA8002 01180202  . ......!Z......
07802060: 190C0404 54524150 040C3972 C05E1654  ....TRAP..9r@^.T
07802070: 7D2249D0 11F10408 00000018 F1F81DCD  }"IP.q. ....qx.M
07802080: 0481B021 3372C60A 394B07E8 93FD35A3  ..0!3rF.9K.h.}5#
07802090: 584291FA 887D96B8 6894CCE6 58AAD5DD  XB.z.}.8h.LfX*U]
078020A0: 5F4EACE4 4B30FB5C 25B58A78 09A78EF0  _N,dK0{\%5.x.'.p
078020B0: 863CEE49 3745902A FEE6530C BAD247DC  .<nI7E.*~fS.:RG\
078020C0: DF94530F E5ED993C FC955C8C 3B42FC15  _.S.em.<|.\.;B|.
078020D0: D4C2B05B 8CF2869A FED2EC8E 38570FBB  TB0[.r..~Rl.8W.;
078020E0: C1454016 599591C2 D3FA07B1 0D048B26  AE@.Y..BSz.1...&
078020F0: EF4CBD1F 34B5E63A 0C05AA56 385B32D2  oL=.45f:..*V8[2R
07802100: E5AB5AB6 F9F722F2 13C5610E 4ADE3FC3  e+Z6yw"r.Ea.J^?C
07802110: 73F78791 74AED319 5F86D648 74A04BE9  sw..t.S._.VHt Ki
07802120: 1CCDAB2E 0D022589 6BFBE813 E613B58C  .M+...%.k{h.f.5.
07802130: 739C66                               s.f             -if)#
```

Change the security model for the destination host to “noAuth” and generate a trap message again. Note that now the message is not encrypted.

```
Rack1R6# snmp-server host 155.1.146.100 version 3 noauth TRAP

Rack1R6(config)#interface Loopback 0
Rack1R6(config-if)#no shut
Rack1R6(config-if)#
SNMP: Queuing packet to 155.1.146.100
SNMP: V2 Trap, reqid 24, errstat 0, erridx 0
 sysUpTime.0 = 2928473
 snmpTrapOID.0 = snmpTraps.4
 ifIndex.11 = 11
 ifDescr.11 = Loopback0
 ifType.11 = 24
 lifEntry.20.11 = up
SNMP: Packet sent via UDP to 155.1.146.100
IP: tableid=0, s=155.1.146.6 (local), d=155.1.146.100
(FastEthernet0/0.146), routed via RIB
IP: s=155.1.146.6 (local), d=155.1.146.100 (FastEthernet0/0.146), len
238, sending
    UDP src=58135, dst=162
079D76B0:                   450000EE 00070000          E..n....
079D76C0: FF11608A 9B019206 9B019264 E31700A2  ..`. ......dc.."
079D76D0: 00DA3F33 3081CF02 0103300D 02011302  .Z?30.O...0. ..
079D76E0: 0205DC04 01000201 03042130 1F040C80  ..\. .....!0....
079D76F0: 00000903 00001192 21DA8002 01180202  . ......!Z......
079D7700: 1A680404 54524150 04000400 30819704  .h..TRAP....0...
079D7710: 0C800000 09030000 119221DA 800400A7  . ........!Z...'
079D7720: 81840201 18020100 02010030 79300F06  . .........0y0..
079D7730: 082B0601 02010103 0043032C AF593017  .+. .....C.,/Y0.
079D7740: 060A2B06 01060301 01040100 06092B06  ..+. .........+.
079D7750: 01060301 01050430 0F060A2B 06010201  . .....0...+....
079D7760: 02020101 0B02010B 3017060A 2B060102  . ......0...+...
079D7770: 01020201 020B0409 4C6F6F70 6261636B  . ......Loopback
079D7780: 30300F06 0A2B0601 02010202 01030B02  00...+. .......
079D7790: 01183012 060C2B06 01040109 02020101  ..0...+. ......
079D77A0: 140B0402 7570                        ....up
<snip>
```

Copyright © 2010 Internetwork Expert

12.20 SNMP MAC Address Notifications

- Configure SW1 to notify the NMS station at the IP address 155.X.146.100 about MAC addresses learned on ports connected to R1 and R5.
- Use the community value of CISCO to send notifications.
- Limit the rate of notifications to 1 per second.
- Report both added and removed MAC addresses.
- Store the latest 100 notification events in history buffer.

Configuration

```
SW1:
interface FastEthernet0/1
 snmp trap mac-notification added
 snmp trap mac-notification removed
!
interface FastEthernet0/5
 snmp trap mac-notification added
 snmp trap mac-notification removed
!
snmp-server enable traps mac-notification
!
mac-address-table notification interval 2
mac-address-table notification history-size 100
mac-address-table notification
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> MAC Address table notifications allow the switches to report changes in the CAM table entries (adds and removes) for a particular link. This feature must be first enabled globally, and then configured on per-interface level. On trunk interfaces all VLANs are affected by the configuration. In addition to reporting changes to an NMS the switch can store recent events in the local buffer.

```
Rack1SW1#clear mac-address-table dynamic
Rack1SW1#show mac-address-table notification
MAC Notification Feature is Enabled on the switch
Interval between Notification Traps : 2 secs
Number of MAC Addresses Added : 7
Number of MAC Addresses Removed : 7
Number of Notifications sent to NMS : 9
Maximum Number of entries configured in History Table : 100
Current History Table Length : 2
MAC Notification Traps are Enabled
History Table contents
----------------------
History Index 0, Entry Timestamp 3441456, Despatch Timestamp 3441456
MAC Changed Message :
Operation: Deleted Vlan: 58    MAC Addr: 0004.9a0b.62c1 Dot1dBasePort: 7
Operation: Deleted Vlan: 146   MAC Addr: 0011.2030.f900 Dot1dBasePort: 3
Operation: Added   Vlan: 58    MAC Addr: 0004.9a0b.62c1 Dot1dBasePort: 7
History Index 1, Entry Timestamp 3441657, Despatch Timestamp 3441657
MAC Changed Message :
Operation: Added   Vlan: 146   MAC Addr: 0011.2030.f900 Dot1dBasePort: 3
```

Copyright © 2010 Internetwork Expert

12.21 SNMP Notifications of Syslog Messages

- Configure R1 so that all debugging and higher priority level messages are sent via SNMP to an NMS host at the IP address 155.X.146.100.
- Set the syslog to SNMP buffer size to 100 messages.

Configuration

```
R1:
snmp-server enable traps syslog
snmp-server host 155.1.146.100 CISCO
!
logging history debugging
logging history size 100
```

Verification

> **Note**
>
> SNMP logging of syslog messages allows the router to forward syslog messages to a remote NMS station using SNMP trap PDUs. Syslog fist sends the logs to a special history buffer, then the SNMP agent replicates the messages as SNMP traps. This requires that "syslog" traps have been enabled globally or for that particular NMS.
>
> To verify this configuration enable SNMP packet debugging and generate a syslog message by shutting down an interface.

```
Rack1R1#debug snmp packets
SNMP packet debugging is on
Rack1R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#interface serial 0/0
Rack1R1(config-if)#shutdown
Rack1R1(config-if)#^Z
Rack1R1#
SNMP: Queuing packet to 155.1.146.100
SNMP: V1 Trap, ent ciscoSyslogMIB.2, addr 155.1.146.1, gentrap 6,
spectrap 1
 clogHistoryEntry.2.18 = LINK
 clogHistoryEntry.3.18 = 6
 clogHistoryEntry.4.18 = CHANGED
 clogHistoryEntry.5.18 = Interface Serial0/0, changed state to
administratively down
 clogHistoryEntry.6.18 = 3559736
SNMP: Packet sent via UDP to 155.1.146.100
SNMP: Queuing packet to 155.1.146.100
SNMP: V1 Trap, ent ciscoSyslogMIB.2, addr 155.1.146.1, gentrap 6,
spectrap 1
 clogHistoryEntry.2.19 = SYS
```

Copyright © 2010 Internetwork Expert

```
 clogHistoryEntry.3.19 = 6
 clogHistoryEntry.4.19 = CONFIG_I
 clogHistoryEntry.5.19 = Configured from console by console
 clogHistoryEntry.6.19 = 3559777
SNMP: Packet sent via UDP to 155.1.146.100
SNMP: Queuing packet to 155.1.146.100
SNMP: V1 Trap, ent ciscoSyslogMIB.2, addr 155.1.146.1, gentrap 6,
spectrap 1
 clogHistoryEntry.2.20 = LINEPROTO
 clogHistoryEntry.3.20 = 6
 clogHistoryEntry.4.20 = UPDOWN
 clogHistoryEntry.5.20 = Line protocol on Interface Serial0/0, changed
state to down
 clogHistoryEntry.6.20 = 3559836
SNMP: Packet sent via UDP to 155.1.146.100

Rack1R1#show snmp
Chassis: JAE0817F53F
<snip>
3 SNMP packets output
    0 Too big errors (Maximum packet size 1500)
    0 No such name errors
    0 Bad values errors
    0 General errors
    0 Response PDUs
    3 Trap PDUs

SNMP logging: enabled
    Logging to 155.1.146.100.162, 0/10, 6 sent, 0 dropped.

Rack1R1#show logging history
Syslog History Table:100 maximum table entries,
saving level debugging or higher
 47 messages ignored, 11 dropped, 0 recursion drops
 11 table entries flushed
 SNMP notifications enabled, 6 notifications sent
<snip>
   entry number 18 : LINK-5-CHANGED
    Interface Serial0/0, changed state to administratively down
    timestamp: 3559736
   entry number 19 : SYS-5-CONFIG_I
    Configured from console by console
    timestamp: 3559777
   entry number 20 : LINEPROTO-5-UPDOWN
    Line protocol on Interface Serial0/0, changed state to down
    timestamp: 3559836
```

Copyright © 2010 Internetwork Expert

12.22 CDP

- Configure R4 to send CDP announcement every 10 seconds, and instruct other devices to hold the updates for 40 seconds.
- Use the Loopback0 interface for IP address advertisements in CDP messages.
- Disable logging of duplex mismatch detected via CDP messages.
- Do not send or receive CDP message on R4's connection to VLAN 43.

Configuration

```
R4:
no cdp log mismatch duplex
cdp source-interface Loopback0
cdp timer 10
cdp holdtime 40
!
interface FastEthernet 0/0
 no cdp enable
```

Verification

> **Note**
>
> Cisco Discovery Protocol (CDP) is used to exchange basic device information and aid in troubleshooting. CDP can be enabled/disabled globally with the **`[no] cdp run`** command, and at the interface level with the **`[no] cdp enable`**.

```
Rack1R4#show cdp
Global CDP information:
       Sending CDP packets every 10 seconds
       Sending a holdtime value of 40 seconds
       Sending CDPv2 advertisements is  enabled
       Source interface is Loopback0

Rack1R4#show cdp interface
FastEthernet0/1 is up, line protocol is up
  Encapsulation ARPA
  Sending CDP packets every 10 seconds
  Holdtime is 40 seconds
Serial0/1 is up, line protocol is up
  Encapsulation HDLC
  Sending CDP packets every 10 seconds
  Holdtime is 40 seconds
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show cdp traffic
CDP counters :
      Total packets output: 160, Input: 148
      Hdr syntax: 0, Chksum error: 0, Encaps failed: 0
      No memory: 0, Invalid packet: 0, Fragmented: 0
      CDP version 1 advertisements output: 0, Input: 0
      CDP version 2 advertisements output: 160, Input: 148

Rack1SW4#show cdp neighbors fastEthernet 0/4 detail
-------------------------
Device ID: Rack1R4
Entry address(es):
  IP address: 150.1.4.4
Platform: Cisco 2611XM,  Capabilities: Router Switch IGMP
Interface: FastEthernet0/4,  Port ID (outgoing port): FastEthernet0/1
Holdtime : 33 sec

Version :
Cisco IOS Software, C2600 Software (C2600-ADVENTERPRISEK9-M), Version
12.4(10), RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2006 by Cisco Systems, Inc.
Compiled Wed 16-Aug-06 01:42 by prod_rel_team

advertisement version: 2
VTP Management Domain: ''
Duplex: full
Management address(es):
```

12.23 RMON Alarms

- Configure R1 to monitor the rate of packets entering its interface connecting to VLAN 146 based on the total input packet counter (ifEntry.11).
- If the rate is above 100 packets per minute generate the log message "VLAN146 Interface Congested" and send a trap to the management host at 155.X.146.100.
- Use the SNMP community string CISCO.
- When the packet rate falls below 50 packets per minute, generate another log message "VLAN146 Interface UnCongested" and send a corresponding trap.
- The owner of events should be set to "CISCO".

Configuration

```
R1:
snmp-server host 155.1.146.100 CISCO
snmp-server ifindex persist
!
rmon event 1 log trap CISCO description "VLAN146 Interface Congested"
owner CISCO
rmon event 2 log trap CISCO description "VLAN146 Interface UnCongested"
owner CISCO
!
rmon alarm 1 ifEntry.11.1 60 delta rising-threshold 100 1 falling-
threshold 50 2 owner CISCO
```

Verification

Note

The Remote Monitoring (RMON) feature is used to monitor specified SNMP MIB variables and generate events (syslog messages or SNMP traps) when a threshold or change threshold is crossed. The common misconception about the RMON feature is the difference between an *absolute* sampling and a *delta* sampling.

Copyright © 2010 Internetwork Expert

An absolute sampling threshold is used for variables that increase or decrease over time, and have an upper or lower bound for when a log should be generated. Examples of absolute samplings would be CPU utilization, memory utilization, interface queue depth, etc. A delta sampling threshold is used for variables that either constantly increase (most common) or constantly decrease. Examples of delta samplings would be for interface errors, input packets, output bytes, etc.

For example if the CPU utilization is at 60% at interval 1, and increases to 90% at interval 2, it makes more sense to want to know that the CPU utilization is exactly 90% (absolute) versus it having changed 30% (delta) over the interval. On the other hand if we find that at interval 1 the total number of bytes received inbound on an interface is 1000, and at interval 2 the total bytes increases to 10000, it makes more sense to want to know that the inbound link utilization was 9000 bytes per interval (delta) versus having increased to 10000 bytes (absolute). The problem with the absolute sampling in this second case is that there's no way to tell if those 10000 bytes were received in the last minute or the last month.

RMON configuration consists of defining events and creating an alarm on a known MIB variable. In the presented case, the variable given is a an interface counter (lifEntry.11) and therefore to get rate from it the delta sampling method is applied. Note that the variable name is based on interface index, so additional steps are needed to discover the interface index and ensure that the indexes are persistent across reboots.

To verify the configuration enable SNMP packet debugging and generate traffic across VLAN 146 to make sure the counter increases by more than 100 packets.

```
Rack1R1#debug snmp packet
Rack1R1#ping 155.1.146.6 repeat 150
Type escape sequence to abort.
Sending 150, 100-byte ICMP Echos to 155.1.146.6, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
<snip>
Success rate is 100 percent (150/150), round-trip min/avg/max = 1/2/4 ms

%RMON-5-RISINGTRAP: Rising trap is generated because the value of
ifInUcastPkts.1 exceeded the rising-threshold value 100
Rack1R1#
SNMP: Queuing packet to 155.1.146.100
SNMP: V1 Trap, ent rmon, addr 155.1.146.1, gentrap 6, spectrap 1
 alarmEntry.1.1 = 1
 alarmEntry.3.1 = ifInUcastPkts.1
 alarmEntry.4.1 = 2
 alarmEntry.5.1 = 150
 alarmEntry.7.1 = 100
SNMP: Packet sent via UDP to 155.1.146.100
```

Copyright © 2010 Internetwork Expert

Do not send any packets after that, and wait for a minute.

```
%RMON-5-FALLINGTRAP: Falling trap is generated because the value of
ifInUcastPkts.1 has fallen below the falling-threshold value 50
Rack1R1#
SNMP: Queuing packet to 155.1.146.100
SNMP: V1 Trap, ent rmon, addr 155.1.146.1, gentrap 6, spectrap 2
 alarmEntry.1.1 = 1
 alarmEntry.3.1 = ifInUcastPkts.1
 alarmEntry.4.1 = 2
 alarmEntry.5.1 = 0
 alarmEntry.8.1 = 50
SNMP: Packet sent via UDP to 155.1.146.100
```

Display the RMON event and alarms configuration information and statistics as follows.

```
Rack1R1#show rmon events
Event 1 is active, owned by CISCO
 Description is VLAN146 Interface Congested
 Event firing causes log and trap to community CISCO,
 last event fired at  0y0w0d,09:39:11,
 Current uptime       0y0w0d,09:51:22
 Current log entries:
  index  uptime              description
  1      0y0w0d,09:39:11     VLAN146 Interface Congested
Event 2 is active, owned by CISCO
 Description is VLAN146 Interface UnCongested
 Event firing causes log and trap to community CISCO,
 last event fired at  0y0w0d,09:40:12,
 Current uptime       0y0w0d,09:51:22
 Current log entries:
  index  uptime              description
  1      0y0w0d,03:59:10     VLAN146 Interface UnCongested
  2      0y0w0d,09:38:11     VLAN146 Interface UnCongested
  3      0y0w0d,09:40:12     VLAN146 Interface UnCongested

Rack1R1#show rmon alarm
Alarm 1 is active, owned by CISCO
 Monitors ifInUcastPkts.1 every 60 second(s)
 Taking delta samples, last value was 0
 Rising threshold is 100, assigned to event 1
 Falling threshold is 50, assigned to event 2
 On startup enable rising or falling alarm
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.24 RMON Statistics Collection

- Configure SW1 to collect RMON statistics on packets entering the interface connected to R3.
- Store historical information for the last 100 samples of packet statistics taken every 30 seconds.

Configuration

```
SW1:
interface FastEthernet 0/3
 rmon promiscuous
 rmon collection history 1 owner monitor buckets 100 interval 30
```

Verification

Note

The RMON agent functionality contains a comprehensive set of statistical analysis tools. However, only a few Cisco routers and IOS versions implement the full RMON specification (notably Cisco 2500 models).

Catalyst switches have additional RMON packet statistics collection functionality which allows them to collect and store information on packets entering an interface. RMON collection can be configured using native mode (only analyze packets destined to interface) or promiscuous mode (capture all packets on the segment, even those not destined to interface).

RMON statistics collection can be configured on either layer 2 switchports or native layer 3 routed ports, but not SVIs. History collection allows accumulating packet statistics in time-intervals (buckets) and storing the buckets in the history buffer.

Statistics collection can be verified as follows.

```
Rack1SW1#show rmon statistics
Collection 10003 on FastEthernet0/3 is active, and owned by config,
 Monitors ifIndex.10003 which has
 Received 89005 octets, 397 packets,
 0 broadcast and 221 multicast packets,
 0 undersized and 0 oversized packets,
 0 fragments and 0 jabbers,
 0 CRC alignment errors and 0 collisions.
 # of dropped packet events (due to lack of resources): 0
 # of packets received of length (in octets):
  64: 176, 65-127: 1, 128-255: 0,
  256-511: 220, 512-1023: 0, 1024-1518:0
```

Copyright © 2010 Internetwork Expert

```
Rack1SW1#ping 155.1.37.3 size 500 repeat 100

Type escape sequence to abort.
Sending 100, 500-byte ICMP Echos to 155.1.37.3, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 1/2/9 ms

Rack1SW1#show rmon statistics
Collection 10003 on FastEthernet0/3 is active, and owned by config,
 Monitors ifIndex.10003 which has
 Received 194961 octets, 607 packets,
 0 broadcast and 227 multicast packets,
 0 undersized and 0 oversized packets,
 0 fragments and 0 jabbers,
 0 CRC alignment errors and 0 collisions.
 # of dropped packet events (due to lack of resources): 0
 # of packets received of length (in octets):
  64: 180, 65-127: 1, 128-255: 0,
  256-511: 226, 512-1023: 200, 1024-1518:0
```

Note that the history collection clearly displays the packet "burst" generated by the ping command.

```
Rack1SW1#show rmon history
Entry 1 is active, and owned by monitor
 Monitors ifIndex.10003 every 30 second(s)
 Requested # of time intervals, ie buckets, is 100,
  Sample # 1 began measuring at 07:06:00
   Received 2914 octets, 13 packets,
   0 broadcast and 7 multicast packets,
   0 undersized and 0 oversized packets,
   0 fragments and 0 jabbers,
   0 CRC alignment errors and 0 collisions.
   # of dropped packet events is 0
   Network utilization is estimated at 0
  Sample # 2 began measuring at 07:06:30
   Received 3228 octets, 14 packets,
   0 broadcast and 8 multicast packets,
   0 undersized and 0 oversized packets,
   0 fragments and 0 jabbers,
   0 CRC alignment errors and 0 collisions.
   # of dropped packet events is 0
   Network utilization is estimated at 0
<snip>
Sample # 20 began measuring at 07:15:30
   Received 106828 octets, 214 packets,
   0 broadcast and 8 multicast packets,
   0 undersized and 0 oversized packets,
   0 fragments and 0 jabbers,
   0 CRC alignment errors and 0 collisions.
   # of dropped packet events is 0
   Network utilization is estimated at 2
```

Copyright © 2010 Internetwork Expert

12.25 HTTP Server and Client

- Configure R4 to support the transfer of the IOS image stored in R6's flash using HTTP.
- R4 should authenticate with a username/password pair of CISCO/CISCO stored in the local database of R6
- R6 should only allow HTTP connections at port 8080, from the subnet 150.X.0.0/16, and limit the maximum number of concurrent connections to two.
- Change the default WWW port to 8080.
- Allow R4 to use secure HTTP connection on port 4043, but limit the security features to use only the DES symmetric cipher.

Configuration

```
R6:
username CISCO password 0 CISCO
!
ip http server
ip http max-connections 2
ip http path flash:
ip http port 8080
!
access-list 80 permit 150.1.0.0 0.0.255.255
!
ip http access-class 80
ip http authentication local
ip http secure-server
ip http secure-port 4043
ip http secure-ciphersuite des-cbc-sha

R4:
ip http client source-interface Loopback0
ip http client username CISCO
ip http client password CISCO
ip http client secure-ciphersuite des-cbc-sha
```

Copyright © 2010 Internetwork Expert

Verification

Note

IOS HTTP server is enabled by default to allow for remote router management (SDM). The default authentication is via the enable password (any name and the enable password) and can be changed to use the local database if AAA is not configured. Note that the local user needs privilege level 15 to access the router via HTTP. Additional security settings include setting the access-class or changing the default port number.

Secure HTTP server enables the use of SSL to protect communications. Note that by default the secure server is disabled, and automatically generates a server SSL X.509 certificate when enabled (this will also bring the SSH server up). When changing the server cipher-suite make sure the client will accept it during the handshake.

In addition to server functionality IOS implements an HTTP client. Using this client it's possible to transfer files to the local router from the remote server. The client is even capable of using a proxy server.

To verify the client/server issue a copy http command from R4.

```
Rack1R4#copy http://155.1.146.6:8080/c2600-adventerprisek9-mz.124-10.bin null:
Loading http://155.1.146.6:8080/c2600-adventerprisek9-mz.124-10.bin
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Try to connect to the HTTP server from a different source interface or using different credentials.

```
Rack1R4#conf t
Rack1R4(config)#ip http client source FastEthernet 0/0
Rack1R4(config)#^Z
Rack1R4#copy http://155.1.146.6:8080/c2600-adventerprisek9-mz.124-
10.bin null:
%Error opening http://155.1.146.6:8080/c2600-adventerprisek9-mz.124-
10.bin (I/O error)

Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#ip http client source-interface Loopback 0
Rack1R4(config)#ip http client username USER
Rack1R4(config)#exit
Rack1R4#copy http://155.1.146.6:8080/c2600-adventerprisek9-mz.124-
10.bin null:
%Error opening http://155.1.146.6:8080/c2600-adventerprisek9-mz.124-
10.bin (Permission denied)
```

Copyright © 2010 Internetwork Expert

Verify the HTTP settings on the server. Note that "HTTP secure server client authentication: Disabled" means that the server will not ask for the client's SSL certificate.

```
Rack1R6#show ip http server status
HTTP server status: Enabled
HTTP server port: 8080
HTTP server authentication method: local
HTTP server access class: 0
HTTP server base path: flash:
Maximum number of concurrent server connections allowed: 2
Server idle time-out: 180 seconds
Server life time-out: 180 seconds
Maximum number of requests allowed on a connection: 1
HTTP server active session modules: ALL
HTTP secure server capability: Present
HTTP secure server status: Enabled
HTTP secure server port: 4043
HTTP secure server ciphersuite: des-cbc-sha
HTTP secure server client authentication: Disabled
HTTP secure server trustpoint:
HTTP secure server active session modules: ALL
```

Copyright © 2010 Internetwork Expert

Verify the HTTP client settings on R4 as follows.

```
Rack1R4#show ip http client all

HTTP client status: Enabled
HTTP client application session modules:
Id               : 1
Application Name : HTTP CFS
Version          : HTTP/1.0
Persistent       : persistent
Response-timeout : 0
Retries          : 0
Proxy            :

HTTP client current connections:
Persistent connection = enabled (default)
Connection establishment timeout = 10s (default)
Connection idle timeout = 30s (default)
Maximum number of connection establishment retries = 1 (default)
Maximum http client connections per host : 2
HTTP secure client capability: Present
HTTP secure client ciphersuite: des-cbc-sha
HTTP secure client trustpoint:

local-ipaddress:port  remote-ipaddress:port in-bytes   out-bytes


Total client connections : 0


HTTP client cache:
Maximum Memory size for cache    : 100000 bytes (default)
Maximum memory per cache entry   : 2000 bytes (default)
Memory used                      : 0 bytes
Memory Available                 : 100000 bytes
Cache Ager interval              : 5 minutes (default)
Total entries created            : 0
Id    Type    Url              Memory-size(Bytes)     Refcnt     Valid(Sec)
--------------------------------------------------------------------------


HTTP client statistics:

HTTP client history:
     GET 01:10:45 UTC Wed MMM DD YYYY 150.1.6.6/c2600-adventerprisek9-mz.124...
     GET 01:12:37 UTC Wed MMM DD YYYY 150.1.6.6/c2600-adventerprisek9-mz.124...
```

Copyright © 2010 Internetwork Expert

12.26 FTP Server and Client

- Configure the FTP server service on R6 to serve IOS images off of flash.
- R4 should initiate an FTP session to R6 with the username/password CISCO/CISCO.
- When a transfer occurs R6 should initiate the FTP data channel back to the client.
- Source FTP connections off R4 Loopback0 interface

Configuration

```
R6:
ftp-server enable
ftp-server topdir flash:

R4:
no ip ftp passive
ip ftp source-interface Loopback0
ip ftp username CISCO
ip ftp password CISCO
```

Verification

Note

In addition to FTP client support, IOS may act as a simple FTP server. The server service does not support any authentication, and due to numerous vulnerabilities (such as bug ID CSCse29244, *IOS crash when transferring files via FTP*) Cisco has *removed* the FTP server functionality from recent IOS releases, promising to re-implement a more stable and advanced feature set at a later time. The FTP client feature can be used to save core dumps and transfer files from external FTP servers. For file transfers between IOS routers the HTTP or TFTP protocols should be used. Due to the FTP server crashing IOS, it is impossible to properly verify the FTP server functionality.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.27 TFTP Server and Client

- Configure the TFTP server service on R6.
- When someone requests the file "r6-ios" from R6 it should return its IOS image file.
- Ensure only R1's Loopback0 interface is allowed to connect and retrieve files via TFTP from R6.
- Configure R1 to source TFTP packets off its Loopback0 interface.

Configuration

```
R6:
tftp-server flash:c2600-adventerprisek9-mz.124-10.bin alias r6-ios 10
!
access-list 10 permit 150.1.1.1

R1:
ip tftp source-interface Loopback0
```

Verification

> **Note**
>
> IOS TFTP server allows read-only access to specified files. An access-list can be used to limit the scope of users accessing the files. The alias feature provides abbreviated names for long filenames stored in the flash.

```
Rack1R6#debug tftp events
TFTP Event debugging is on

Rack1R1#copy tftp://150.1.6.6/r6-ios null:
Accessing tftp://150.1.6.6/r6-ios...
Loading r6-ios from 150.1.6.6 (via FastEthernet0/0): !!

Rack1R6#
Looking for r6-ios
TFTP: Opened flash:c2600-adventerprisek9-mz.124-10.bin, fd 0, size
29631128 for process 68
TFTP: Finished flash:c2600-adventerprisek9-mz.124-10.bin, time 00:00:00
for process 68
```

R4 cannot access the same file using TFTP.

```
Rack1R4#copy tftp://150.1.6.6/r6-ios null:
Accessing tftp://150.1.6.6/r6-ios...

Rack1R6#
TFTP: Looking for r6-ios
TFTP: Looking for r6-ios
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.28 Remote Shell

- Configure the network so that R1 is to view the running configuration of R6 using remote shell.
- Source remote-shell connections off the Loopback0 interface of R1
- Allow RCP file-copy from R6 to R1.

Configuration

```
R1:
ip rcmd remote-username RCP
ip rcmd source-interface Loopback0

R6:
ip rcmd rcp-enable
ip rcmd rsh-enable
!
ip rcmd remote-host Rack1R6 150.1.1.1 Rack1R1 enable
ip rcmd remote-host RCP 150.1.1.1 Rack1R1 enable
```

Verification

Note

IOS implementation includes the UNIX-line remote command execution environment (RSH). No password is required to access the local system, just an entry in local ".rhosts"-like table. The table maps a remote station's IP and username to the local username and privilege level (in a classic UNIX system, the privilege level is determined by the local username) using the following syntax:

ip rcmp remote-host <local-name> <remote-IP> <remote-name> <privilege>

The local name in the ".rhosts" table does not need to match any user in the loca database, and commonly the target router's hostname is used as the RCMD local username. When a remote host connects to the local router via remote-shell the remote host's IP address is looked up in ".rhosts" table, the target username specified by remote system is matched against <local-name>, and the source username specified by remote system is matched against <remote-name>. If all fields are matched, remote-shell access is granted. Without the "enable" privilege, only exec privilege level 1 commands are available to the remote user.

Copyright © 2010 Internetwork Expert

Additionally, remote users may use the remote copy (RCP) feature to transfer files from the router to the remote system (similar to HTTP and TFTP transfers). IOS may act as an rsh and rcp client itself, allowing for remote command execution and file transfers. The local hostname is always used as the source username, but the remote (target) username can be changed.

This configuration can be verified as follows.

```
Rack1R6#debug ip tcp rcmd
RCMD transactions debugging is on
Rack1R6#

Rack1R1#rsh 150.1.6.6 /user Rack1R6 show run interface Serial 0/0

Building configuration...

Current configuration : 110 bytes
!
interface Serial0/0
 ip address 54.1.1.6 255.255.255.0
 ip rip advertise 10
 encapsulation frame-relay
end

Rack1R6#
RCMD: [514 <- 150.1.1.1:988] recv \0
RCMD: [514 <- 150.1.1.1:988] recv Rack1R1\0Rack1R6\0show run interface
Serial 0/0\0
RCMD: [514 -> 150.1.1.1:988] send <OK>
```

Copyright © 2010 Internetwork Expert

Transfer a file from R6 to R1 using RCP

```
Rack1R6#show flash:

System flash directory:
File  Length   Name/status
  1   29631128  c2600-adventerprisek9-mz.124-10.bin
  2   1941      saved-config
  3   1036      r6i
  4   274807    crashinfo_20080727-230115
[29909172 bytes used, 3120968 available, 33030140 total]
32768K bytes of processor board System flash (Read/Write)

Rack1R1#copy rcp://150.1.6.6/saved-config null:
Source username [RCP]?
Accessing rcp://RCP@150.1.6.6/saved-config...!
1941 bytes copied in 0.056 secs (34661 bytes/sec)

Rack1R6#
RCMD: [514 <- 150.1.1.1:974] recv Rack1R1\0RCP\0rcp -f saved-config\0
RCMD: [514 -> 150.1.1.1:974] send <OK>
RCMD: [514 <- 150.1.1.1:974] recv <OK>
RCP: [514 -> 150.1.1.1:974] send C0644 1941 saved-config\n
RCMD: [514 <- 150.1.1.1:974] recv <OK>
RCP: [514 -> 150.1.1.1:974] send 1941 bytes
RCMD: [514 -> 150.1.1.1:974] send <OK>
RCMD: [514 <- 150.1.1.1:974] recv <BAD, Write failed>
RCMD: [514 <- 150.1.1.1:967] recv \0
RCMD: [514 <- 150.1.1.1:967] recv Rack1R1\0RCP\0rcp -f saved-config\0
RCMD: [514 -> 150.1.1.1:967] send <OK>
RCMD: [514 <- 150.1.1.1:967] recv <OK>
RCP: [514 -> 150.1.1.1:967] send C0644 1941 saved-config\n
RCMD: [514 <- 150.1.1.1:967] recv <OK>
RCP: [514 -> 150.1.1.1:967] send 1941 bytes
RCMD: [514 -> 150.1.1.1:967] send <OK>
RCMD: [514 <- 150.1.1.1:967] recv <OK>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.29 NTP

- Configure R4 and R6 as authoritative NTP time sources with stratum 5.
- Both R4 and R6 should synchronize with each other.
- R5 should poll R4 and R6 for time updates, but prefer R4.
- Configure R5 to broadcast NTP updates on its connection to VLAN 58, and for SW2 to listen to NTP broadcasts on this link.
- Configure R6 to send multicast NTP packets to the address 239.1.1.1 on its connection to VLAN 67.
- SW1 and SW3 should be able to synchronize with R6 using these multicast packets.

Configuration

```
R4:
ntp master 5
ntp peer 150.1.6.6
ntp source Loopback0

R5:
ntp server 150.1.6.6
ntp server 150.1.4.4 prefer
ntp source Loopback0
!
interface FastEthernet 0/0
 ntp broadcast

R6:
ntp master 5
ntp peer 150.1.4.4
ntp source Loopback0

SW1:
ip multicast-routing distributed
!
interface Vlan67
 ip pim dense-mode
 ntp multicast client 239.1.1.1
!
interface Vlan79
 ip pim dense-mode

SW2:
interface Vlan58
 ntp broadcast client

SW3:
ip multicast-routing
!
interface Vlan79
 ip pim dense-mode
 ntp multicast client 239.1.1.1
```

Copyright © 2010 Internetwork Expert

Verification

Note

NTP time distribution is based on a loop-less tree topology. In the root of the tree resides the absolute time source (server), which is physically connected to an atomic clock, radio clock, or some other highly accurate time source. This server is described as being in *stratum 1*, meaning that it is one hop away from the source of the time.

Every next router that pulls time down from stratum N is automatically assigned to stratum N+1. That router in turn may become a time source (server) for clients in stratum N+2. A router in stratum N will not accept time updates from stratum N+1, enforcing a tree-level hierarchy of client-server relationships and preventing time-synchronization loops. If a router has multiple servers, it will only select one for time synchronization (the one which is closer to NTP root and has the least offset from the currently running clock) and use others as backup. The preference may be adjusted using the **`prefer`** keyword, but only among candidates having other parameters equal (e.g. same stratum).

Two routers in the same stratum may also form peer-to-peer, symmetric relationships. In this mode, both peers update each other's clock. However, just one router in pair may be configured for NTP peering (active peer), the other router will automatically convert into passive mode, and will provide time updates to the dynamic peer.

NTP is also capable of broadcasting or multicasting time updates on a shared interface, and clients may listen to NTP packets on the other side. This allows a single server to synchronize multiple hosts in unsolicited mode.

Remember that NTP may take considerably time to synchronize, especially if router clocks are far out of synch. To speed up this process adjust the clocks manually to values close to the server, in order to accelerate convergence (e.g. when router considers all servers as "insane").

Copyright © 2010 Internetwork Expert

Note that in the following show commands output reference “127.127.7.1” means “self”. Before beginning verifications, ensure you set all clocks to the same value manually. It may take some time still for both masters to synchronize and consider each other “sane”.

```
Rack1R4#clock set 00:00:01 Jan 1 2012
Rack1R5#clock set 00:00:01 Jan 1 2012
Rack1R6#clock set 00:00:01 Jan 1 2012
Rack1SW1#clock set 00:00:01 Jan 1 2012
Rack1SW2#clock set 00:00:01 Jan 1 2012
Rack1SW3#clock set 00:00:01 Jan 1 2012

Rack1R1#show ntp status
Clock is synchronized, stratum 5, reference is 127.127.7.1
nominal freq is 249.5901 Hz, actual freq is 249.5901 Hz, precision is
2**18
reference time is D2AA2FD6.B73C8C6D (01:05:26.715 UTC Sun Jan 1 2012)
clock offset is 0.0000 msec, root delay is 0.00 msec
root dispersion is 21.13 msec, peer dispersion is 21.13 msec

Rack1R4#show ntp associations

      address         ref clock     st  when  poll reach  delay  offset    disp
+~150.1.6.6        127.127.7.1       5    49    64  337     2.7  -28.16     3.7
*~127.127.7.1      127.127.7.1       4     9    64  377     0.0    0.00     0.0
 * master (synced), # master (unsynced), + selected, - candidate, ~ configured

Rack1R6#show ntp status
Clock is synchronized, stratum 5, reference is 127.127.7.1
nominal freq is 249.5901 Hz, actual freq is 249.5901 Hz, precision is 2**16
reference time is D2AA3016.B76908FA (01:06:30.716 UTC Sun Jan 1 2012)
clock offset is 0.0000 msec, root delay is 0.00 msec
root dispersion is 20.49 msec, peer dispersion is 20.49 msec

Rack1R6#show ntp associations

      address         ref clock     st  when  poll reach  delay  offset    disp
+~150.1.4.4        127.127.7.1       5    50    64  376     4.5   22.03     4.1
*~127.127.7.1      127.127.7.1       4    59    64  377     0.0    0.00     0.0
 * master (synced), # master (unsynced), + selected, - candidate, ~ configured

Rack1R5#show ntp status
Clock is synchronized, stratum 6, reference is 150.1.4.4
nominal freq is 249.5901 Hz, actual freq is 249.5903 Hz, precision is 2**16
reference time is D2AA3023.F4764E72 (01:06:43.954 UTC Sun Jan 1 2012)
clock offset is -6.2522 msec, root delay is 46.68 msec
root dispersion is 27.80 msec, peer dispersion is 1.65 msec
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ntp associations

      address         ref clock     st  when  poll reach  delay  offset    disp
*~150.1.4.4         127.127.7.1      5    37    64  377    46.7   -6.25     1.6
+~150.1.6.6         127.127.7.1      5    26    64  377    47.0  -30.80     1.7
 * master (synced), # master (unsynced), + selected, - candidate, ~ configured

Rack1SW2#show ntp status
Clock is synchronized, stratum 7, reference is 155.1.58.5
nominal freq is 119.2092 Hz, actual freq is 119.2076 Hz, precision is 2**17
reference time is D2AA3058.E84973C0 (01:07:36.907 UTC Sun Jan 1 2012)
clock offset is -1.1060 msec, root delay is 48.45 msec
root dispersion is 905.29 msec, peer dispersion is 876.36 msec

Rack1SW2#show ntp associations

      address         ref clock     st  when  poll reach  delay  offset    disp
* 155.1.58.5         150.1.4.4        6    54    64  176     1.8   -1.11   876.4
 * master (synced), # master (unsynced), + selected, - candidate, ~ configured

Rack1SW3#show ip mroute 239.1.1.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C - Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
       V - RD & Vector, v - Vector
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.1.1.1), 00:35:37/stopped, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Vlan79, Forward/Dense, 00:35:37/00:00:00

(155.1.67.6, 239.1.1.1), 00:35:17/00:02:18, flags: PLTX
  Incoming interface: Vlan79, RPF nbr 155.1.79.7
  Outgoing interface list: Null

Rack1SW3#show ntp status
Clock is synchronized, stratum 6, reference is 155.1.67.6
nominal freq is 250.0000 Hz, actual freq is 249.9983 Hz, precision is 2**18
reference time is D2AA28D1.974FD2EF (00:35:29.591 UTC Sun Jan 1 2012)
clock offset is 3.3951 msec, root delay is 2.58 msec
root dispersion is 6.93 msec, peer dispersion is 3.51 msec
```

Copyright © 2010 Internetwork Expert

12.30 NTP Authentication

- Configure R4 and R6 to authenticate each others peering session using the key “CISCO46”.
- R5 should authenticate NTP packets received from R4 and R6 using key values “CISCO4” and “CISCO6” respectively.
- SW2 should only accept NTP updates on VLAN58 if they are authenticated using key value of “CISCO58”.

Configuration

```
R4:
ntp authenticate
ntp authentication-key 46 md5 CISCO46
ntp trusted-key 46
ntp peer 150.1.6.6 key 46
ntp authentication-key 4 md5 CISCO4

R5:
ntp authenticate
ntp authentication-key 4 md5 CISCO4
ntp authentication-key 6 md5 CISCO6
ntp trusted-key 4
ntp trusted-key 6
ntp server 150.1.4.4 key 4
ntp server 150.1.6.6 key 6
ntp authentication-key 58 md5 CISCO58
!
interface FastEthernet 0/0
 ntp broadcast key 58

R6:
ntp authenticate
ntp authentication-key 46 md5 CISCO46
ntp trusted-key 46
ntp peer 150.1.4.4 key 46
ntp authentication-key 6 md5 CISCO6

SW2:
ntp authenticate
ntp authentication-key 58 md5 CISCO58
ntp trusted-key 58
```

Copyright © 2010 Internetwork Expert

Verification

Note

NTP authentication is based on two concepts. First, NTP packets that can update (change) the local clock must be authenticated. Secondly, authenticated NTP packets are signed using an HMAC MD5 signature, and carry the signing key index (number) inside.

Therefore based on the above, only the router which updates its clock (e.g. client or peer) must be configured to require authentication. However, authentication keys must be configured on **both** sides (e.g. client and server) using the **same** key number (index). For example, if R1 is the NTP client and R2 is the NTP server, R1 may send authenticated (signed) NTP request to R2 with the key index 12. R2 will not verify the signature (it's not going to update its clock), but instead it simply looks up the key with index 12 in its local key chain. Using this key the NTP response is signed and sent back to R1.

In order to make router authenticate incoming updates, you must explicitly enable authentication, configure a key with key index, and ensure the authenticating router trusts this key. There is no need to trust keys which are **not** used to authenticate updates (e.g. reply keys).

```
Rack1R4#show ntp associations detail  | inc auth
150.1.6.6 configured, authenticated, selected, sane, valid, stratum 5

Rack1R6#show ntp associations detail  | inc auth
150.1.4.4 configured, authenticated, selected, sane, valid, stratum 5

Rack1R5#show ntp associations detail | inc auth
150.1.4.4 configured, authenticated, selected, sane, valid, stratum 5
150.1.6.6 configured, authenticated, our_master, sane, valid, stratum 5

Rack1SW2#show ntp associations detail | inc auth
155.1.58.5 dynamic, authenticated, our_master, sane, valid, stratum 6
```

Copyright © 2010 Internetwork Expert

12.31 NTP Access Control

- Configure R4 and R6 so that no other devices can update their clocks.
- R5 should only allow R4 and R6 to update its clock.
- R4 and R6 should not serve time out to other devices besides R5.
- SW2 should only accept time from R5 and not other sources, but R5 should be allowed to serve time out to other devices.

Configuration

```
R4:
access-list 6 permit 150.1.6.6
access-list 6 permit 127.127.7.1
access-list 5 permit 150.1.5.5
!
ntp access-group peer 6
ntp access-group serve-only 5

R5:
access-list 46 permit 150.1.4.4
access-list 46 permit 150.1.6.6
!
ntp access-group peer 46

R6:
access-list 4 permit 150.1.4.4
access-list 4 permit 127.127.7.1
access-list 5 permit 150.1.5.5
!
ntp access-group peer 4
ntp access-group serve-only 5

SW2:
access-list 58 permit 155.1.58.5
!
ntp access-group peer 58
```

Copyright © 2010 Internetwork Expert

Verification

Note

NTP access-control divides NTP messages in two categories, control messages and update/request messages. Control messages are those needed to extract specific management information, such as the peer status or set a management parameter. NTP control messages are not needed for proper time synchronization. NTP update/request messages are those messages needed for time synchronization.

NTP access control defines four levels, peer, serve, serve-only, and query-only. Peer permits NTP updates/requests to the host as well as control queries. This is the only access type which allows the host to be synchronized by others. Serve permits NTP requests, but rejects NTP updates (does not change local clock). Control queries are also permitted. Serve-only permits NTP requests only. It rejects attempts to synchronize the local system and does not accept control queries. Query-only accepts NTP control queries. No response to NTP requests are sent, and no local system time synchronization with a remote system is allowed.

When these levels are associated with access-lists on a router an incoming message source IP address is matched in the order listed above (peer, serve, serve-only, and query-only). The first match determines the type of access. If some access-groups are configured but not all, all other types of access from any other sources are implicitly denied.

Note that in this configuration the masters must allow the NTP process to synchronize with themselves, the peer “127.127.7.1”. This is only needed since they have a “peer” access-group for other sources, which controls who can update the clock.

Copyright © 2010 Internetwork Expert

To verify the access-groups change R5's VLAN 5 IP address and see how SW2 is affected.

```
Rack1SW2#show ntp associations detail
155.1.58.5 dynamic, authenticated, our_master, sane, valid, stratum 6
ref ID 150.1.6.6, time D2AA384E.ECE8E83D (01:41:34.925 UTC Sun Jan 1 2012)
our mode bdcast client, peer mode bdcast, our poll intvl 64, peer poll intvl 64
root delay 57.43 msec, root disp 25.04, reach 376, sync dist 59.677
delay 1.77 msec, offset 4.2912 msec, dispersion 5.04
precision 2**16, version 3
org time D2AA3858.DE49674B (01:41:44.868 UTC Sun Jan 1 2012)
rcv time D2AA3858.DD302D03 (01:41:44.864 UTC Sun Jan 1 2012)
xmt time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
filtdelay =     1.77    1.77    1.77    1.77    1.77    1.77    1.77    1.77
filtoffset =    4.29    0.98    0.18   -1.07   -1.43   -1.69   -1.91   -2.21
filterror =     0.99    1.97    2.94    3.92    4.90    5.87    6.85    7.83

Rack1R5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface fastEthernet 0/0
Rack1R5(config-if)#ip address 155.1.58.55 255.255.255.0
Rack1R5(config-if)#
```

Note that although SW2 receives packets from R5's new IP address it does not accept it as a new dynamic server. The old information is kept until it expires.

```
Rack1SW2#debug ntp packet
Rack1SW2#
NTP: rcv packet from 155.1.58.55 to 255.255.255.255 on Vlan58:
 leap 0, mode 5, version 3, stratum 6, ppoll 64
 rtdel 0E97 (56.992), rtdsp 0663 (24.948), refid 96010606 (150.1.6.6)
 ref D2AA398E.EDCC3A82 (01:46:54.928 UTC Sun Jan 1 2012)
 org 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
 rec 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
 xmt D2AA3998.DF41F31B (01:47:04.872 UTC Sun Jan 1 2012)
 inp D2AA3998.DCA90836 (01:47:04.861 UTC Sun Jan 1 2012)

Rack1SW2#show ntp associations

      address         ref clock     st  when  poll reach  delay  offset    disp
* 155.1.58.5       150.1.6.6         6   307    64  340     1.8    5.39  16000.
 * master (synced), # master (unsynced), + selected, - candidate, ~ configured
```

Copyright © 2010 Internetwork Expert

12.32 Auto-Install over LAN Interfaces using DHCP

- Configure the network to support auto-install as follows:
  - Upon booting R4 should request an IP address for its FastEthernet0/1 interface via DHCP.
  - R1 should forward this request onto R5.
  - R5 should reply with the IP address 155.X.146.4/24, the default gateway 155.X.146.1, the DNS server 155.X.146.6, and the TFTP server “RackXSW2”.
  - R4 should then ask R6 what its own hostname is, and download the config file “RackXR4-confg” from SW2 via TFTP.

Configuration

```
R6:
ip dns server
ip host Rack1SW2 155.1.58.8
ip host Rack1R4 155.1.146.4

R1:
interface FastEthernet 0/0
 ip helper-address 155.1.0.5

R5:
ip dhcp pool HOST_R4
   host 155.1.146.4 255.255.255.0
   client-identifier
0063.6973.636f.2d30.3030.372e.6562.6465.2e35.3632.322d.4574.302f.31
   default-router 155.1.146.1
   dns-server 155.1.146.6
   option 66 ascii "Rack1SW2"

SW2:
tftp-server flash:Rack1R4-confg
```

Copyright © 2010 Internetwork Expert

Verification

Note

AutoInstall tries to obtain an IP address and configuration by broadcasting DHCP/BOOTP/RARP requests over all LAN interfaces, using HDLC encapsulation on the first serial interface and the SLARP protocol to obtain an IP address, and by using Frame-Relay encapsulation on first serial interface and the BOOTP protocol to obtain an IP address.

On regular Ethernet LAN segments the AutoInstall process relies primarily on DHCP for obtaining an IP address and TFTP for downloading the configuration. The procedure is performed in the following stages.

First the router tries to obtain an IP address and subnet mask via DHCP or BOOTP by broadcasting DHCP/BOOTP requests out of all LAN interfaces. In addition to IP address information, the DHCP server can provide the default-router, DNS server list, TFTP server, and a boot-file name. In parallel to DHCP, IOS sends BOOTP queries, and may the learn TFTP server's name ("sname") or IP address ("siaddr") and boot-file name ("file") values from BOOTP.

Next the router tries to download its configuration using unicast transfer from the TFTP server learned via DHCP. If the DHCP/BOOTP server has specified the boot-file, the router tries to download this file as its configuration. Prior to version 12.1(5)T IOS only tried broadcast requests to reach the TFTP server and ignored the boot-file name learned via DHCP/BOOTP using generic names as described later.

Next, for the destination IP address of the TFTP server, the router either uses Option 150 or tries to resolve Option 66 using the DNS server list supplied by DHCP. If a BOOTP response is available with the "sname" and "siaddr" fields they could also be used. The order of preference is: "sname", "option 66", "option 150", and "siaddr". In case if the TFTP server's IP address could not be obtained, the router broadcasts download requests to the TFTP port. If the TFTP server does not respond to three unicast requests, the router falls back to broadcasts.

Copyright © 2010 Internetwork Expert

If the DHCP/BOOTP server did not provide a boot-file name, the router tries to download a network-wide configuration file named "network-confg" and then "cisconet.cfg" from the TFTP server (using either unicasts if the IP address of the TFTP server is known, or broadcasts). The purpose of network-wide configuration file is to allow the router to determine its hostname.

If the network-wide configuration has been successfully downloaded, the router looks into the **`ip host`** commands trying to find a match to its newly obtained IP address and map it to a hostname. If the hostname is successfully found, the router requests the host-specific configuration file "<hostname>-confg" via TFTP, and if this fails it requests "<hostname>.cfg". If the file has been downloaded, it is merged with the router's configuration.

If the hostname has not been determined the router tries to resolve its name via DNS (using either DNS servers or broadcasts) and downloads the respective configuration file. If the name lookup or host-specific configuration file download has failed, the router tries to download the generic configuration file named either "router-confg" or "router.cfg"

To verify this copy R4's configuration to SW2 via TFTP, then erase and reload R4.

```
Rack1R4#copy running-config flash:Rack1R4-confg
Destination filename [Rack1R4-confg]?
Erase flash: before copying? [confirm]n
Verifying checksum...  OK (0xC982)
1459 bytes copied in 8.033 secs (182 bytes/sec)

Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#tftp-server flash:Rack1R4-confg

Rack1SW2#copy tftp: flash:
Address or name of remote host []? 150.1.4.4
Source filename []? Rack1R4-confg
Destination filename [Rack1R4-confg]?
Accessing tftp://150.1.4.4/Rack1R4-confg...
Loading Rack1R6-confg from 150.1.4.4 (via Vlan58): !
[OK - 1459 bytes]
```

Copyright © 2010 Internetwork Expert

Now find out the Client-ID of R4's FastEthernet 0/1 interface so we know what client-identifier to put in R5's DHCP host pool. To accomplish this temporarily set up the respective interface for DHCP IP address assignment and use **show dhcp lease** command.

```
Rack1R4#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface FastEthernet0/1
Rack1R4(config-if)#ip address dhcp
Rack1R4(config-if)#end
Rack1R4#show dhcp lease
Temp IP addr: 0.0.0.0  for peer on Interface: FastEthernet0/1
Temp  sub net mask: 0.0.0.0
   DHCP Lease server: 0.0.0.0, state: 1 Selecting
   DHCP transaction id: 14F4
   Lease: 0 secs,  Renewal: 0 secs,  Rebind: 0 secs
   Next timer fires after: 00:00:01
   Retry count: 0   Client-ID: cisco-0007.ebde.5622-Et0/1
   Client-ID hex dump: 636973636F2D303030372E656264652E
                       353632322D4574302F31
   Hostname: Rack1R4

Rack1R4#show interfaces FastEthernet 0/1
FastEthernet0/1 is up, line protocol is up
  Hardware is AmdP2, address is 0007.ebde.5622 (bia 0007.ebde.5622)
```

Cisco uses the MAC address of the interface along with interface short name to create a client-identifier. In the Cisco IOS DHCP server the client-ID must be specified as hex value, starting with '00' (e.g. 0063.6973.636F.2D30.3030.372E.6562.6465.2E35.3632.322D.4574.302F.31).

When R4 is reset the DHCP request is received by R5, and the TFTP request is received by SW2.

```
Rack1R4#show start
startup-config is not present
Rack1R4#reload
Proceed with reload? [confirm]

SYS-5-RELOAD: Reload requested by console. Reload Reason: Reload
Command.

Rack1R5#debug  ip dhcp server events
Rack1R5#debug  ip dhcp server packet

Rack1SW2#debug tftp events
TFTP Event debugging is on

Rack1SW2#debug tftp packets
TFTP Packet debugging is on
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#
DHCPD: DHCPDISCOVER received from client
0063.6973.636f.2d30.3030.372e.6562.6465.2e35.3632.322d.4574.302f.31
through relay 155.1.146.1.
DHCPD: Seeing if there is an internally specified pool class:
  DHCPD: htype 1 chaddr 0007.ebde.5622
  DHCPD: circuit id 01f50000
DHCPD: Sending DHCPOFFER to client
0063.6973.636f.2d30.3030.372e.6562.6465.2e35.3632.322d.4574.302f.31
(155.1.146.4).
DHCPD: unicasting BOOTREPLY for client 0007.ebde.5622 to relay
155.1.146.1.
DHCPD: DHCPREQUEST received from client
0063.6973.636f.2d30.3030.372e.6562.6465.2e35.3632.322d.4574.302f.31.
DHCPD: Sending notification of ASSIGNMENT:
 DHCPD: address 155.1.146.4 mask 255.255.255.0
  DHCPD: lease time remaining (secs) = 4294967295
DHCPD: No default domain to append - abort update
DHCPD: Sending DHCPACK to client
0063.6973.636f.2d30.3030.372e.6562.6465.2e35.3632.322d.4574.302f.31
(155.1.146.4).
DHCPD: unicasting BOOTREPLY for client 0007.ebde.5622 to relay
155.1.146.1.

Rack1SW2#
TFTP: read request from host 155.1.146.4(56461) via Vlan58
TFTP: Opened flash:Rack1R4-confg, fd 0, size 1494
TFTP: Sending block 1 (retry 0), socket_id 0x2373918
TFTP: Received ACK for block 1, socket_id 0x2373918
TFTP: Sending block 2 (retry 0), socket_id 0x2373918
TFTP: Received ACK for block 2, socket_id 0x2373918
TFTP: Sending block 3 (retry 0), socket_id 0x2373918
TFTP: Received ACK for block 3, socket_id 0x2373918
TFTP: Finished flash:Rack1R4-confg, time 00:00:00
TFTP: Server request for port 54974, socket_id 0x3B89C38
TFTP: read request from host 155.1.146.4(54974) via Vlan58
TFTP: Opened flash:Rack1R4-confg, fd 0, size 1494
TFTP: Sending block 1 (retry 0), socket_id 0x3B89C38
TFTP: Received ACK for block 1, socket_id 0x3B89C38
TFTP: Sending block 2 (retry 0), socket_id 0x3B89C38
TFTP: Received ACK for block 2, socket_id 0x3B89C38
TFTP: Sending block 3 (retry 0), socket_id 0x3B89C38
TFTP: Received ACK for block 3, socket_id 0x3B89C38
TFTP: Finished flash:Rack1R4-confg, time 00:00:00
```

Copyright © 2010 Internetwork Expert

```
R4:
32K bytes of NVRAM.
32768K bytes of processor board System flash (Read/Write)

Translating "Rack1SW2"...domain server (155.1.146.6) [OK]

%Error opening tftp://155.1.58.8/network-confg (No such file or directory)
%Error opening tftp://155.1.58.8/cisconet.cfg (No such file or directory)
Domain server mapped address 155.1.146.4 to Rack1R4
Loading rack1r4-confg from 155.1.58.8 (via FastEthernet0/1): !
[OK - 1358 bytes]
Configuration mapped ip address 155.1.58.8 to Rack1SW2
 Slot is empty or does not support clock participate
 WIC slot is empty or does not support clock participate
%Error opening tftp://255.255.255.255/network-confg (Timed out)
%Error opening tftp://255.255.255.255/cisconet.cfg (Timed out)

         --- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]: no

Press RETURN to get started!

<snip>
*Mar  1 00:01:15.215: AUTOINSTALL: FastEthernet0/1 is assigned 155.1.146.4
*Mar  1 00:01:15.215: AUTOINSTALL: Obtain tftp server name Rack1SW2resolved to
155.1.58.8
*Mar  1 00:01:18.288: AUTOINSTALL: Obtain default router (opt 3) 155.1.146.1

<snip>
```

Rack1R4#show ip int brief

```
Interface                  IP-Address      OK? Method Status          Protocol
FastEthernet0/0            204.12.1.4      YES TFTP   up                  up
Serial0/0                  unassigned      YES TFTP   up                  up
Serial0/0.1                155.1.0.4       YES TFTP   up                  up
FastEthernet0/1            155.1.146.4     YES TFTP   up                  up
Serial0/1                  155.1.45.4      YES TFTP   up                  up
Loopback0                  150.1.4.4       YES TFTP   up                  up
```

12.33 Auto-Install over Frame-Relay

- Disable R1's link to VLAN 146.
- Configure R5 as a staging router for AutoInstall over Frame-Relay for R4 as follows:
    - R5 should assign R4 the IP address 155.X.0.4.
    - When R5 receives a TFTP request from R4 for the file "network-confg" it should be forwarded to SW2.
    - R4 should download this config from SW2 and learn its hostname RackXR4.
    - R4 should then download the config file "RackXR4-confg" from SW2.

Configuration

```
R1:
inteface FastEthernet 0/0
 shutdown

R5:
interface Serial 0/0
 ip helper-address 155.1.58.8

SW2:
tftp-server flash:rack1r4-confg
tftp-server flash:network-confg
```

Verification

Note

Auto-Install over Frame-Relay is tried after auto-install attempts on all LAN interfaces and after HDLC/SLARP has been tried. As with all auto-install attempts over Serial interfaces, only the first serial interface of the router is tried.

On the first try, the encapsulation used is HDLC and SLARP requests are sent to the other side. If this fails, Frame-Relay encapsulation is configured on the first serial interface (e.g. Serial0 or Serial 1/0), and BOOTP broadcast packets are sent on all active DLCIs. The source IP address used in BOOTP packets is "0.0.0.0". The staging router receiving BOOTP packets on the other side of the Frame-Relay cloud will generate BOOTP reply based on its own **`frame-relay map`** commands. In this case R5 sees a BOOTP requests arrive on DLCI 504, and this DLCI is mapped to the IP address 155.X.0.4. This causes R5 to send a BOOTP reply with the IP address 155.X.0.4.

Copyright © 2010 Internetwork Expert

After the IP address has been obtained on the Frame-Relay connection, the router broadcasts TFTP requests out of all available DLCIs using its newly obtained IP address as a source. The staging router should either forward those requests to a TFTP server using the **ip helper-address** or reply to the TFTP requests itself.

The new router will request the network-wide configuration file, followed by host-specific configuration file as described in the previous scenario.

Follow the procedure below to create network-wide and host-specific configuration files, and upload them to SW2. The files differ in that the network-specific file contains hostname to IP address mapping for R4.

```
Rack1R4#copy running-config flash:Rack1R4-confg
Destination filename [Rack1R4-confg]?
Erase flash: before copying? [confirm]n
Verifying checksum...  OK (0xC982)
1459 bytes copied in 8.033 secs (182 bytes/sec)

Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#tftp-server flash:Rack1R4-confg

Rack1SW2#copy tftp: flash:
Address or name of remote host []? 150.1.4.4
Source filename []? Rack1R4-confg
Destination filename [Rack1R4-confg]?
Accessing tftp://150.1.4.4/Rack1R4-confg...

Rack1R4(config)#ip host Rack1R4 155.1.0.4
Rack1R4(config)#end
Rack1R4#copy running-config flash:Rack1R4-confg
Destination filename [network-confg]?
Erase flash: before copying? [confirm]n
Verifying checksum...  OK (0xC982)
1459 bytes copied in 8.033 secs (182 bytes/sec)

Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#tftp-server flash:network-confg

Rack1SW2#copy tftp: flash:
Address or name of remote host []? 150.1.4.4
Source filename []? network-confg
Destination filename [network-confg]?
Accessing tftp://150.1.4.4/network-confg...
```

Copyright © 2010 Internetwork Expert

To ensure SW2 exports both configuration files via TFTP and erase R4's configuration and reload the router. Observe the debugging output on R5 and SW2 to see the auto-configuration process.

```
Rack1R5#debug  ip dhcp server events
Rack1R5#debug  ip dhcp server packet

Rack1SW2#debug tftp events
TFTP Event debugging is on

Rack1SW2#debug tftp packets
TFTP Packet debugging is on

Rack1R5#
DHCPD: BOOTREQUEST received from BOOTP client 0007.ebde.5621 on
interface Serial0/0.
DHCPD: there is no address pool for 155.1.0.5.

Rack1SW2#
TFTP: Server request for port 49515, socket_id 0x3B89420
TFTP: read request from host 155.1.0.4(49515) via Vlan58
TFTP: Opened flash:network-confg, fd 0, size 1361
TFTP: Sending block 1 (retry 0), socket_id 0x3B89420
TFTP: Received ACK for block 1, socket_id 0x3B89420
TFTP: Sending block 2 (retry 0), socket_id 0x3B89420
TFTP: Received ACK for block 2, socket_id 0x3B89420
TFTP: Sending block 3 (retry 0), socket_id 0x3B89420
TFTP: Received ACK for block 3, socket_id 0x3B89420
TFTP: Finished flash:network-confg, time 00:00:00
<snip>
TFTP: Opened flash:Rack1R4-confg, fd 0, size 1494
TFTP: Sending block 1 (retry 0), socket_id 0x3989B80
TFTP: Received ACK for block 1, socket_id 0x3989B80
TFTP: Sending block 2 (retry 0), socket_id 0x3989B80
TFTP: Received ACK for block 2, socket_id 0x3989B80
TFTP: Sending block 3 (retry 0), socket_id 0x3989B80
TFTP: Received ACK for block 3, socket_id 0x3989B80
TFTP: Finished flash:Rack1R4-confg, time 00:00:00
<snip>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.34 Auto-Install over LAN Interfaces using RARP

- Configure R1 to supply an IP address to R4 via RARP on VLAN 146 when it boots.
- SW2 should supply the network and host-specific configuration files to R4.

Configuration

```
R1:
!
! ARP entry for R4's VLAN 146 interface
!
arp 155.1.146.4 0007.ebde.5622 arpa
!
inteface FastEthernet 0/0
 no shutdown
 ip rarp-server 155.1.146.1
 ip helper-address 155.1.58.8

SW2:
tftp-server flash:rack1r4-confg
tftp-server flash:network-confg
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> RARP is a simple alternative to using DHCP/BOOTP for IP address configuration on the LAN interface. A router sends out RARP requests for an IP address after it fails to obtain an address via DHCP/BOOTP. Another IOS router could be configured as RARP server, responding to a MAC-to-IP mapping request from a configured static ARP entry. After obtaining an IP address the router may proceed with TFTP broadcasts for network and host specific configuration files. Another router on the same subnet could relay the requests to the TFTP server.
>
> Follow the procedure below to create a network-wide and route-specific configuration files and upload them to SW2. The files differ in that network-specific file contains hostname to IP address mapping for R4, as in the previous example.

```
Rack1R4#copy running-config flash:Rack1R4-confg
Destination filename [Rack1R4-confg]?
Erase flash: before copying? [confirm]n
Verifying checksum...  OK (0xC982)
1459 bytes copied in 8.033 secs (182 bytes/sec)

Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#tftp-server flash:Rack1R4-confg

Rack1SW2#copy tftp: flash:
Address or name of remote host []? 150.1.4.4
Source filename []? Rack1R4-confg
Destination filename [Rack1R4-confg]?
Accessing tftp://150.1.4.4/Rack1R4-confg...

Rack1R4(config)#no ip host Rack1R4
Rack1R4(config)#ip host Rack1R4 155.1.146.4
Rack1R4(config)#end
Rack1R4#copy running-config flash:Rack1R4-confg
Destination filename [network-confg]?
Erase flash: before copying? [confirm]n
Verifying checksum...  OK (0xC982)
1459 bytes copied in 8.033 secs (182 bytes/sec)

Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#tftp-server flash:network-confg

Rack1SW2#copy tftp: flash:
Address or name of remote host []? 150.1.4.4
Source filename []? network-confg
Destination filename [network-confg]?
Accessing tftp://150.1.4.4/network-confg...
```

Copyright © 2010 Internetwork Expert

To ensure SW2 exports both configuration files via TFTP erase R4's configuration and reload the router. Observe the debugging output on R1 and SW2 to see the auto-configuration process.

```
Rack1SW2#debug tftp events
TFTP Event debugging is on

Rack1SW2#debug tftp packets
TFTP Packet debugging is on

Rack1R1#debug arp
ARP packet debugging is on

Rack1R1#
RARP: Rcvd RARP req for 0007.ebde.5622
RARP: Sending reply out FastEthernet0/0
IP RARP: sent rev-rep src 155.1.146.1 000d.bc04.e340,
                dst 155.1.146.4 0007.ebde.5622 FastEthernet0/0

Rack1SW2#
TFTP: read request from host 155.1.146.4(54757) via Vlan58
TFTP: Opened flash:network-confg, fd 0, size 1440
TFTP: Sending block 1 (retry 0), socket_id 0x3B89500
TFTP: Received ACK for block 1, socket_id 0x3B89500
TFTP: Sending block 2 (retry 0), socket_id 0x3B89500
TFTP: Received ACK for block 1, socket_id 0x3B89500
TFTP: Sending block 2 (retry 1), socket_id 0x3B89500
TFTP: Received ACK for block 2, socket_id 0x3B89500
TFTP: Sending block 3 (retry 0), socket_id 0x3B89500
TFTP: Received ACK for block 2, socket_id 0x3B89500
TFTP: Sending block 3 (retry 1), socket_id 0x3B89500
TFTP: Received ACK for block 3, socket_id 0x3B89500
TFTP: Finished flash:network-confg, time 00:00:08
<snip>
TFTP: Opened flash:Rack1R4-confg, fd 0, size 1494
TFTP: Sending block 1 (retry 0), socket_id 0x3989B80
TFTP: Received ACK for block 1, socket_id 0x3989B80
TFTP: Sending block 2 (retry 0), socket_id 0x3989B80
TFTP: Received ACK for block 2, socket_id 0x3989B80
TFTP: Sending block 3 (retry 0), socket_id 0x3989B80
TFTP: Received ACK for block 3, socket_id 0x3989B80
TFTP: Finished flash:Rack1R4-confg, time 00:00:00
<snip>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.35 IOS Menus

- When a user named OPERATOR telnets into R1 and authenticates with the password of CISCO it should see the following menu:

```
Operator Menu


    1           Display IP Routing Table

    2           Display Running Config

    3           Escape to Shell

    4           Disconnect
```

- Option 1 should show the routing table.
- Option 2 should show the running configuration.
- Option 3 should exit to the exec prompt.
- Option 4 should exit the telnet session.
- Clear the screen when the menu starts.

Configuration

```
R1:
menu OPERATOR title #
Operator Menu
#
menu OPERATOR text 1 Display IP Routing Table
menu OPERATOR command 1 show ip route
menu OPERATOR text 2 Display Running Config
menu OPERATOR command 2 show run
menu OPERATOR text 3 Escape to Shell
menu OPERATOR command 3 menu-exit
menu OPERATOR text 4 Disconnect
menu OPERATOR command 4 exit
menu OPERATOR clear-screen
!
username OPERATOR autocommand menu OPERATOR
username OPERATOR password CISCO
username OPERATOR privilege 15
!
line vty 0 4
 login local
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Menus allow simple automation of many network-monitoring tasks for unskilled/untrusted personnel. Each menu selection is associated with a command which is executed when a user chooses the selection. The command is executed with privileges of the local user who executed the menu command, or the privileges of the terminal line if the menu command is associated with a line. Therefore if the menu uses privileged commands (e.g. level 15), ensure they are available in the privilege level associated with the user/line.
>
> The special menu command **menu-exit** allows a user to leave the menu but to not disconnect from the line. That means once this command is executed the user is put back into the exec shell.

```
Rack1R1#telnet 150.1.1.1
Trying 150.1.1.1 ... Open


User Access Verification

Username: OPERATOR
Password: CISCO

Operator Menu


    1           Display IP Routing Table

    2           Display Running Config

    3           Escape to Shell

    4           Disconnect
```

```
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

R    204.12.1.0/24 [120/1] via 155.1.146.4, 00:00:06, FastEthernet0/0
     155.1.0.0/24 is subnetted, 15 subnets
C       155.1.146.0 is directly connected, FastEthernet0/0
R       155.1.23.0 [120/1] via 155.1.13.3, 00:00:06, Serial0/1
R       155.1.10.0 [120/3] via 155.1.0.5, 00:00:01, Serial0/0
R       155.1.8.0 [120/2] via 155.1.0.5, 00:00:01, Serial0/0
R       155.1.9.0 [120/3] via 155.1.146.6, 00:00:02, FastEthernet0/0
                  [120/3] via 155.1.13.3, 00:00:06, Serial0/1
C       155.1.13.0 is directly connected, Serial0/1
C       155.1.0.0 is directly connected, Serial0/0
R       155.1.7.0 [120/2] via 155.1.146.6, 00:00:02, FastEthernet0/0
                  [120/2] via 155.1.13.3, 00:00:06, Serial0/1

Operator Menu


    1           Display IP Routing Table

    2           Display Running Config

    3           Escape to Shell

    4           Disconnect


Building configuration...

Current configuration : 1989 bytes
!
! Last configuration change at 23:15:46 UTC Tue Jul 29 2008
! NVRAM config last updated at 22:03:31 UTC Tue Jul 29 2008
!
version 12.4
service timestamps debug datetime msec
<snip>

Operator Menu


    1           Display IP Routing Table

    2           Display Running Config

    3           Escape to Shell

    4           Disconnect
Rack1R1#
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.36 IOS Banners

- Change the default IOS banners on R3 as follows:
  - All connecting users, with the exception to console users, should see the notification message “Welcome to IOS Router”.
  - Before the login prompt is displayed users should see the message “Please, authenticate yourself”.
  - Before a user can see the shell prompt, display “Hi, you are on the line <Line> have a nice time at <Hostname>” where <Line> is the line number the user is connected to and <Hostname> is the current router’s hostname.
  - Set the banner for reverse telnet connections to “This is a reverse telnet connection”.

Configuration

```
R3:
banner motd #
Welcome to IOS Router
#
banner login #
Please authenticate yourself
#
banner exec #
Hi, you are on the line $(line), have a nice time at $(hostname)
#
banner incoming #
This is a reverse telnet connection
#
line console 0
 no motd-banner
 no exec-banner
```

Copyright © 2010 Internetwork Expert

Verification

Note

For remote access and local console connections the following types of banners can be displayed:

1) MOTD banner – displayed first to everyone connecting to the router
2) Login banner – displayed before the authentication prompt is presented to the user (if login is disabled, the banner is not displayed)
3) Exec-banner – displayed before the exec prompt is displayed to the user
4) Incoming-banner – displayed to users connecting via reverse telnet

Special tokens can be used the banner messages such as $(line) or $(hostname) to create dynamic notifications. Some lines can be selectively exempted from viewing banner messages, such as the console is in this example.

```
Rack1R3#telnet 150.1.3.3
Trying 150.1.3.3 ... Open

Welcome to IOS Router

Please authenticate yourself


User Access Verification

Password: cisco
Hi, you are on the line 130, have a nice time at Rack1R3

Rack1R3>
```

Login to R3 via the console and type **exit**. Login again to check the banners. Note that only the login banner is displayed.

```
Rack1R3#exit

<snip>

Welcome to IOS
```

Copyright © 2010 Internetwork Expert

Simulate a reverse telnet connection to R3's AUX port to verify the incoming-banner message. To do so, configure the AUX line to accept incoming telnet connections. In the below example the AUX port is line 129.

```
R3:
line aux 0
 no modem inOut
 transport input telnet
```

```
Rack1R3# show line
   Tty Typ     Tx/Rx    A Modem  Roty AccO AccI   Uses   Noise  Overruns   Int
*    0 CTY              -    -      -    -    -      6       0     0/0       -
   129 AUX    9600/9600  -    -      -    -    -      1       0     0/0       -
   130 VTY               -    -      1    -    -      5       0     0/0       -
   131 VTY               -    -      -    -    -      0       0     0/0       -
   132 VTY               -    -      -    -    -      0       0     0/0       -
   133 VTY               -    -      -    -    -      0       0     0/0       -
   134 VTY               -    -      -    -    -      0       0     0/0       -

Line(s) not in async mode -or- with no hardware support:
1-128
```

```
Rack1R3#telnet 150.1.3.3 6129
Trying 150.1.3.3, 6129 ... Open

Welcome to IOS Router

This is a reverse telnet connection
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.37 KRON Command Schedule

- Configure a KRON occurrence on R4 that saves the running config to a remote TFTP server daily at 08:00.
- Use the server 155.1.146.100 and the filename “r3-config”

Configuration

```
R3:
kron occurrence SAVE_DAILY at 8:00 recurring
policy-list SAVE_CONFIG
!
kron policy-list SAVE_CONFIG
cli show running-config | redirect tftp://155.1.146.100/r3-config
```

Verification

> **Note**
>
> The KRON Command Scheduler allows exec commands or TCL scripts to run at a specified time. The KRON configuration consists of a policy-list containing exec commands, and a scheduler to execute the commands in the policy-list at a specified time or recurring interval.

```
Rack1R3#debug kron all
All kron debug flags are on

Rack1R3#clock set 07:59:59 Jan 1 2010
Jan  1 07:59:59.000: Clock Set Seen
Jan  1 07:59:59.000: %SYS-6-CLOCKUPDATE: System clock has been updated
from 08:00:21 UTC Mon Jan 1 2001 to 07:59:59 UTC Fri Jan 1 2010,
configured from console by console.
Jan  1 07:59:59.000: Major 4, Minor 9
Jan  1 07:59:59.000: Start timer for SAVE_DAILY, expires in 60000 Msec

Rack1R3#show kron schedule
Kron Occurrence Schedule
SAVE_DAILY inactive, will run again in 0 days 00:00:05 at 8 :00 on

Rack1R3#show kron schedule
Kron Occurrence Schedule
SAVE_DAILY   active, will run again in 0 days 23:59:59 at 8 :00 on

Jan  1 08:00:59.002: Major 1, Minor 0
Jan  1 08:00:59.002: Timer Event SAVE_DAILY
Jan  1 08:00:59.002: Kron delay for next SAVE_DAILY 86400000
Jan  1 08:00:59.006: Call parse_cmd 'show running-config | redirect
tftp://155.1.146.100/r3-config'
Jan  1 08:01:24.083: Kron CLI return 0
<snip>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.38 EEM Scripting: Interface Events

- Create an EEM applet in R5 that send an email message to noc@INE.com from r5@ine.com once the Frame-Relay interface receive load exceeds 50% of interface bandwith.
- Use the SMTP server at the IP address 155.X.146.100.
- After this, apply the access-list CRITICAL_TRAFFIC_ONLY ingress on R5's connection to the Frame-Relay interface.
- The access-list should be pre-configured and permit just the HTTP and DNS traffic responses.
- The interface bandwidth is 64Kbps.

Configuration

Note

Embedded Event Manager is a feature of Cisco IOS operating system that allows you writing handlers for various system events. The core of EEM is a special process known as EEM server that acts as a middleware agent between event detectors and EEM event subscribers. There is a fixed amount of event detectors that post an event when a programmed condition is met. They include the following most important:

1) CLI event detector – detects various commands typed in CLI based on regular expression matching.
2) Syslog event detector – responds to various syslog stings, allowing for matching on regular expressions just like CLI detector.
3) Interface Counter – responds to various interfaces counters crossing threshold settings.
4) Counter – responds to the change of value of a generic system counter.
5) SNMP – monitors specified SNMP objects and post an event upon the programmed condition met.
6) None – this is a special case of event detector triggered when a user issues the command **`event manager run`** to execute a named EEM script/applet.
7) Watchdog – generates periodic timer events and allows the EEM script to be run at repeating time intervals.

Copyright © 2010 Internetwork Expert

The other part of the EEM system – event subscribers – are defined and registered with EEM server as either EEM applets or scripts. Applets are simple programs written using a very basic set of CLI commands that start with an **action** keyword. Scripts are special TCL scripts written to handle EEM events. The applets are easy to write and they are powerful enough to perform many functions including CLI commands, email sending, SNMP/Syslog message generation, implementing basic program logic (e.g. branching or computations). The TCL scripts are more powerful but require more skills to develop. In this workbook will be dealing with EEM applets only.

Every EEM applet has a name and a detector condition defined to trigger the applet. The applet may access global variables defined using the command **event manager environment** or the parameters passed to them by an event detector. There are pre-defined environment variables accessible to the EEM scripts, specific to every event detector. You may find a full list by reading the document "Writing Embedded Event Manager Policies Using Cisco IOS CLI" in the Cisco IOS documentation set. You may also list the variables for an event detector using the command **show event manager detector <NAME> detailed**. The same document describes the most common actions used with EEM applets. For the detailed description of every **action** that could be used in an EEM applet, refer to the "IOS Network Management Command Reference" list and locate the action commands there.

Notice that every applet definition starts with a group of **event** commands that specify the event detector and the condition to trigger the applet. Normally there is only one event defined for an applet – if you define multiple events you need to further specify how they co-relate to each other (covered in separate task). For the complete list of events supported by EEM with various event detectors refer to the "IOS Network Management Command Reference" document and locate the **event** commands there.

Copyright © 2010 Internetwork Expert

R5:

```
!
! Notice that the access-list also permits routing traffic and
! the returning STMP packets to allow email sending.
!
ip access-list extended CRITICAL_TRAFFIC
 permit tcp any eq 80 any
 permit udp any eq 43 any
 permit tcp any eq 25 any
 permit udp any any eq rip
!
!
!
no event manager applet INTERFACE_LOAD
event manager applet INTERFACE_LOAD
!
! Notice that the event defined below show take only a single line
! Use CLI help function to enter all the parameter values
!
! The condition specifies to monitor R5's FR interface "TXLoad"
! counter (range 1-255) and post an event when the value crosses
! 153 (60%*255).
!
 event tag 1 interface name Serial0/0/0 parameter rxload entry-op gt
entry-val 153 entry-type value poll-interval 30

!
! The first action set applies the  access-list
!
 action 0.0 cli command "enable"
 action 1.0 cli command "conf t"
 action 2.0 cli command "interface Serial 0/0/0"
 action 3.0 cli command "ip access-group CRITICAL_TRAFFIC in"
 action 4.0 mail server "155.1.146.100" to "noc@INE.com" from
"r5@INE.com" subject "Interface Alert" body "Interface Serial 0/0/0
over 60% RX Load"
!
! We set the load interval to the minimum value to achieve faster
! response of the interface counters.
!
interface Serial 0/0/0
 bandwidth 64
 load-interval 30
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> We registered the applet with the "none" event detector to be able to run the applet from the CLI for "dry-run" testing purposes. Enable EEM debugging commands to track the CLI and E-Mail actions and run the applet manually first:

```
R5:
event manager applet INTERFACE_LOAD
 event tag 2.0 none

Rack1R5#debug event manager action cli
Debug EEM action cli debugging is on

Rack1R5#debug event manager action mail
Debug EEM action mail debugging is on

Rack1R5#event manager run INTERFACE_LOAD

%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : CTL : cli_open
called.
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : OUT : Rack1R5>
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : IN  : Rack1R5>enable
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : OUT : Rack1R5#
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : IN  : Rack1R5#conf t
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : OUT : Enter
configuration commands, one per line.  End with CNTL/Z.
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : OUT :
Rack1R5(config)#
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : IN  :
Rack1R5(config)#interface Serial 0/0/0
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : OUT : Rack1R5(config-
if)#
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : IN  : Rack1R5(config-
if)#ip access-group CRITICAL_TRAFFIC in
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : OUT : Rack1R5(config-
if)#
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(smtp_lib) : smtp_connect_attempt:
1
Rack1R5#
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(smtp_lib) : smtp_connect connect
fail 260
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(smtp_lib) : fh_smtp_connect
callback timer is awake
%HA_EM-3-FMPD_SMTP: Error occurred when sending mail to SMTP server:
155.1.146.100 : timeout error
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : CTL : cli_close
called.
Rack1R5#
%SYS-5-CONFIG_I: Configured from console by vty0
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip interface serial 0/0/0 | include access
  Outgoing access list is not set
  Inbound  access list is CRITICAL_TRAFFIC
  IP access violation accounting is disable

R5:
event manager applet INTERFACE_LOAD
 no event tag 2.0 none
```

> **Note**
>
> Notice that the configuration portion of the script succeeds, while the SMTP connection cannot be established to send out an email, as there SMTP server does not exist. Now attempt a "real" test by simulating interface oversubscription. First, make sure the applet is registered with the event detector:

```
Rack1R5#show event manager policy registered event-type interface
No.  Class     Type    Event Type          Trap  Time Registered
Name
1    applet    user    interface           Off   Sat Dec 12 18:51:49
2009  INTERFACE_LOAD
 1: interface: name {Serial0/0/0} parameter {rxload} entry_op gt
entry_val 153 entry_type value poll_interval 30.000
 maxrun 20.000
 action 0.0 cli command "enable"
 action 1.0 cli command "conf t"
 action 2.0 cli command "interface Serial 0/0/0"
 action 3.0 cli command "ip access-group CRITICAL_TRAFFIC in"
 action 4.0 mail server "155.1.146.100" to "noc@INE.com" from
"r5@INE.com" subject "Interface Alert" body "Interface Serial 0/0/0
over 60% RX Load"
```

Now flood R5's Frame-Relay interface:

```
Rack1R3#ping 155.1.0.5 repeat 1000000 size 1000 timeout 0

Type escape sequence to abort.
Sending 1000000, 1000-byte ICMP Echos to 155.1.0.5, timeout is 0
seconds:
 ..........................................................................
 ..........................................................................
 ..........................................................................
 ..........................................................................
 ..........................................................................
```

Add one extra debugging command in R5 to track interface detector events and notice how the applet is triggered:

Copyright © 2010 Internetwork Expert

```
Rack1R5#debug event manager detector interface
Debug EEM Interface Event Detector debugging is on

Rack1R5#
fh_fd_intf_param_fetch:I/F=Serial0/0/0 rxload=251
fh_fd_intf_start_poll_timer: start_t=30000
fh_fd_intf_event_register: re=0x66BD3658, sid=17
fh_send_intf_fd_msg: sval=0
fh_fd_intf_process_async
fh_fd_intf_param_fetch:I/F=Serial0/0/0 rxload=251
intf_value_uint_compare:op1=251 op2=153 ret=TRUE
intf_entry_value_check:Returning TRUE
fh_fd_intf_event_match: re = 0x66BD3658, num_matches = 1

fh_fd_intf_start_poll_timer: start_t=30000
fh_fd_intf_process_poll_timer: update_t=0
fh_send_intf_fd_msg: msg_type=64
fh_send_intf_fd_msg: sval=0
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : CTL : cli_open
called.
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : OUT : Rack1R5>
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : IN  : Rack1R5>enable
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : OUT : Rack1R5#
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : IN  : Rack1R5#conf t
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : OUT : Enter
configuration commands, one per line.  End with CNTL/Z.
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : OUT : %HA_EM-6-LOG:
INTERFACE_LOAD : DEBUG(cli_lib) : : IN  : Rack1R5(config)#interface
Serial 0/0/0
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : OUT : Rack1R5(config-
if)#
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : IN  : Rack1R5(config-
if)#ip access-group CRITICAL_TRAFFIC in
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(cli_lib) : : OUT : Rack1R5(config-
if)#
%HA_EM-6-LOG: INTERFACE_LOAD : DEBUG(smtp_lib) : smtp_connect_attempt:
1
fh_fd_intf_process_async
fh_fd_intf_param_fetch:I/F=Serial0/0/0 rxload=255
intf_exit_comb_check:Returning TRUE
intf_value_uint_compare:op1=255 op2=153 ret=TRUE
intf_entry_value_check:Returning TRUE
fh_fd_intf_event_match: re = 0x66BD3658, num_matches = 1
```

Copyright © 2010 Internetwork Expert

12.39 EEM Scripting: Syslog Events

- Create an EEM script in R5 that applies the “no shutdown” command every time someone attempts to shutdown the interface Serial 0/0/0.
- Send an email containing the result of the command “show users” to the address admin@INE.com if this event occurs.
- Make sure your script does not apply to any other interface.

Configuration

Note

In this example we deal with the Syslog event detector. When the interface has been shutdown, the following message will be posted to syslog:

```
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to
administratively down
```

We create an event subscriber to intercept this message and apply the no shutdown command to the interface. Notice the regular expression used to match the Syslog messages. Also, pay attention to the CLI command flow – it should be exactly the same set you would use to configure something manually.

R5:

```
no event manager applet INTERFACE_SHUTDOWN
event manager applet INTERFACE_SHUTDOWN
 event tag 1.0 syslog pattern "Interface Serial0/0/0.*changed.*down"
 action 1.0 cli command "enable"
 action 2.0 cli command "conf t"
 action 3.0 cli command "interface Serial 0/0/0"
 action 4.0 cli command "no shutdown"
 action 5.0 cli command "end"
 action 6.0 cli command "show users"
 action 7.0 mail server "155.1.146.100" to "admin@INE.com" from
"r5@INE.com" subject "Interface Shutdown Alert" body "Interface Serial
0/0/0 unshut, current users $_cli_result"
```

Copyright © 2010 Internetwork Expert

Verification

Note

Start by making sure your applet is registered with the EEM server:

```
Rack1R5#show event manager policy registered
No.  Class     Type    Event Type          Trap  Time Registered
Name
1    applet    user    interface           Off   Sat Dec 12 18:51:49
2009  INTERFACE_LOAD
 1: interface: name {Serial0/0/0} parameter {rxload} entry_op gt
entry_val 153 entry_type value poll_interval 30.000
 maxrun 20.000
 action 0.0 cli command "enable"
 action 1.0 cli command "conf t"
 action 2.0 cli command "interface Serial 0/0/0"
 action 3.0 cli command "ip access-group CRITICAL_TRAFFIC in"
 action 4.0 mail server "155.1.146.100" to "noc@INE.com" from
"r5@INE.com" subject "Interface Alert" body "Interface Serial 0/0/0
over 60% RX Load"

2    applet    user    syslog              Off   Sat Dec 12 21:26:10
2009  INTERFACE SHUTDOWN
 1.0: syslog: pattern {Interface Serial0/0/0.*changed.*down}
 maxrun 20.000
 action 1.0 cli command "enable"
 action 2.0 cli command "conf t"
 action 3.0 cli command "interface Serial 0/0/0"
 action 4.0 cli command "no shutdown"
 action 5.0 cli command "end"
 action 6.0 cli command "show users"
 action 7.0 mail server "155.1.146.100" to "admin@INE.com" from
"r5@INE.com" subject "Interface Shutdown Alert" body "Interface Serial
0/0/0 unshut, current users $_cli_result"
```

Now enable EEM action and detector debugging and shutdown the Frame-Relay interface. Observe the event being detected and the applet triggered:

```
Rack1R5#debug event manager detector syslog
Rack1R5#debug event manager action cli
Rack1R5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface serial 0/0/0
Rack1R5(config-if)#shutdown

%LINK-5-CHANGED: Interface Serial0/0/0, changed state to
administratively down

syslog_pubinfo_enqueue: matched pattern="Interface
Serial0/0/0.*changed.*down", matched message="
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to
administratively down"
```

Copyright © 2010 Internetwork Expert

```
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : CTL : cli_open
called.
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT : Rack1R5>
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : IN  :
Rack1R5>enable
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT : Rack1R5#
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : IN  :
Rack1R5#conf t
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT : Enter
configuration commands, one per line.  End with CNTL/Z.
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT :
Rack1R5(config)#
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : IN  :
Rack1R5(config)#interface Serial 0/0/0
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT :
Rack1R5(config-if)#
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : IN  :
Rack1R5(config-if)#no shutdown
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT :
Rack1R5(config-if)#
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : IN  :
Rack1R5(config-if)#end
%SYS-5-CONFIG_I: Configured from console by  on vty0
(EEM:INTERFACE_SHUTDOWN)
fh_fd_syslog_event_match: num_matches = 0
fh_fd_data_syslog: num_matches = 0
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT : Rack1R5#
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : IN  :
Rack1R5#show users
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT :     Line
User       Host(s)              Idle       Location
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT :    0 con 0
idle                 00:00:02
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT : *194 vty 0
Rack1R5(config-if)#               idle                 00:00:00
EEM:INTERFACE_SHUTDOWN
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT :
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT :   Interface
User               Mode         Idle     Peer Address
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT :
%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : OUT : Rack1R5#
%LINK-3-UPDOWN: Interface Serial0/0/0, changed state to up

%HA_EM-3-FMPD_SMTP: Error occurred when sending mail to SMTP server:
155.1.146.100 : timeout error

%HA_EM-6-LOG: INTERFACE_SHUTDOWN : DEBUG(cli_lib) : : CTL : cli_close
called.
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.40 EEM Scripting: CLI Events

- Configure R5 so that every time someone types the **show run** command it does not display the usernames.

Configuration

> **Note**
>
> In this example we learn about the CLI event detector. This detector allows for monitoring certain CLI patterns and publishes an event if a match occurs. The event monitoring configuration could use the parameter sync set to either "yes" or "no" to define a synchronous or asynchronous applet. When the applet event condition is synchronous, the EEM server will hold the matched CLI command execution until the script terminates. The script should return an exit value in the variable **$_exit_status** and this will determine whether the triggered command will run (status 1) or not (status 0). Asynchronous event handlers will let the CLI command execute and the event will be posted after that. The script cannot affect the command execution. Notice that asynchronous CLI events require a set of additional parameters, such as number of occurrences and the time window for the occurrences.
>
> Another command we use in the script below is the puts action. It allows you displaying arbitrary text on the console, provided that the script is synchronous.

```
R5:
no event manager applet SHOW_RUN_FILTER
event manager applet SHOW_RUN_FILTER
 event tag 1.0 cli pattern "show run" sync yes
 action 1.0 cli command "enable"
 action 2.0 cli command "show run | exclude username"
 action 3.0 puts $_cli_result
!
! Exit status of 0 block execution of the original command
!
 action 4.0 set $_exit_status 0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Start by making sure your applet is registered with the EEM server:

```
Rack1R5#show event manager policy registered event-type cli
No.  Class     Type    Event Type          Trap  Time Registered
Name
1    applet    user    cli                 Off   Sat Dec 12 21:47:48
2009  SHOW_RUN_FILTER
 1.0: cli: pattern {show run} sync yes
 maxrun 20.000
 action 1.0 cli command "enable"
 action 2.0 cli command "show run | exclude username"
 action 3.0 puts "$_cli_result"
 action 4.0 set $_exit_status "0"
```

Now enable EEM action and detector debugging and execute the command show running-config. Notice that the output will not contain the usernames, as the actual command has been substituted with a “filtered” version.

```
Rack1R5#debug event manager detector cli
Rack1R5#debug event manager action cli

Rack1R5#show run

Building configuration...

Current configuration : 5325 bytes
!
version 12.4
no service timestamps debug uptime
no service timestamps log uptime
no service password-encryption
!
hostname Rack1R5
!
boot-start-marker
boot-end-marker
!
logging message-counter syslog
enable password cisco
!
no aaa new-model
dot11 syslog
ip source-route
```

Copyright © 2010 Internetwork Expert

```
check_eem_cli_policy_handler: command_string=show running-config
check_eem_cli_policy_handler: num_matches = 1, response_code = 4
fh_send_cli_fd_msg: msg_type=64
fh_send_cli_fd_msg: sval=0
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : CTL : cli_open
called.
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : Rack1R5>
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : IN  : Rack1R5>enable
cli_history_entry_add: free_hist_list size=0, hist_list size=7
eem_no_scan flag set, skipping scan of
command_string=check_eem_cli_policy_handler
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : Rack1R5#
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : IN  : Rack1R5#show
run | exclude username
cli_history_entry_add: free_hist_list size=0, hist_list size=7
eem_no_scan flag set, skipping scan of
command_string=check_eem_cli_policy_handler
fh_send_cli_fd_msg: msg_type=114
fh_send_cli_fd_msg: sval=0
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : Building
configuration...
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT :
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : Current
configuration : 5325 bytes
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : !
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : version 12.4
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : no service
timestamps debug uptime
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : no service
timestamps log uptime
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : no service
password-encryption
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : !
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : hostname
Rack1R5
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : !
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : boot-start-
marker
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : boot-end-
marker
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : !
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : logging
message-counter syslog
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : enable
password cisco
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : !
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : no aaa new-
model
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : dot11 syslog
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : OUT : ip source-
route
%HA_EM-6-LOG: SHOW_RUN_FILTER : DEBUG(cli_lib) : : CTL : 20+ lines read
from cli, debug output truncated
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

12.41 EEM Scripting: Periodic Scheduling

- Configure R5 to send a copy of the running configuration every 5 minutes to noc@INE.com.
- Log a syslog message "Configuration saved" every time the script does that.
- Use the same mail server as in the previous tasks.

Configuration

> **Note**
>
> Event applets may be configured to respond to periodic or timed events, e.g. fire every time interval or start at a fixed time in the future. In this example we create a script that registers with the watchdog event publisher and rung every 300 seconds to save and email the running configuration.

```
R5:
no event manager applet SHOW_RUN_EVERY_5MIN
event manager applet SHOW_RUN_EVERY_5MIN
 event tag 1.0 timer watchdog time 300
!
! We use write term as we intercepted show run in the previous task
!
 action 1.0 cli command "enable"
 action 2.0 cli command "write term"
 action 3.0 syslog msg "Configuration Saved"
 action 4.0 mail server "155.1.146.100" to "noc@INE.com" from
"r5@INE.com" subject "Configuration" body " $_cli_result"
```

Verification

> **Note**
>
> Start by making sure your applet is registered with the EEM server:

```
Rack1R5#show event manager policy registered event-type timer-watchdog
No.  Class     Type    Event Type          Trap  Time Registered
Name
1    applet    user    timer watchdog      Off   Sat Dec 12 22:49:51
2009  SHOW_RUN_EVERY_5MIN
 1.0: timer watchdog: time 300.000
 maxrun 20.000
 action 1.0 cli command "enable"
 action 2.0 cli command "write term"
 action 3.0 syslog msg "Configuration Saved"
 action 4.0 mail server "155.1.146.100" to "noc@INE.com" from
"r5@INE.com" subject "Configuration" body " $_cli_result"
```

Copyright © 2010 Internetwork Expert

Now enable EEM detector debugging and wait about five minutes for the event to be published:

```
Rack1R5#debug event manager detector timer
Debug EEM Timer Event Detector debugging is on

fh_fd_timer_process_async
fh_fd_timer_event_expire: re=0x66BD3018
fh_send_timer_fd_msg: msg_type=64
fh_send_timer_fd_msg: sval=0
Rack1R5#
%HA_EM-6-LOG: SHOW_RUN_EVERY_5MIN: Configuration Saved
Rack1R5#
%HA_EM-3-FMPD_SMTP: Error occurred when sending mail to SMTP server:
155.1.146.100 : timeout error
Rack1R5# _fd_timer_process_async
cron_tick: num_matches 0
```

Copyright © 2010 Internetwork Expert

12.42 EEM Scripting: Advanced Features

- Create an EEM applet in R5 that clears all Serial/FastEthernet interface counters every 3 minutes.
- Ensure that the Serial interfaces counters are never cleared though.
- Account for any interfaces added to the router in the future.

Configuration

Note

Below is an example of an advanced script that use loops and conditional constructs. You may see the well-known foreach and if/else blocks being implemented for EEM applets. A fewother interesting action are

1) **`info type interface-name`** – pulls out the list of the interface names in the system matching a regular expression.
2) **`regexp PATTERN STRING`** performs regular expression match against a string. If there is a match, the variable $_regexp_result is set to "1". This command takes extra arguments for extracting the matched substrings.
3) **`continue`** – classic structured programming command, starts another interation.
4) **`cli command COMMAND pattern PATTERN`**. This syntax allows for the CLI commands that await a user's response, e.g. a confirmation. The PATTERN is the regular expression matching the command prompt.

R5:

```
no event manager applet CLEAR_COUNTERS
event manager applet CLEAR_COUNTERS
 event tag 1.0 timer watchdog time 180
 action 1.0 info type interface-names regexp "Fa|Se"
 action 2.0 foreach _iface $_info_interface_names
 action 3.0 regexp ".*(Serial).*" $_iface
 action 4.0 if $_regexp_result eq "1"
 action 5.0 continue
 action 6.0 else
 action 7.0 cli command "enable"
 action 8.0 cli command "clear counters $_iface" pattern "confirm"
 action 9.0 cli command "y"
 action 9.1 end
 action 9.2 end
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Complex scripts are better be debugged by associating with the “none” event detector. You may then enable various event manager debugging and run the script as many times as you want.

```
R5:
no event manager applet CLEAR_COUNTERS
event manager applet CLEAR_COUNTERS
 event tag 1.0 none
 action 1.0 info type interface-names regexp "Fa|Se"
 action 2.0 foreach _iface $_info_interface_names
 action 3.0 regexp ".*(Serial).*" $_iface
 action 4.0 if $_regexp_result eq "1"
 action 5.0 continue
 action 6.0 else
 action 7.0 cli command "enable"
 action 8.0 cli command "clear counters $_iface" pattern "confirm"
 action 9.0 cli command "y"
 action 9.1 end
 action 9.2 end

Rack1R5#debug event manager action cli
Debug EEM action cli debugging is on
```

Notice that only the FastEthernet interface counters are cleared.

```
Rack1R5#event manager run CLEAR_COUNTERS

Rack1R5#
%SYS-5-CONFIG_I: Configured from console by console
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : CTL : cli_open
called.
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : OUT : Rack1R5>
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : IN  : Rack1R5>enable
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : OUT : Rack1R5#
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : IN  : Rack1R5#clear
counters FastEthernet0/0
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : OUT : Clear "show
interface" counters on this interface [confirm]
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : IN  : y
%CLEAR-5-COUNTERS: Clear counter on interface FastEthernet0/0 by  on
vty0 (EEM:CLEAR_COUNTERS)
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : OUT : y
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : OUT : Rack1R5#
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : OUT : Rack1R5#
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : IN  : Rack1R5#enable
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : OUT : Rack1R5#
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : IN  : Rack1R5#clear
counters FastEthernet0/1
```

Copyright © 2010 Internetwork Expert

```
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : OUT : Clear "show
interface" counters on this interface [confirm]
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : IN  : y
%CLEAR-5-COUNTERS: Clear counter on interface FastEthernet0/1 by  on
vty0 (EEM:CLEAR_COUNTERS)
Rack1R5#
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : OUT : y
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : OUT : Rack1R5#
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : OUT : Rack1R5#
%HA_EM-6-LOG: CLEAR_COUNTERS : DEBUG(cli_lib) : : CTL : cli_close
called.
```

After you have debugged your script by running it manually you may associate it back with the watchdog event detector and use the debugging commands demonstrated in the previous task.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

IP Services Solutions

> **Note**
>
> Revert all devices to the *IP Services* initial configurations prior to continuing.

13.1 Proxy ARP

- Disable IP routing on SW1, configure interface VLAN 146 with the IP address 155.X.146.7/24, and shutdown all other interfaces running IP.
- Disable Proxy-ARP on R4's connection to VLAN 146.
- Test reachability from SW1 to the rest of the devices in the routing domain, and check the ARP cache of SW1.
- Configure R5's interface FastEthernet0/0 in VLAN 146 with the IP address 155.X.146.5/24, and disable Proxy-ARP on the link.
- Configure the layer 2 switchports on SW1 connecting to R1 and R5 as protected ports, and enable Local Proxy ARP on R6's connection to VLAN 146.
- Test reachability from R1 to R5 on VLAN 146 and check the ARP cache of both of these devices.
- Revert to the original configuration on R5 and SW1.

Copyright © 2010 Internetwork Expert

Configuration

```
R4:
interface FastEthernet0/1
 no ip proxy-arp

R5:
interface FastEthernet0/0
 ip address 155.1.146.5 255.255.255.0
 no ip proxy-arp

R6:
interface FastEthernet0/0.146
 ip proxy-arp
 ip local-proxy-arp

SW1:
interface Vlan 146
 ip address 155.1.146.7 255.255.255.0
!
interface Vlan 67
 shutdown
!
interface Vlan 79
 shutdown
!
interface FastEthernet0/3
 shutdown
!
no ip routing
!
interface FastEthernet 0/5
 switchport access vlan 146
 switchport mode access
 switchport protected
!
interface FastEthernet 0/1
 switchport protected
```

Copyright © 2010 Internetwork Expert

Verification

```
Rack5SW1#sh ip route
Default gateway is not set

Host               Gateway           Last Use    Total Uses  Interface
ICMP redirect cache is empty
```

> **Note**
>
> With IP routing disabled and no default-gateway configured on SW1 it will ARP for all destinations, including non-local destinations.

```
Rack5SW1#clear arp-cache
Rack5SW1#ping 155.1.146.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 58/260/1065 ms

Rack5SW1#ping 155.1.0.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 58/258/1057 ms

Rack5SW1#show ip arp 155.1.146.6
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  155.1.146.6            1   0011.5cf9.97c0  ARPA   Vlan146

Rack5SW1#show ip arp 155.1.0.2
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  155.1.58.5             1   0011.5cf9.97c0  ARPA   Vlan146
```

The MAC addresses in SW1's ARP cache for R2 and R6 are the same, since R6 performs Proxy-ARP for all of SW1's requests.

Next, protected ports would normally prohibit R5 from communicating with R1. However, with Local Proxy-ARP enabled on R6 it will reply to R1 and R5 with its own MAC address for them to reach each other.

Copyright © 2010 Internetwork Expert

```
Rack1R1#ping 155.1.146.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.5, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 1/3/4 ms

Rack1R1#ping 155.1.146.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1R1#show ip arp 155.1.146.5
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  155.1.146.5              1  0011.5cf9.97c0  ARPA   FastEthernet0/0

Rack1R1#show ip arp 155.1.146.6
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  155.1.146.6              1  0011.5cf9.97c0  ARPA   FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

13.2 DHCP Server

- Configure R6 as a DHCP server for hosts on VLAN 146, but to ignore BOOTP packets.
- Store the DHCP binding in the flash memory of R6.
- Clients should use R4 and R6 as the redundant default gateways.
- DHCP clients should use the default-router IP addresses for DNS servers.
- Exclude the address range 155.X.146.100 - 155.X.146.254 from allocation.
- Lease the addresses for 12 hours.

Configuration

```
R6:
ip dhcp pool VLAN146
 network 155.1.146.0 /24
 default-router 155.1.146.6 155.1.146.4
 dns-server 155.1.146.6 155.1.146.4
 lease 0 12
!
no ip bootp server
!
ip dhcp excluded-address 155.1.146.100 155.1.146.254
!
ip dhcp database flash:/bindings
```

Verification

```
Rack1R6#show ip dhcp pool

Pool VLAN146 :
 Utilization mark (high/low)    : 100 / 0
 Subnet size (first/next)       : 0 / 0
 Total addresses                : 254
 Leased addresses               : 0
 Pending event                  : none
 1 subnet is currently in the pool :
 Current index        IP address range                    Leased
addresses
 155.1.146.1          155.1.146.1      - 155.1.146.254     0
```

Copyright © 2010 Internetwork Expert

Note

Storing the DHCP database in flash allows R6 to retain host allocations when the router reboots.

```
Rack1R6#show ip dhcp database
URL      : flash:/bindings
Read     : Never
Written  : Mar 01 2002 01:26 AM
Status   : Last write succeeded. Agent information is up-to-date.
Delay    : 300 seconds
Timeout  : 300 seconds
Failures : 0
Successes: 1
```

Pitfall

Configure the excluded IP address ranges before configuring the DHCP pool itself. If the pool is configure first clients may request and be allocated IP addresses faster than the exclusion can be applied.

Copyright © 2010 Internetwork Expert

13.3 DHCP Client

- Configure R1 to obtain its IP address for the link to VLAN 146 dynamically via DHCP.
- The DHCP client-id should be based off of the MAC address of the interface connecting to VLAN 146.

Configuration

```
R1:
interface FastEthernet 0/0
 ip address dhcp client-id FastEthernet 0/0
```

Verification

> **Note**
>
> Enable DHCP packet debugging on the client before requesting the IP address via DHCP.

```
Rack1R1#debug dhcp detail

RAC: Starting DHCP discover on FastEthernet0/0
DHCP: Try 1 to acquire address for FastEthernet0/0
DHCP: allocate request
DHCP: zapping entry in DHC_PURGING state for Fa0/0
DHCP: new entry. add to queue
```

R1 sends a DHCP DISCOVER on the interface with a source IP address of 0.0.0.0. Note that the client-ID contains the MAC address of the FastEthernet interface.

```
DHCP: SDiscover attempt # 1 for entry:
Temp IP addr: 0.0.0.0  for peer on Interface: FastEthernet0/0
Temp  sub net mask: 0.0.0.0
   DHCP Lease server: 0.0.0.0, state: 1 Selecting
   DHCP transaction id: 6AAFAD
   Lease: 0 secs,  Renewal: 0 secs,  Rebind: 0 secs
   Next timer fires after: 00:00:03
   Retry count: 1   Client-ID: 0007.8500.27c0
   Hostname: Rack1R1
DHCP: SDiscover: sending 279 byte length DHCP packet
DHCP: SDiscover 279 bytes
            B'cast on FastEthernet0/0 interface from 0.0.0.0
```

Copyright © 2010 Internetwork Expert

R1 receives the OFFER message from R6, as seen in the Server ID field. Additional DHCP options, such as DNS, are also seen here inside the offer.

```
DHCP: Received a BOOTREP pkt
DHCP: Scan: Message type: DHCP Offer
DHCP: Scan: Server ID Option: 155.1.146.6 = 9B059206
DHCP: Scan: Lease Time: 43200
DHCP: Scan: Renewal time: 21600
DHCP: Scan: Rebind time: 37800
DHCP: Scan: Subnet Address Option: 255.255.255.0
DHCP: Scan: Router Option: 155.1.146.6, 155.1.146.4
DHCP: Scan: DNS Name Server Option: 155.1.146.6, 155.1.146.4
DHCP: rcvd pkt source: 155.1.146.6,  destination:  255.255.255.255
   UDP  sport: 43,  dport: 44,  length: 308
   DHCP op: 2, htype: 1, hlen: 6, hops: 0
   DHCP server identifier: 155.1.146.6
        xid: 6AAFAD, secs: 0, flags: 8000
        client: 0.0.0.0, your: 155.1.146.3
        srvr:   0.0.0.0, gw: 0.0.0.0
        options block length: 60

DHCP Offer Message   Offered Address: 155.1.146.3
DHCP: Lease Seconds: 43200    Renewal secs:  21600    Rebind secs:
37800
DHCP: Server ID Option: 155.1.146.6
DHCP: offer received from 155.1.146.6
```

R1 sends a DHCP REQUEST for the IP address it was offered.

```
DHCP: SRequest attempt # 1 for entry:
Temp IP addr: 155.1.146.3  for peer on Interface: FastEthernet0/0
Temp  sub net mask: 255.255.255.0
   DHCP Lease server: 155.1.146.6, state: 2 Requesting
   DHCP transaction id: 6AAFAD
   Lease: 43200 secs,  Renewal: 0 secs,  Rebind: 0 secs
   Next timer fires after: 00:00:01
   Retry count: 1   Client-ID: 0007.8500.27c0
   Hostname: Rack1R1
DHCP: SRequest- Server ID option: 155.1.146.6
DHCP: SRequest- Requested IP addr option: 155.1.146.3
DHCP: SRequest placed lease len option: 43200
DHCP: SRequest: 297 bytes
DHCP: SRequest: 297 bytes
           B'cast on FastEthernet0/0 interface from 0.0.0.0
```

Copyright © 2010 Internetwork Expert

R1 receives a DHCP ACK message from the server, and the allocation is complete.

```
DHCP: Received a BOOTREP pkt
DHCP: Scan: Message type: DHCP Ack
DHCP: Scan: Server ID Option: 155.1.146.6 = 9B059206
DHCP: Scan: Lease Time: 43200
DHCP: Scan: Renewal time: 21600
DHCP: Scan: Rebind time: 37800
DHCP: Scan: Host Name: Rack1R1
DHCP: Scan: Subnet Address Option: 255.255.255.0
DHCP: Scan: Router Option: 155.1.146.6, 155.1.146.4
DHCP: Scan: DNS Name Server Option: 155.1.146.6, 155.1.146.4
DHCP: rcvd pkt source: 155.1.146.6,  destination:  255.255.255.255
   UDP  sport: 43,  dport: 44,  length: 311
   DHCP op: 2, htype: 1, hlen: 6, hops: 0
   DHCP server identifier: 155.1.146.6
        xid: 6AAFAD, secs: 0, flags: 8000
        client: 0.0.0.0, your: 155.1.146.3
        srvr:   0.0.0.0, gw: 0.0.0.0
        options block length: 63
```

The binding can be verified on the DHCP server as seen below.

```
Rack1R6#sh ip dhcp binding
Bindings from all pools not associated with VRF:
IP address          Client-ID/            Lease expiration        Type
                 Hardware address/
                 User name
155.1.146.3         0100.0785.0027.c0       Mar 01 2002 01:53 PM
Automatic
```

Finally, verify that the agent database stores the bindings in the file on flash.

```
Rack1R6#more flash:bindings
*time* Mar 01 2002 01:58 AM
*version* 2
!IP address     Type  Hardware address   Lease expiration      VRF
155.1.146.3     id    0100.0785.0027.c0  Mar 01 2002 01:53 PM

!IP address      Type     Hardware address  Interface-index


!IP address     Interface-index  Lease expiration      Server IP
address  Hardware address  Vrf
*end*
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.4 DHCP Relay

- Configure R6 to allocate IP addresses via DHCP to clients on VLAN 58.
- The clients on VLAN 58 should use R5 as the default gateway.
- Configure SW2's VLAN 58 interface to obtain an IP address via DHCP.
- Base the client-id on the MAC address of its VLAN 58 interface.
- Ensure to avoid IP address conflicts with the existing hosts on VLAN 58.

Configuration

```
R6:
ip dhcp pool VLAN58
 network 155.1.58.0 /24
 default-router 155.1.58.5
!
ip dhcp excluded-address 155.1.58.5

R5:
interface FastEthernet 0/0
 ip helper-address 155.1.146.6

SW2:
interface Vlan 58
 ip address dhcp client-id Vlan79
```

Verification

Note

DHCP relay acts as a proxy agent, forwarding all broadcast packets from the clients to the unicast IP address of the actual DHCP server. The server uses a special "giaddr" field in the DHCP packets which contains the IP address of the DHCP relay interface. This field allows the server to select the proper DHCP pool and then unicast a reply back to the DHCP relay agent.

To verify the configuration enable DHCP debugging on the server and request a new IP address on SW2.

Copyright © 2010 Internetwork Expert

```
Rack1R6#debug ip dhcp server packet
Rack1R6#debug ip dhcp server events
```

R6 receives a DHCP DISCOVER packet from the relay IP address (R5) with the Client-ID is based on the MAC address of SW2's VLAN 58 interface.

```
DHCPD: DHCPDISCOVER received from client 0100.1bd4.d47a.42 through
relay 155.1.58.5.
DHCPD: Seeing if there is an internally specified pool class:
  DHCPD: htype 1 chaddr 001b.d4d4.7a42
  DHCPD: remote id 020a00009b01920600000092
  DHCPD: circuit id 00000000
DHCPD: Allocate an address without class information (155.1.58.0)
DHCPD: Adding binding to radix tree (155.1.58.9)
DHCPD: Adding binding to hash tree
```

R6 sends a DHCP OFFER packet to the IP address of the relay (R5).

```
DHCPD: assigned IP address 155.1.58.9 to client 0100.1bd4.d47a.42.
DHCPD: Sending DHCPOFFER to client 0100.1bd4.d47a.42 (155.1.58.9).
DHCPD: unicasting BOOTREPLY for client 001b.d4d4.7a42 to relay
155.1.58.5.
```

SW2 requests the IP address using a REQUEST packet.

```
DHCPD: DHCPREQUEST received from client 0100.1bd4.d47a.42.
DHCPD: Sending notification of ASSIGNMENT:
 DHCPD: address 155.1.58.9 mask 255.255.255.0
  DHCPD: htype 1 chaddr 001b.d4d4.7a42
  DHCPD: lease time remaining (secs) = 86400
DHCPD: No default domain to append - abort update
```

The server acknowledges the assignment by sending ACK to the relay.

```
DHCPD: Sending DHCPACK to client 0100.1bd4.d47a.42 (155.1.58.9).
DHCPD: unicasting BOOTREPLY for client 001b.d4d4.7a42 to relay
155.1.58.5.
```

Allocation can be verified by checking the DHCP binding database on R6.

```
Rack1R6#show ip dhcp binding
Bindings from all pools not associated with VRF:
IP address          Client-ID/              Lease expiration        Type
                  Hardware address/
                  User name
155.1.58.9          0100.1bd4.d47a.42       Mon DD YYYY 10:21 PM
Automatic
```

Copyright © 2010 Internetwork Expert

13.5 DHCP Host Pools

- Create a DHCP host pool on R6 corresponding to R1's VLAN 146 interface to lease R1 the IP address 155.X.146.1.

Configuration

> **Note**
>
> To find the DHCP Client-ID corresponding to R1 view the binding database on R6. You can easily distinguish a Client-ID from the Hardware Address from additional byte value "01" in the beginning of the string. You may need to release the IP address of R1 before adding a static host pool in R6.

```
Rack1R6#show ip dhcp binding
Bindings from all pools not associated with VRF:
IP address          Client-ID/              Lease expiration        Type
                    Hardware address/
                    User name
155.1.58.9          0100.1bd4.d47a.42       Jul 14 2008 10:21 PM
Automatic
155.1.146.1         0100.0c85.96a4.a0       Jul 13 2008 10:35 PM
Automatic

R6:
ip dhcp pool R1_HOST
 client-id 0100.0c85.96a4.a0
 host 155.1.146.1 /24
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Issue **no ip address dhcp** command on R1's interface followed by **ip address dhcp client-id** <Interface>. Ensure the **debug dhcp detail** has been enabled on R1 prior to that.

```
Rack1R1#debug dhcp detail

DHCP: DHCP client process started: 10
RAC: Starting DHCP discover on FastEthernet0/0
DHCP: Try 1 to acquire address for FastEthernet0/0
DHCP: allocate request
DHCP: new entry. add to queue
DHCP: SDiscover attempt # 1 for entry:
Temp IP addr: 0.0.0.0  for peer on Interface: FastEthernet0/0
Temp  sub net mask: 0.0.0.0
   DHCP Lease server: 0.0.0.0, state: 1 Selecting
   DHCP transaction id: 62C016
   Lease: 0 secs,  Renewal: 0 secs,  Rebind: 0 secs
   Next timer fires after: 00:00:03
   Retry count: 1   Client-ID: 000c.8596.a4a0
   Hostname: Rack1R1
DHCP: SDiscover: sending 279 byte length DHCP packet
DHCP: SDiscover 279 bytes
            B'cast on FastEthernet0/0 interface from 0.0.0.0
```

In addtion to allocating a static IP address to R1 per the host pool, the DHCP server (R6) also imports the other settings (DNS servers, default routers, etc.) from the more specific subnet-wide DHCP pool.

```
DHCP: Received a BOOTREP pkt
DHCP: Scan: Message type: DHCP Offer
DHCP: Scan: Server ID Option: 155.1.146.6 = 9B019206
DHCP: Scan: Lease Time: 86400
DHCP: Scan: Renewal time: 43200
DHCP: Scan: Rebind time: 75600
DHCP: Scan: Subnet Address Option: 255.255.255.0
DHCP: Scan: Router Option: 155.1.146.6, 155.1.146.4
DHCP: Scan: DNS Name Server Option: 155.1.146.6, 155.1.146.4
DHCP: rcvd pkt source: 155.1.146.6,  destination:  255.255.255.255
   UDP  sport: 43,  dport: 44,  length: 308
   DHCP op: 2, htype: 1, hlen: 6, hops: 0
   DHCP server identifier: 155.1.146.6
        xid: 62C016, secs: 0, flags: 8000
        client: 0.0.0.0, your: 155.1.146.1
        srvr:   0.0.0.0, gw: 0.0.0.0
        options block length: 60
```

Copyright © 2010 Internetwork Expert

```
DHCP Offer Message   Offered Address: 155.1.146.1
DHCP: Lease Seconds: 86400    Renewal secs:  43200    Rebind secs:
75600
DHCP: Server ID Option: 155.1.146.6
DHCP: offer received from 155.1.146.6
```

R1 requests the IP address offered from R6.

```
DHCP: SRequest attempt # 1 for entry:
Temp IP addr: 155.1.146.1  for peer on Interface: FastEthernet0/0
Temp  sub net mask: 255.255.255.0
   DHCP Lease server: 155.1.146.6, state: 2 Requesting
   DHCP transaction id: 62C016
   Lease: 86400 secs,  Renewal: 0 secs,  Rebind: 0 secs
   Next timer fires after: 00:00:01
   Retry count: 1   Client-ID: 000c.8596.a4a0
   Hostname: Rack1R1
DHCP: SRequest- Server ID option: 155.1.146.6
DHCP: SRequest- Requested IP addr option: 155.1.146.1
DHCP: SRequest placed lease len option: 86400
DHCP: SRequest: 297 bytes
DHCP: SRequest: 297 bytes
           B'cast on FastEthernet0/0 interface from 0.0.0.0
```

The new IP address has been acknowledged by R6.

```
DHCP: Received a BOOTREP pkt
DHCP: Scan: Message type: DHCP Ack
DHCP: Scan: Server ID Option: 155.1.146.6 = 9B019206
DHCP: Scan: Lease Time: 86400
DHCP: Scan: Renewal time: 43200
DHCP: Scan: Rebind time: 75600
DHCP: Scan: Subnet Address Option: 255.255.255.0
DHCP: Scan: Router Option: 155.1.146.6, 155.1.146.4
DHCP: Scan: DNS Name Server Option: 155.1.146.6, 155.1.146.4
DHCP: rcvd pkt source: 155.1.146.6,  destination:
255.255.255.2Interface FastEthernet0/0 assigned DHCP address
155.1.146.1, mask 255.255.255.0
55
   UDP  sport: 43,  dport: 44,  length: 308
   DHCP op: 2, htype: 1, hlen: 6, hops: 0
   DHCP server identifier: 155.1.146.6
        xid: 62C016, secs: 0, flags: 8000
        client: 0.0.0.0, your: 155.1.146.1
        srvr:   0.0.0.0, gw: 0.0.0.0
        options block length: 60
```

Copyright © 2010 Internetwork Expert

```
DHCP Ack Message
DHCP: Lease Seconds: 86400    Renewal secs:  43200    Rebind secs:
75600
DHCP: Server ID Option: 155.1.146.6-if)#
Rack1R1(config-if)#
DHCP Client Pooling: ***Allocated IP address: 155.1.146.1
Allocated IP address = 155.1.146.1  255.255.255.0
```

When you're done with R1, verify the binding database on R6.

```
Rack1R6#show ip dhcp binding
Bindings from all pools not associated with VRF:
IP address          Client-ID/              Lease expiration        Type
                    Hardware address/
                    User name
155.1.58.9          0100.1bd4.d47a.42       Jul 14 2008 10:21 PM
Automatic
155.1.146.1         0100.0c85.96a4.a0       Infinite
Manual
```

Copyright © 2010 Internetwork Expert

13.6 DHCP on-Demand Pool

- Configure the link between R1 and R3 to use PPP.
- Configure R3 to assign the IP address 155.X.13.1/24 to R1 using IPCP.
- Create a DHCP pool on R1 that imports the subnet, the netmask, and the DNS servers (R4 and R6) from R3 via IPCP.
- Ensure that RIP routing is still functional once this configuration is complete.

Configuration

```
R1:
interface Serial 0/1
 encapsulation ppp
 ip address negotiated
 ppp ipcp mask request
 ppp ipcp dns request
 no peer neighbor-route
!
ip dhcp pool ODAP_POOL
 import all
 origin ipcp
!
router rip
 no validate-update-source

R3:
interface Serial 1/2
 encapsulation ppp
 ip address 155.1.13.3 255.255.255.0
 peer default ip address 155.1.13.1
 ppp ipcp mask 255.255.255.0
 ppp ipcp dns 155.1.146.4 155.1.146.6
 no peer neighbor-route
```

Copyright © 2010 Internetwork Expert

Verification

Note

ODAP allows importing DHCP settings using various methods, one of them being IPCP (Internet Protocol Control Protocol for PPP). IPCP allows importing an IP address along with the subnet mask (while PPP ignores the mask, DHCP can use it) as well as DNS/WINS servers. This is useful in PPPoE deployments, such as with DSL, where the DSL CPE router gets an address assignment through DHCP from the provider, and forwards options such as DNS to the clients behind the CPE.

To verify this check the DHCP pool settings and verify the imported parameters.

```
Rack1R1#show ip dhcp import

Address Pool Name: ODAP_POOL
Domain Name Server(s): 155.1.146.4 155.1.146.6

Rack1R1#show ip dhcp pool

Pool ODAP_POOL :
 Utilization mark (high/low)    : 100 / 0
 Subnet size (first/next)       : 0 / 0
 Total addresses                : 254
 Leased addresses               : 0
 Pending event                  : none
 1 subnet is currently in the pool :
 Current index        IP address range                    Leased
addresses
 155.1.13.1           155.1.13.1       - 155.1.13.254      0
```

Pitfall

When R1 receives RIP updates from R3 they are discarded because they are on different networks. This is due to the fact that R1 uses the host route 155.X.13.1/32, and R3 is outside this range. To resolve this R1 must disable the update source validation feature. Additionally the peer-neighbor-route feature is disabled so that R1 does not inject R3's IP address as a host route into the routing domain.

Copyright © 2010 Internetwork Expert

13.7 DHCP Proxy

- Configure R2 to request an IP address on its serial connection to R3 via IPCP.
- R3 should relay this request via DHCP to R6, and R6 should allocate the IP address 155.X.23.2 to R2.

Configuration

```
R1:
ip route 155.1.23.0 255.255.255.0 155.1.13.3

R2:
interface Serial 0/1
 encapsulation ppp
 ip address negotiated
 no peer neighbor-route

R3:
interface Serial 1/3
 encapsulation ppp
 ip address 155.1.23.3 255.255.255.0
 peer default ip address dhcp
 no peer neighbor-route
!
ip address-pool dhcp-proxy-client
ip dhcp-server 155.1.146.6

R6:
ip dhcp pool R2_SERIAL
 network 155.1.23.0 /24
!
ip dhcp excluded-address 155.1.23.1
ip dhcp excluded-address 155.1.23.3 155.1.23.254
!
ip route 155.1.23.0 255.255.255.0 155.1.146.1
```

Copyright © 2010 Internetwork Expert

Verification

Pitfall

The static route is needed in this exercise because R3's link to R2 isn't in the UP/UP state until IP is actually negotiated. This means that when the proxy request is received at the server, R6 does not have a route back to the relay in order to send the reply back.

Note

DHCP Proxy is an address allocation solution for PPP. When a client requests an IP address via IPCP, the server may issue a DHCP request (act as a proxy) using its own IP address on the PPP interface as the "giaddr" field.

To verify the task, enable debugging of DHCP events in R3 and R6.

```
Rack1R3#debug dhcp detail
Rack1R6#debug ip dhcp server packet
Rack1R6#debug ip dhcp server event
```

Shutdown and no-shutdown R3's connection to R2 and observe the debugging output. Ensure R3 released the IP address before no-shutting the interface.

```
Rack1R3#
```

R3 sends a DHCP discover packet to the globally configured DHCP server as a proxy IPCP address request.

Copyright © 2010 Internetwork Expert

```
DHCP: proxy allocate request
DHCP: zapping entry in DHC_PURGING state for Se1/3
DHCP: deleting entry 64F20E80 155.1.23.2 from list
Temp IP addr: 155.1.23.2  for peer on Interface: Serial1/3
Temp  sub net mask: 255.255.255.0
   DHCP Lease server: 155.1.146.6, state: 9 Purging
   DHCP transaction id: 152B
   Lease: 86400 secs,  Renewal: 43200 secs,  Rebind: 75600 secs
   Next timer fires after: 00:00:21
   Retry count: 0   Client-ID: cisco-155.1.23.3-Serial1/3
   Client-ID hex dump: 636973636F2D3135352E312E32332E33
                       2D53657269616C312F33
DHCP: new entry. add to queue, interface Serial1/3
DHCP: SDiscover attempt # 1 for entry:
Rack1R3(config-if)#
Temp IP addr: 0.0.0.0  for peer on Interface: Serial1/3
Temp  sub net mask: 0.0.0.0
   DHCP Lease server: 0.0.0.0, state: 1 Selecting
   DHCP transaction id: 152C
   Lease: 0 secs,  Renewal: 0 secs,  Rebind: 0 secs
   Next timer fires after: 00:00:04
   Retry count: 1   Client-ID: cisco-155.1.23.3-Serial1/3
   Client-ID hex dump: 636973636F2D3135352E312E32332E33
                       2D53657269616C312F33
   Hostname: Rack1R3
DHCP: SDiscover: sending 292 byte length DHCP packet
DHCP: SDiscover 292 bytes
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/3, changed
state to up
```

Copyright © 2010 Internetwork Expert

R3 receives an OFFER from the DHCP server with the client IP address.

```
DHCP: XID MATCH in dhcpc_for_us()
DHCP: Received a BOOTREP pkt
DHCP: Scan: Message type: DHCP Offer
DHCP: Scan: Server ID Option: 155.1.146.6 = 9B019206
DHCP: Scan: Lease Time: 86400
DHCP: Scan: Renewal time: 43200
DHCP: Scan: Rebind time: 75600
DHCP: Scan: Subnet Address Option: 255.255.255.0
DHCP: rcvd pkt source: 155.1.146.6,  destination:  155.1.23.3
   UDP  sport: 43,  dport: 43
Rack1R3(config,  length: 308
   DHCP op: 2, htype: 1, hlen: 6, hops: 0
   DHCP server identifier: 155.1.146.6
        xid: 152C, secs: 0, flags: 0
        client: 0.0.0.0, your: 155.1.23.2
        srvr:   0.0.0.0, gw: 155.1.23.3
        options block length: 60

DHCP Offer Message   Offered Address: 155.1.23.2
DHCP: Lease Seconds: 86400    Renewal secs:  43200    Rebind secs:
75600
DHCP: Server ID Option: 155.1.146.6
DHCP: offer received from 155.1.146.6
DHCP: SRequest attempt # 1 for entry:
Temp IP addr: 155.1.23.2  for peer on Interface: Serial1/3
Temp  sub net mask: 255.255.255.0
   DHCP Lease server: 155.1.146.6, state: 2 Requesting
   DHCP transaction id: 152C
   Lease: 86400 secs,  Renewal: 0 secs,  Rebind: 0 secs
   Next timer fires after: 00:00:03
   Retry count: 1   Client-ID: cisco-155.1.23.3-Serial1/3
   Client-ID hex dump: 636973636F2D3135352E312E32332E33
                       2D53657269616C312F33
   Hostname: Rack1R3
```

Copyright © 2010 Internetwork Expert

A request is sent for the IP address offered by the DHCP server.

```
DHCP: SRequest- Server ID option: 155.1.146.6
DHCP: SRequest- Requested IP addr option: 155.1.23.2
DHCP: SRequest placed lease len option: 86400
DHCP: SRequest: 310 bytes
DHCP: SRequest: 310 bytes
DHCP: XID MATCH in dhcpc_for_us()
DHCP: Received a BOOTREP pkt
DHCP: Scan: Message type: DHCP Ack
DHCP: Scan: Server ID Option: 155.1.146.6 = 9B019206
DHCP: Scan: Lease Time: 86400
DHCP: Scan: Renewal time: 43200
DHCP: Scan: Rebind time: 75600
DHCP: Scan: Subnet Address Option: 255.255.255.0
DHCP: rcvd pkt source: 155.1.146.6,  destination:  155.1.23.3
   UDP  sport: 43,  dport: 43,  length: 308
   DHCP op: 2, htype: 1, hlen: 6, hops: 0
   DHCP server identifier: 155.1.146.6
        xid: 152C, secs: 0, flags: 0
        client: 0.0.0.0, your: 155.1.23.2
        srvr:   0.0.0.0, gw: 155.1.23.3
        options block length: 60
```

The server ACKed the request and the IP address is now assigned to R2.

```
DHCP Ack Message
DHCP: Lease Seconds: 86400    Renewal secs:  43200    Rebind secs:
75600
DHCP: Server ID Option: 155.1.146.6
DHCP Proxy Client Pooling: ***Allocated IP address: 155.1.23.2
```

The debugging output at R6 is as follows. The server receives a DHCP DISCOVER message from the relay, R3.

```
Rack1R6#
DHCPD: DHCPDISCOVER received from client
0063.6973.636f.2d31.3535.2e31.2e32.332e.332d.5365.7269.616c.312f.33
through relay 155.1.23.3.
DHCPD: Seeing if there is an internally specified pool class:
  DHCPD: htype 1 chaddr 0007.eb49.04c1
  DHCPD: remote id 020a00009b01920600000092
  DHCPD: circuit id 00000000
DHCPD: Allocate an address without class information (155.1.23.0)
DHCPD: Adding binding to radix tree (155.1.23.2)
DHCPD: Adding binding to hash tree
DHCPD: assigned IP address 155.1.23.2 to client
0063.6973.636f.2d31.3535.2e31.2e32.332e.332d.5365.7269.616c.312f.33.
```

Copyright © 2010 Internetwork Expert

The server sends a DHCP OFFER message to the relay.

```
DHCPD: Sending DHCPOFFER to client
0063.6973.636f.2d31.3535.2e31.2e32.332e.332d.5365.7269.616c.312f.33
(155.1.23.2).
DHCPD: unicasting BOOTREPLY for client 0007.eb49.04c1 to relay
155.1.23.3.
```

The client requests the IP address and the server acknowledges the request.

```
DHCPD: DHCPREQUEST received from client
0063.6973.636f.2d31.3535.2e31.2e32.332e.332d.5365.7269.616c.312f.33.
DHCPD: Sending notification of ASSIGNMENT:
 DHCPD: address 155.1.23.2 mask 255.255.255.0
  DHCPD: htype 1 chaddr 0007.eb49.04c1
  DHCPD: lease time remaining (secs) = 86400
DHCPD: No default domain to append - abort update
DHCPD: Sending DHCPACK to client
0063.6973.636f.2d31.3535.2e31.2e32.332e.332d.5365.7269.616c.312f.33
(155.1.23.2).
DHCPD: unicasting BOOTREPLY for client 0007.eb49.04c1 to relay
155.1.23.3.
DHCPD: checking for expired leases.
DHCPD: checking for expired leases.
DHCPD: writing bindings to flash:/bindings.

Rack1R6#show ip dhcp binding
Bindings from all pools not associated with VRF:
IP address          Client-ID/              Lease expiration        Type
                    Hardware address/
                    User name
155.1.23.2          0063.6973.636f.2d31.    Jul 09 2008 10:05 PM
Automatic
                    3535.2e31.2e32.332e.
                    332d.5365.7269.616c.
                    312f.33
```

Copyright © 2010 Internetwork Expert

13.8 DHCP Information Option

- Configure R5 to insert the DHCP information option field into DHCP requests forwarded to R6 from clients on VLAN 58.
- R5 should set the subscriber-id string for these requests to “VLAN58”.
- Configure R6 to allocate IP addresses based on Option 82 and assign a static IP address to SW2 based on the information option contents.

Configuration

```
R5:
ip dhcp relay information option
!
interface FastEthernet0/0
 ip dhcp relay information option subscriber-id VLAN58

R6:
ip dhcp class TEST
   relay agent information
      relay-information hex
020c020a00009b013a0500000000606564c414e3538
!
ip dhcp pool VLAN58
   class TEST
     address range 155.1.58.8 155.1.58.8
```

Copyright © 2010 Internetwork Expert

Verification

Note

The DHCP information option format may vary from vendor to vendor and even from platform to platform. Therefore, IOS matches a variable hex string against an information option (you can use wildcard "*" and bitmask to match Option 82 ranges, e.g. remote-id with multiple subscriber-ids) to define a DHCP class.

DHCP debugging commands will not reveal the information option passed in relay packets. To find out the actual value use the detailed IP packets dumping procedere as follows.

First, create an access-list that matches only DHCP packets.

```
Rack1R6#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#access-list 100 permit udp any any eq bootps
```

Then enable DHCP packets debugging using an access-list 100 and hidden **dump** option, and configure SW2 to release/request an IP address via DHCP.

```
Rack1R6#debug ip packet detail 100 dump
IP packet debugging is on (detailed) (dump) for access list 100

Rack1SW2#conf t
Rack1SW2(config)#interface Vlan58
Rack1SW2(config-if)#shutdown
Rack1SW2(config-if)#no shutdown
```

Look at the packet dumps below. The two captured packets are DHCP DISCOVER and DHCP REQUEST packets sent from SW2. Note that the information option (you can locate it by looking for ASCII subscriber-id sring "VLAN58") is located at the very end of UDP packet.

The information option starts with a byte having decimal value 82 (hex value 0x52, information option number), followed by the total option length (e.g. 0x16 = 22 bytes). After that, suboptions follow (e.g. remote-id, circuit-id). The suboptions are always TLVs, i.e. type-length-value blocks, with type and length being one-byte quantities. Cisco may add sub-TLVs into each suboption to distinguish remote-id type etc.

Copyright © 2010 Internetwork Expert

Look at the byte-string extracted from the captured DHCP packet.

|5216|020C|020A|00009B013A0500000000|0606|564C414E35|

This is DHCP option 82 (0x52) of total length 22 (0x16). The first suboption is remote-id (0x02 – suboption type) of total length 12 (0xC). The remote-id suboption starts with yet another TLV (type 0x02 length 0x0A) defining remote-id type, followed by 10 bytes of remote-id value. The next suboption is type 0x06 and length 0x06, which represents ASCII value assigned to R5 VLAN58 interface.

However, in a DHCP class, you need to match either a complete Option 82 or its part, using either a wildcard "*" or bitmask. In our case, the whole content of Option 82 (starting after initial 0x52 0x16) is matched by the class statement.

Copyright © 2010 Internetwork Expert

```
Rack1R6#
<snip>
07C57820: 63825363 35010236 049B0192 06330400  c.Sc5..6. ...3..
07C57830: 0151803A 040000A8 C03B0400 01275001  .Q.:...(@;...'P.
07C57840: 04FFFFFF 0003049B 013A0552 16020C02  . .......:.R....
07C57850: 0A00009B 013A0500 00000006 06564C41  .....:. .....VLA
07C57860: 4E3538FF                             N58.

IP: tableid=0, s=155.1.58.5 (FastEthernet0/0.146), d=155.1.146.6
(FastEthernet0/0.146), routed via RIB
IP: s=155.1.58.5 (FastEthernet0/0.146), d=155.1.146.6
(FastEthernet0/0.146), len 350, rcvd 3
    UDP src=67, dst=67

07E28FE0:                      000F 24093CE0           ..$.<`
07E28FF0: 000D2846 8F210800 4500015E 002E0000  ..(F.!..E..^....
07E29000: FD11BA52 9B013A05 9B019206 00430043  }.:R..:. ....C.C
07E29010: 014A1983 01010601 000010F3 00008000  .J. .......s....
07E29020: 00000000 00000000 00000000 9B013A05  . ............:.
07E29030: 001BD4D4 7A420000 00000000 00000000  ..TTzB. .......
07E29040: 00000000 00000000 00000000 00000000  . ..............
07E29050: 00000000 00000000 00000000 00000000  . ..............
07E29060: 00000000 00000000 00000000 00000000  . ..............
07E29070: 00000000 00000000 00000000 00000000  . ..............
07E29080: 00000000 00000000 00000000 00000000  . ..............
07E29090: 00000000 00000000 00000000 00000000  . ..............
07E290A0: 00000000 00000000 00000000 00000000  . ..............
07E290B0: 00000000 00000000 00000000 00000000  . ..............
07E290C0: 00000000 00000000 00000000 00000000  . ..............
07E290D0: 00000000 00000000 00000000 00000000  . ..............
07E290E0: 00000000 00000000 00000000 00000000  . ..............
07E290F0: 00000000 00000000 00000000 00000000  . ..............
07E29100: 63825363 35010339 0204803D 0701001B  c.Sc5..9...=....
07E29110: D4D47A42 36049B01 92063204 9B013A08  TTzB6. ...2...:.
07E29120: 33040001 51800C08 5261636B 31535732  3...Q...Rack1SW2
07E29130: 37080106 0F2C0321 962B3401 03521602  7....,.!.+4..R..
07E29140: 0C020A00 009B013A 05000000 00060656  . .....:.......V
07E29150: 4C414E35 38FF                        LAN58.

IP: tableid=0, s=155.1.146.6 (local), d=155.1.58.5
(FastEthernet0/0.146), routed via FIB
IP: s=155.1.146.6 (local), d=155.1.58.5 (FastEthernet0/0.146), len 342,
sending
    UDP src=67, dst=67
```

Copyright © 2010 Internetwork Expert

```
07E28970:                   45000156 001B0000          E..V....
07E28980: FF11B86D 9B019206 9B013A05 00430043  ..8m. ..:..C.C
07E28990: 0142546B 02010600 000010F3 00008000  .BTk. ...s....
07E289A0: 00000000 9B013A08 00000000 9B013A05  ......:. ...:.
07E289B0: 001BD4D4 7A420000 00000000 00000000  ..TTzB. .....
07E289C0: 00000000 00000000 00000000 00000000  . ...........
07E289D0: 00000000 00000000 00000000 00000000  . ...........
07E289E0: 00000000 00000000 00000000 00000000  . ...........
07E289F0: 00000000 00000000 00000000 00000000  . ...........
07E28A00: 00000000 00000000 00000000 00000000  . ...........
07E28A10: 00000000 00000000 00000000 00000000  . ...........
07E28A20: 00000000 00000000 00000000 00000000  . ...........
07E28A30: 00000000 00000000 00000000 00000000  . ...........
07E28A40: 00000000 00000000 00000000 00000000  . ...........
07E28A50: 00000000 00000000 00000000 00000000  . ...........
07E28A60: 00000000 00000000 00000000 00000000  . ...........
07E28A70: 00000000 00000000 00000000 00000000  . ...........
07E28A80: 63825363 35010536 049B0192 06330400  c.Sc5..6. .3..
07E28A90: 0151803A 040000A8 C03B0400 0127500C  .Q.:...(@;...'P.
07E28AA0: 08526163 6B315357 320104FF FFFF0003  .Rack1SW2. ..
07E28AB0: 049B013A 05521602 0C020A00 009B013A  ...:.R. .....:
07E28AC0: 05000000 00060656 4C414E35 38FF      . .....VLAN58.
```

Observe how R6 allocates a DHCP address based on the class. Enable DHCP server debugging and let SW2 renew its IP address.

```
Rack1R6#debug ip dhcp server packet
Rack1R6#debug ip dhcp server event
Rack1R6#debug ip dhcp server class

Rack1SW2#conf t
Rack1SW2(config)#interface Vlan58
Rack1SW2(config-if)#shutdown
Rack1SW2(config-if)#no shutdown

Rack1R6#
DHCPD: DHCPRELEASE message received from client 0100.1bd4.d47a.42
(155.1.58.8).
DHCPD: Sending notification of TERMINATION:
 DHCPD: address 155.1.58.8 mask 255.255.255.0
 DHCPD: reason flags: RELEASE
  DHCPD: htype 1 chaddr 001b.d4d4.7a42
  DHCPD: lease time remaining (secs) = 84898
DHCPD: returned 155.1.58.8 to address pool VLAN58.
DHCPD: checking for expired leases.
DHCPD: DHCPRELEASE message received from client 0100.1bd4.d47a.42
(155.1.58.8).
DHCPD: Finding a relay for client 0100.1bd4.d47a.42 on interface
FastEthernet0/0.146.
```

Copyright © 2010 Internetwork Expert

Note that remote-id and circuit-id shown in DHCP debugging output do not represent the information received from R5.

```
DHCPD: Seeing if there is an internally specified pool class:
  DHCPD: htype 1 chaddr 001b.d4d4.7a42
  DHCPD: remote id 020a00009b01920600000092
  DHCPD: circuit id 00000000
DHCPD: Sending notification of DISCOVER:
  DHCPD: htype 1 chaddr 001b.d4d4.7a42
  DHCPD: remote id 020a00009b01920600000092
  DHCPD: circuit id 00000000
DHCPD: DHCPDISCOVER received from client 0100.1bd4.d47a.42 through
relay 155.1.58.5.
DHCPD: Seeing if there is an internally specified pool class:
  DHCPD: htype 1 chaddr 001b.d4d4.7a42
  DHCPD: remote id 020a00009b01920600000092
  DHCPD: circuit id 00000000
```

The configured DHCP class matched the information option received, and therefore the configured address subrange is used for allocation.

```
DHCPD: Class 'TEST' matched by default
DHCPD: Searching for a match to 'relay-information
020c020a00009b013a05000000000606564c414e3538' in class TEST
DHCPD: input pattern 'relay-information
020c020a00009b013a05000000000606564c414e3538' matches class TEST
DHCPD: input matches class TEST
DHCPD: Adding binding to radix tree (155.1.58.8)
DHCPD: Adding binding to hash tree
DHCPD: assigned IP address 155.1.58.8 to client 0100.1bd4.d47a.42.
DHCPD: Sending DHCPOFFER to client 0100.1bd4.d47a.42 (155.1.58.8).
DHCPD: unicasting BOOTREPLY for client 001b.d4d4.7a42 to relay
155.1.58.5.
DHCPD: DHCPREQUEST received from client 0100.1bd4.d47a.42.
DHCPD: Sending notification of ASSIGNMENT:
 DHCPD: address 155.1.58.8 mask 255.255.255.0
  DHCPD: htype 1 chaddr 001b.d4d4.7a42
  DHCPD: lease time remaining (secs) = 86400
DHCPD: No default domain to append - abort update
DHCPD: Sending DHCPACK to client 0100.1bd4.d47a.42 (155.1.58.8).
DHCPD: unicasting BOOTREPLY for client 001b.d4d4.7a42 to relay
155.1.58.5.
```

Copyright © 2010 Internetwork Expert

13.9 DHCP Authorized ARP

- Configure R6 to update the ARP cache on its VLAN 146 interface with the information from the DHCP binding database.
- Ensure that R6 does not accept any ARP mappings on its VLAN 146 interface other than those learned from the DHCP binding database.

Configuration

```
R6:
ip dhcp pool VLAN146
 update arp
!
ip dhcp pool R1_HOST
 update arp
!
interface FastEthernet 0/0.146
 arp authorize

R6:
!
! R4's MAC address is statically mapped
!
arp 155.1.146.4 0007.0e9c.dac2 arpa
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> The Update ARP feature allows the DHCP process to populate the ARP cache with the DHCP-based entries. Authorized ARP enabled at interface level will disable any dynamic ARP learning on that interface. Therefore a static ARP entry for R4 is needed to reach the router (which has static IP address).

```
Rack1R6#ping 155.1.146.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms

Rack1R6#ping 155.1.146.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.1, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 1/1/4 ms
```

When the static ARP entry for R4 is removed it can't be reached anymore since ARP learning is disabled.

```
Rack1R6(config)#no arp 155.1.146.4 0007.0e9c.dac2 arpa
Rack1R6(config)#do ping 155.1.146.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)
```

Copyright © 2010 Internetwork Expert

> **Note**
>
> Revert all devices to the *IP Services* initial configurations prior to continuing.

13.10 IP SLA

- Configure R6 to automatically monitor BB1's availability using ping and telnet every one minute.

Configuration

```
R6:
ip sla 1
 icmp-echo 54.1.1.254
 timeout 1000
 frequency 1
!
ip sla schedule 1 life forever start-time now
!
ip sla 2
 Tcp-connect 54.1.1.254 23 control disable
 timeout 5000
!
ip sla schedule 2 life forever start-time now
```

Copyright © 2010 Internetwork Expert

Verification

```
Rack1R6#show ip sla statistics 1
Round trip time (RTT)     Index 1
      Latest RTT: 24 ms
Latest operation start time: 00:00:51.921 UTC Wed Jul 9 2008
Latest operation return code: OK
Number of successes: 117
Number of failures: 0
Operation time to live: Forever

Rack1R6#show ip sla statistics 2
Round trip time (RTT)     Index 2
      Latest RTT: 28 ms
Latest operation start time: 00:00:04.011 UTC Wed Jul 9 2008
Latest operation return code: OK
Number of successes: 2
Number of failures: 0
Operation time to live: Forever
```

Copyright © 2010 Internetwork Expert

13.11 Object Tracking

- Create two enhanced objects on R6 to track the state of previously created IP SLA operations.
- Create a third enhanced object on R6 that tracks the first two objects; this third object should only be "up" if both of the first two objects are "up".
- The first two objects should wait 10 seconds before reporting their state from up to down, or from down to up.

Configuration

```
R6:
track 1 rtr 1
 delay down 10 up 10
!
track 2 rtr 2
 delay down 10 up 10
!
track 3 list boolean and
 object 1
 object 2
```

Verification

> **Note**
>
> In this example a Boolean AND operation is used to calculate the resulting state of the two member objects. Since AND only returns TRUE if all inputs are TRUE, the state of object 3 is only "up" if both objects 1 and 2 are "up".
>
> In addition to SLA operations, interface states or IP routes can also be tracked. For IP routes it is possible to track routing metrics and see when they cross thresholds.
>
> To verify this configuration display state of the tracked objects as follows.

```
Rack1R6#show track
Track 1
  Response Time Reporter 1 state
  State is Up
    1 change, last change 00:04:08
  Delay up 10 secs, down 10 secs
  Latest operation return code: OK
  Latest RTT (millisecs) 24
  Tracked by:
    Track-list 3
```

Copyright © 2010 Internetwork Expert

```
Track 2
  Response Time Reporter 2 state
  State is Up
    1 change, last change 00:02:54
  Delay up 10 secs, down 10 secs
  Latest operation return code: OK
  Latest RTT (millisecs) 28
  Tracked by:
    Track-list 3
Track 3
  List boolean and
  Boolean AND is Up
    2 changes, last change 00:02:27
    object 1 Up
    object 2 Up
```

Stop ICMP packets coming back from BB1 and see how that effects tracked objects.

```
Rack1R6#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#access-list 100 deny icmp any any
Rack1R6(config)#access-list 100 permit ip any any
Rack1R6(config)#interface Serial 0/0/0
Rack1R6(config-if)#ip access-group 100 in

Rack1R6#show track
Track 1
  Response Time Reporter 1 state
  State is Down
    2 changes, last change 00:06:19
  Delay up 10 secs, down 10 secs
  Latest operation return code: Timeout
  Tracked by:
    Track-list 3
Track 2
  Response Time Reporter 2 state
  State is Up
    1 change, last change 00:11:35
  Delay up 10 secs, down 10 secs
  Latest operation return code: OK
  Latest RTT (millisecs) 100
  Tracked by:
    Track-list 3
Track 3
  List boolean and
  Boolean AND is Down
    3 changes, last change 00:06:18
    object 1 Down
    object 2 Up
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.12 HSRP

- Configure HSRP on R4 and R6 so that hosts on VLAN 146 can use the virtual IP address 155.X.146.254 as their default gateway.
- Ensure that R6 is the active physical gateway unless it loses connectivity to VLAN 146.
- Set the hello interval to 1 second and dead interval to 3 seconds for faster failed gateway detection.
- Authenticate HSRP message exchanges using an MD5 hash of the key CISCO.

Configuration

```
R6:
!
! Using "preempt" is recommended to ensure deterministic
! primary gateway election
!
interface FastEthernet 0/0.146
 standby 146 ip 155.1.146.254
 standby 146 timers 1 3
 standby 146 preempt
 standby 146 authentication md5 key-string CISCO
 standby 146 name VLAN146
 standby 146 priority 110

R4:
!
! Using "preempt" is recommended to ensure deterministic
! primary gateway election
!
interface FastEthernet 0/1
 standby 146 ip 155.1.146.254
 standby 146 timers 1 3
 standby 146 preempt
 standby 146 authentication md5 key-string CISCO
 standby 146 name VLAN146
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Issue the **show standby** command to ensure that R6 is the active HSRP gateway, and to verify the backup gateway activation. Note the last octet of the virtual HSRP MAC address which represents the group number (shown in hex).

```
Rack1R6#show standby
FastEthernet0/0.146 - Group 146
  State is Active
    2 state changes, last state change 00:10:37
  Virtual IP address is 155.1.146.254
  Active virtual MAC address is 0000.0c07.ac92
    Local virtual MAC address is 0000.0c07.ac92 (v1 default)
  Hello time 1 sec, hold time 3 sec
    Next hello sent in 0.539 secs
  Authentication MD5, key-string "CISCO"
  Preemption enabled
  Active router is local
  Standby router is 155.1.146.4, priority 100 (expires in 2.058 sec)
  Priority 110 (configured 110)
  IP redundancy name is "VLAN146" (cfgd)

Rack1R6(config)#interface fastEthernet 0/0.146
Rack1R6(config-subif)#shutdown

Rack1R4#show standby
Ethernet0/1 - Group 146
  State is Active
    2 state changes, last state change 00:00:35
  Virtual IP address is 155.1.146.254
  Active virtual MAC address is 0000.0c07.ac92
    Local virtual MAC address is 0000.0c07.ac92 (v1 default)
  Hello time 1 sec, hold time 3 sec
    Next hello sent in 0.216 secs
  Authentication MD5, key-string "CISCO"
  Preemption enabled
  Active router is local
  Standby router is unknown
  Priority 100 (default 100)
  IP redundancy name is "VLAN146" (cfgd)
```

Copyright © 2010 Internetwork Expert

13.13 VRRP

- Configure VRRP on R4 and R6 so that hosts on VLAN 146 can use the virtual IP address 155.X.146.253 as their default gateway.
- Ensure that R4 is the active physical gateway unless it loses connectivity to VLAN 146.
- Set the hello interval to 3 seconds, and authenticate messages using the clear-text password CISCO.

Configuration

R6:

```
interface FastEthernet 0/0.146
 vrrp 146 ip 155.1.146.253
 vrrp 146 timers advertise 3
 vrrp 146 authentication CISCO
```

R4:

```
interface FastEthernet 0/1
 vrrp 146 ip 155.1.146.253
 vrrp 146 timers advertise 3
 vrrp 146 authentication CISCO
 vrrp 146 priority 110
```

Verification

> **Note**
>
> Verify the status of the virtual router and check the backup activation with the **`show vrrp`** command.

```
Rack1R6#show vrrp
FastEthernet0/0.146 - Group 146
  State is Backup
  Virtual IP address is 155.1.146.253
  Virtual MAC address is 0000.5e00.0192
  Advertisement interval is 3.000 sec
  Preemption enabled
  Priority is 100
  Authentication text "CISCO"
  Master Router is 155.1.146.4, priority is 110
  Master Advertisement interval is 3.000 sec
  Master Down interval is 9.609 sec (expires in 7.709 sec)
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface FastEthernet 0/1
Rack1R4(config-if)#shutdown

Rack1R6#show vrrp
FastEthernet0/0.146 - Group 146
  State is Master
  Virtual IP address is 155.1.146.253
  Virtual MAC address is 0000.5e00.0192
  Advertisement interval is 3.000 sec
  Preemption enabled
  Priority is 100
  Authentication text "CISCO"
  Master Router is 155.1.146.6 (local), priority is 100
  Master Advertisement interval is 3.000 sec
  Master Down interval is 9.609 sec
```

Copyright © 2010 Internetwork Expert

13.14 GLBP

- Configure GLBP for hosts on VLAN 146 so that first-hop redundancy and load balancing occur between R4 and R6.
- Ensure R6 is elected as the Active Virtual Gateway (AVG), and both R4 and R6 are used as Active Virtual Forwarders (AVFs) for the virtual IP address 155.X.146.252.
- Use a 1 second hello interval and a 3 second hold time.
- Distribute the traffic load between R4 and R6 in 2:1 ratio.
- Authenticate GLBP UDP packet exchange using a secure method based on a password value of CISCO.

Configuration

```
R6:
interface FastEthernet 0/0.146
 glbp 146 ip 155.1.146.252
 glbp 146 timers 1 3
 glbp 146 priority 110
 glbp 146 preempt
 glbp 146 weighting 10
 glbp 146 name VLAN146
 glbp 146 authentication md5 key-string CISCO
 glbp 146 load-balancing weighted

R4:
interface FastEthernet 0/1
 glbp 146 ip 155.1.146.252
 glbp 146 timers 1 3
 glbp 146 preempt
 glbp 146 weighting 20
 glbp 146 name VLAN146
 glbp 146 authentication md5 key-string CISCO
 glbp 146 load-balancing weighted
```

Copyright © 2010 Internetwork Expert

Verification

Note

GLBP allows multiple active physical gateways serve the same virtual gateway IP address. The AVG (active virtual gateway) responds to ARP requests sent to to virtual gateway IP address, and replies with different virtual MAC addresses that correspond to multiple active virtual forwarders (AVFs). Both the AVG and AVFs are redundant, i.e. if a primary physical router representing the AVG or an AVF fails, another physical router will take its role.

To verify this first check status of the GLBP group and virtual forwarders state with the **`show glbp`** command.

```
Rack1R4#show glbp
FastEthernet0/1 - Group 146
```

R4 is the STANDBY AVG, since R6 has a higher priority.

```
State is Standby
    4 state changes, last state change 00:02:31
  Virtual IP address is 155.1.146.252
  Hello time 1 sec, hold time 3 sec
    Next hello sent in 0.864 secs
  Redirect time 600 sec, forwarder time-out 14400 sec
  Authentication MD5, key-string "CISCO"
  Preemption enabled, min delay 0 sec
  Active is 155.1.146.6, priority 110 (expires in 2.268 sec)
  Standby is local
  Priority 100 (default)
  Weighting 20 (configured 20), thresholds: lower 1, upper 20
  Load balancing: weighted
  IP redundancy name is "VLAN146"
  Group members:
```

There are two members in the GLBP group, and R6 (the other member) is authenticated

```
    0007.0e9c.dac2 (155.1.146.4) local
    0012.019a.27c0 (155.1.146.6) authenticated
  There are 2 forwarders (1 active)
```

Copyright © 2010 Internetwork Expert

It is important to understand that each AVF is in fact a virtual gateway consisting of primary and backup physical gateways. From the output below you can see that R4 is the primary physical gateway for Forwarder 1 and is in "Listen" (backup state) for Forwarder 2. However, for each AVF only the primary physical gateway would forwards packets, the other is listening to become active for that AVF if the primary fails. The load between AVFs is distributed based on the amount of times the AVG will respond to ARP requests for the IP address of the virtual gateway with the virtual MAC of a particular AVF. The load is proportional to the weight assigned to an AVF if the load-balancing scheme is "weighted".

```
  Forwarder 1
    State is Active
      1 state change, last state change 00:25:34
    MAC address is 0007.b400.9201 (default)
    Owner ID is 0007.0e9c.dac2
    Preemption enabled, min delay 30 sec
    Active is local, weighting 20
    Arp replies sent: 12
  Forwarder 2
    State is Listen
    MAC address is 0007.b400.9202 (learnt)
    Owner ID is 0012.019a.27c0
    Time to live: 14399.896 sec (maximum 14400 sec)
    Preemption enabled, min delay 30 sec
    Active is 155.1.146.6 (primary), weighting 10 (expires in 2.892
sec)

Rack1R6#show glbp
State is Active
    1 state change, last state change 00:03:29
  Virtual IP address is 155.1.146.252
  Hello time 1 sec, hold time 3 sec
    Next hello sent in 0.956 secs
  Redirect time 600 sec, forwarder time-out 14400 sec
  Authentication MD5, key-string "CISCO"
  Preemption enabled, min delay 0 sec
  Active is local
  Standby is 155.1.146.4, priority 100 (expires in 2.395 sec)
  Priority 110 (configured)
  Weighting 10 (configured 10), thresholds: lower 1, upper 10
  Load balancing: weighted
  IP redundancy name is "VLAN146"
  Group members:
    0007.0e9c.dac2 (155.1.146.4) authenticated
    0012.019a.27c0 (155.1.146.6) local
  There are 2 forwarders (0 active)
  Forwarder 1
    State is Listen
    MAC address is 0007.b400.9201 (learnt)
    Owner ID is 0007.0e9c.dac2
    Redirection enabled, 599.335 sec remaining (maximum 600 sec)
    Time to live: 14399.335 sec (maximum 14400 sec)
    Preemption enabled, min delay 30 sec
```

Copyright © 2010 Internetwork Expert

```
    Active is 155.1.146.4 (primary), weighting 20 (expires in 2.335
sec)
  Forwarder 2
    State is Active
      1 state change, last state change 00:00:00
    MAC address is 0007.b400.9202 (default)
    Owner ID is 0012.019a.27c0
    Redirection enabled
    Preemption enabled, min delay 30 sec
    Active is local, weighting 10
```

The below output illustrates how the AVG responds with virtual MACs based on a client's (R1) ARP requests to the GLBP virtual IP address. Note that virtual MAC 0007.b400.9201 is AVF 1 (R4), and 0007.b400.9202 is AVF 2 (R6).

```
Rack1R1#ping 155.1.146.252

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.252, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 1/2/4 ms

Rack1R1#show ip arp 155.1.146.252
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  155.1.146.252           0   0007.b400.9201  ARPA   FastEthernet0/0

Rack1R1#clear arp-cache
Rack1R1#ping 155.1.146.252

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.252, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 1/2/4 ms

Rack1R1#sh ip arp 155.1.146.252
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  155.1.146.252           0   0007.b400.9202  ARPA   FastEthernet0/0

Rack1R1#clear arp-cache
Rack1R1#ping 155.1.146.252

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.252, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms

Rack1R1#sh ip arp 155.1.146.252
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  155.1.146.252           0   0007.b400.9201  ARPA   FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#clear arp-cache
Rack1R1#ping 155.1.146.252

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.252, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 1/7/24 ms

Rack1R1#sh ip arp 155.1.146.252
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  155.1.146.252          0   0007.b400.9201  ARPA   FastEthernet0/0
```

Note that the AVF 1 (R4) virtual MAC appears more often than the AVF 2 MAC address, following the configured 2:1 distribution ratio.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.15 Router Redundancy and Object Tracking

- R6 should be the active HSRP gateway as long as it can ping BB1 and reach it through telnet, but give up the active role once it loses either telnet or ping reachability to BB1.
- Ensure R4 is the active VRRP gateway when it sees the prefix 30.0.0.0/16 received via IGP from BB3.

Configuration

R6:

```
interface FastEthernet 0/0.146
 standby 146 track 3 decrement 20
```

R4:

```
track 1 ip route 30.0.0.0/16 reachability
!
interface FastEthernet 0/1
 standby 146 preempt
 vrrp 146 track 1 decrement 20
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Originally HSRP was only able to track an interface state (e.g. uplink interface) and decrease its priority when the interface failed. As illustrated in the *IP Routing* section of this series, the interface state of a link is not always a good indication of end-to-end connectivity out the link. Combining enhanced object tracking with first hop redundancy extends functionality to any application that can be tracked through IP SLA in addition to basic routing checks. Remember to enable HSRP preemption on the "secondary" router to allow the secondary overtake role of the primary router.
>
> The below output verifies how HSRP tracks the object on R6.

```
Rack1R6#show track 3
Track 3
  List boolean and
  Boolean AND is Up
    2 changes, last change 00:02:15
    object 1 Up
    object 2 Up
  Tracked by:
    HSRP FastEthernet0/0.146 146

Rack1R6#show standby
FastEthernet0/0.146 - Group 146
  State is Active
    1 state change, last state change 00:03:36
  Virtual IP address is 155.1.146.254
  Active virtual MAC address is 0000.0c07.ac92
    Local virtual MAC address is 0000.0c07.ac92 (v1 default)
  Hello time 1 sec, hold time 3 sec
    Next hello sent in 0.940 secs
  Authentication MD5, key-string "CISCO"
  Preemption enabled
  Active router is local
  Standby router is 155.1.146.4, priority 100 (expires in 2.395 sec)
  Priority 110 (configured 110)
    Track object 3 state Up decrement 20
  IP redundancy name is "VLAN146" (cfgd)
```

Copyright © 2010 Internetwork Expert

Shutting the link to BB1 down causes the enhanced objects to go down. This in turn causes R6 to decrement its HSRP priority and stop forwarding packets for that HSRP group.

```
Rack1R6#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface Serial 0/0/0
Rack1R6(config-if)#shutdown

HSRP: Fa0/0.146 Grp 146 Track 3 object changed, state Up -> Down
HSRP: Fa0/0.146 Grp 146 Priority 110 -> 90
HSRP: Fa0/0.146 Grp 146 Ignoring Coup (100/155.1.146.4 <
110/155.1.146.6)
HSRP: Fa0/0.146 Grp 146 Hello  in  155.1.146.4 Active  pri 100 vIP
155.1.146.254
HSRP: Fa0/0.146 Grp 146 Active router is 155.1.146.4, was local
HSRP: Fa0/0.146 Grp 146 Standby router is unknown, was 155.1.146.4
HSRP: Fa0/0.146 Grp 146 Active: g/Hello rcvd from higher pri Active
router (100/155.1.146.4)
HSRP: Fa0/0.146 Grp 146 Active -> Speak
%HSRP-5-STATECHANGE: FastEthernet0/0.146 Grp 146 state Active -> Speak
HSRP: Fa0/0.146 Grp 146 Redundancy "VLAN146" state Active -> Speak
HSRP: Fa0/0.146 Grp 146 Speak: d/Standby timer expired (unknown)
HSRP: Fa0/0.146 Grp 146 Standby router is local
HSRP: Fa0/0.146 Grp 146 Speak -> Standby
%HSRP-5-STATECHANGE: FastEthernet0/0.146 Grp 146 state Speak -> Standby
HSRP: Fa0/0.146 Grp 146 Redundancy "VLAN146" state Speak -> Standby

Rack1R6#show standby
FastEthernet0/0.146 - Group 146
  State is Standby
    3 state changes, last state change 00:02:25
  Virtual IP address is 155.1.146.254
  Active virtual MAC address is 0000.0c07.ac92
    Local virtual MAC address is 0000.0c07.ac92 (v1 default)
  Hello time 1 sec, hold time 3 sec
    Next hello sent in 0.595 secs
  Authentication MD5, key-string "CISCO"
  Preemption enabled
  Active router is 155.1.146.4, priority 100 (expires in 2.334 sec)
  Standby router is local
  Priority 90 (configured 110)
    Track object 3 state Down decrement 20
  IP redundancy name is "VLAN146" (cfgd)
```

Copyright © 2010 Internetwork Expert

The verification procedure is repeated with VRRP on R4 through the local tracking object.

```
Rack1R4#show track 1
Track 1
  IP route 30.0.0.0 255.255.0.0 reachability
  Reachability is Up (RIP)
    2 changes, last change 00:00:09
  First-hop interface is Ethernet0/0
  Tracked by:
    VRRP FastEthernet0/1 146

Rack1R4#show vrrp
FastEthernet0/1 - Group 146
  State is Master
  Virtual IP address is 155.1.146.253
  Virtual MAC address is 0000.5e00.0192
  Advertisement interval is 3.000 sec
  Preemption enabled
  Priority is 110
    Track object 1 state Up decrement 20
  Authentication text "CISCO"
  Master Router is 155.1.146.4 (local), priority is 110
  Master Advertisement interval is 3.000 sec
  Master Down interval is 9.570 sec

Rack1R4#conf t
Rack1R4(config)#inteface FastEthernet 0/0
Rack1R4(config)#shutdown

Rack1R4(config)#do show vrrp
Ethernet0/1 - Group 146
  State is Backup
  Virtual IP address is 155.1.146.253
  Virtual MAC address is 0000.5e00.0192
  Advertisement interval is 3.000 sec
  Preemption enabled
  Priority is 90  (cfgd 110)
    Track object 1 state Down decrement 20
  Authentication text "CISCO"
  Master Router is 155.1.146.6, priority is 100
  Master Advertisement interval is 3.000 sec
  Master Down interval is 9.570 sec (expires in 8.714 sec)
```

Copyright © 2010 Internetwork Expert

13.16 IRDP

- R5 and SW2 should advertise themselves as default gateways for hosts on VLAN58 using ICMP messages.
- R5 should be the preferred gateway on the segment.

Configuration

```
R5:
interface FastEthernet 0/0
 ip irdp
 ip irdp address 155.1.58.5 1000
 ip irdp maxadvertinterval 20
 ip irdp minadvertinterval 10

SW2:
interface Vlan 58
 ip irdp
 ip irdp address 155.1.58.8 500
 ip irdp maxadvertinterval 20
 ip irdp minadvertinterval 10
```

Copyright © 2010 Internetwork Expert

Verification

Note

IRDP uses ICMP messages to advertise candidate default gateway on a segment. Multiple routers can advertise themselves at the same time, with different priorities.

To verify this configuration disable IP routing on SW1, and enable default gateway discovery via IRDP. Additonally create a VLAN 58 interface on SW1 and assign it an IP address of 155.X.58.7/24.

```
SW1:
no ip routing
ip gdp irdp
!
interface Vlan 58
 ip address 155.1.58.7 255.255.255.0
```

```
Rack1SW1#show ip route
Gateway         Using  Interval  Priority   Interface
155.1.58.5      IRDP       30      1000     Vlan58

Default gateway is not set

Host               Gateway           Last Use    Total Uses  Interface
ICMP redirect cache is empty
```

```
Rack1R5#show ip irdp fastethernet 0/0
FastEthernet0/0 has router discovery enabled

Advertisements will occur between every 10 and 20 seconds.
Advertisements are sent with multicasts.
Advertisements are valid for 60 seconds.
Default preference will be 0.
Proxy for 155.1.58.5 with preference 1000.
```

Copyright © 2010 Internetwork Expert

> **Note**
>
> Revert all devices to the *IP Services* initial configurations prior to continuing.

13.17 Router ICMP Settings

- Configure R1's VLAN146 interface to stop sending ICMP messages about discarded packets, ICMP messages to select a better next-hop, and ICMP messages reporting subnets mask.
- Rate-limit ICMP unreachables to 2 per-second globally.

Configuration

```
R1:
interface FastEthernet 0/0
 no ip redirects
 no ip unreachables
 no ip mask-reply
!
ip icmp rate-limit unreachable 500
```

Verification

> **Note**
>
> You can verify the rate limit of unreachable using the **`traceroute`** command on R3. The destination router sends ICMP unreachable packets in response to UDP probes. Note that you can't do traceroute from R6 or R4 since R1 has ICMP unreachables disabled on VLAN146 interface.

```
Rack1R3#traceroute 155.1.13.1

Type escape sequence to abort.
Tracing the route to 155.1.13.1

  1 155.1.13.1 12 msec *  12 msec
```

Copyright © 2010 Internetwork Expert

Change the ICMP unreachables rate to 1 each every 10ms and repeat the traceroute command from R3.

```
Rack1R1(config)#ip icmp rate-limit unreachable 10

Rack1R3#traceroute 155.1.13.1

Type escape sequence to abort.
Tracing the route to 155.1.13.1

  1 155.1.13.1 16 msec 16 msec 16 msec
```

Verify the ICMP messages settings on R1 VLAN146 interface.

```
Rack1R1#show ip interface fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  Internet address is 155.1.146.1/24
  Broadcast address is 255.255.255.255
  Address determined by DHCP
  MTU is 1500 bytes
  Helper address is not set
  Directed broadcast forwarding is disabled
  Multicast reserved groups joined: 224.0.0.9
  Outgoing access list is not set
  Inbound  access list is not set
  Proxy ARP is enabled
  Local Proxy ARP is disabled
  Security level is default
  Split horizon is enabled
  ICMP redirects are never sent
  ICMP unreachables are never sent
  ICMP mask replies are never sent
 <snip>
```

Copyright © 2010 Internetwork Expert

13.18 Basic NAT

- Ensure that R4 and R6 are not advertising any internal networks (i.e. 155.X.0.0/16 or 150.X.0.0/16) to BB3.
- Configure R4 and R6 to translate IP source addresses for packets from the subnet 155.X.0.0/16 going to BB3 and BB1 respectively using a dynamic pool consisting of the subnets of their BB connections.
- Do not use any route-maps to accomplish this.

Configuration

```
R4:
ip nat pool VLAN43 204.12.1.1 204.12.1.253 prefix-length 24
!
ip access-list extended NAT_TRAFFIC
 permit ip 155.1.0.0 0.0.255.255 any
!
ip nat inside source list NAT_TRAFFIC pool VLAN43
!
interface FastEthernet 0/0
 ip nat outside
!
interface FastEthernet 0/1
 ip nat inside
!
interface Serial 0/0/0
 ip nat inside
!
interface Serial 0/1/0
 ip nat inside
!
router rip
 passive-interface FastEthernet 0/0

R6:
ip nat pool SERIAL 54.1.1.1 54.1.1.253 prefix-length 24
!
ip access-list extended NAT_TRAFFIC
 permit ip 155.1.0.0 0.0.255.255 any
!
ip nat inside source list NAT_TRAFFIC pool SERIAL
!
interface Serial 0/0/0
 ip nat outside
!
interface FastEthernet 0/0.146
 ip nat inside
!
interface FastEthernet 0/0.67
 ip nat inside
!
router rip
 passive-interface Serial 0/0/0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Source ICMP packets off the internal interfaces of R4 and R6 to verify that NAT translations are created. Send additional packets from R1 to ensure that the internal routers may also reach the networks behind the backbone routers.

```
Rack1R6#ping 212.18.1.1 source fastEthernet 0/0.146

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 212.18.1.1, timeout is 2 seconds:
Packet sent with a source address of 155.1.146.6
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/36/36 ms

Rack1R6#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
icmp 54.1.1.1:2        155.1.146.6:2      212.18.1.1:2       212.18.1.1:2
--- 54.1.1.1           155.1.146.6        ---                ---

Rack1R4#ping 30.0.0.1 source FastEthernet 0/1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:
Packet sent with a source address of 155.1.146.4
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 4/29/96 ms

Rack1R4#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
icmp 204.12.1.1:0      155.1.146.4:0      30.0.0.1:0         30.0.0.1:0
tcp 204.12.1.1:179     155.1.146.4:179    204.12.1.254:13219 204.12.1.254:13219
--- 204.12.1.1         155.1.146.4        ---

Rack1R1#ping 30.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:
..!!!
Success rate is 60 percent (3/5), round-trip min/avg/max = 4/5/8 ms

Rack1R1#ping 212.18.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 212.18.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/36/36 ms
```

Copyright © 2010 Internetwork Expert

Note that with access-list based NAT, the process creates one “parent” IP to IP address translation entry (e.g. 204.12.1.2 to 155.1.146.1 – inside global/inside local), which is used to multiplex all sessions for this particular inside host. Those IP to IP entries are called non-extendable, for they only have inside local and inside global IP address information. The one-to-one entries also permit outside hosts to connected to the inside using the temporary mapped IP address. Additionally, with non-extendable entries, an inside local address may have only one inside global translation. This effectively prevents multi-homed NAT when using access-lists only for NAT configuration.

```
Rack1R4#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
icmp 204.12.1.2:4546   155.1.146.1:4546   30.0.0.1:4546      30.0.0.1:4546
icmp 204.12.1.2:4547   155.1.146.1:4547   30.0.0.1:4547      30.0.0.1:4547
icmp 204.12.1.2:4548   155.1.146.1:4548   30.0.0.1:4548      30.0.0.1:4548
icmp 204.12.1.2:4549   155.1.146.1:4549   30.0.0.1:4549      30.0.0.1:4549
--- 204.12.1.2         155.1.146.1        ---                ---
tcp 204.12.1.1:179     155.1.146.4:179    204.12.1.254:13231 204.12.1.254:13231
--- 204.12.1.1         155.1.146.4        ---                ---

Rack1R6#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
icmp 54.1.1.2:692      155.1.146.1:692    212.18.1.1:692     212.18.1.1:692
icmp 54.1.1.2:693      155.1.146.1:693    212.18.1.1:693     212.18.1.1:693
icmp 54.1.1.2:694      155.1.146.1:694    212.18.1.1:694     212.18.1.1:694
icmp 54.1.1.2:695      155.1.146.1:695    212.18.1.1:695     212.18.1.1:695
icmp 54.1.1.2:696      155.1.146.1:696    212.18.1.1:696     212.18.1.1:696
--- 54.1.1.2           155.1.146.1        ---                ---
--- 54.1.1.1           155.1.146.6        ---                ---
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.19 NAT Overload

- Configure R4 and R6 to translate packets sourced from the Loopback 0 subnets (150.X.0.0/16) using their backbone connection IP addresses.

Configuration

```
R4:
ip access-list extended LOOPBACKS
 permit ip 150.1.0.0 0.0.255.255 any
!
ip nat inside source list LOOPBACKS interface FastEthernet 0/0 overload

R6:
ip access-list extended LOOPBACKS
 permit ip 150.1.0.0 0.0.255.255 any
!
ip nat inside source list LOOPBACKS interface Serial 0/0/0 overload
```

Copyright © 2010 Internetwork Expert

Verification

Note

NAT Overloading is also known as PAT (Port Address Translations) since it uses the same global IP address for all local IP addresses, changing just the port numbers.

To verify the configuration, source traffic off R1's Loopback 0 interface to the networks behind BB1 and BB3.

```
Rack1R1#ping 212.18.1.1 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 212.18.1.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/36/36 ms

Rack1R1#ping 30.0.0.1 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/6/8 ms
```

Note that "overload" NAT feature does not create any IP to IP (non-extendable) entries, but rather multiplexes all translations in one global IP address, using inside/outside IP addresses and port numbers (IDs) as keys.

```
Rack1R4#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
icmp 204.12.1.4:15     150.1.1.1:15       30.0.0.1:15        30.0.0.1:15

Rack1R6#show ip nat traslations
Pro Inside global      Inside local       Outside local      Outside global
icmp 54.1.1.6:14       150.1.1.1:14       212.18.1.1:14      212.18.1.1:14
```

Copyright © 2010 Internetwork Expert

13.20 NAT with Route Maps

- Configure R2 with route-map based NAT to support multiple outside interfaces.
- Traffic from R2's Loopback0 network going out the point-to-point link to R3 should be translated to 155.X.23.200.
- All other traffic going out the point-to-point link to R3 should be translated to R2's interface IP address.
- Traffic from R2's Loopback0 network going out the Frame Relay link to R5 should be translated to 150.X.2.200.
- All other traffic going out the Frame Relay link to R5 should be translated to R2's interface IP address.

Copyright © 2010 Internetwork Expert

Configuration

```
R2:
ip access-list standard FROM_LOOPBACK
 permit 150.1.2.0 0.0.0.255
!
route-map NAT_OUT_PPP_FROM_LOOPBACK permit 10
 match ip address FROM_LOOPBACK
 match interface Serial0/1
!
route-map NAT_OUT_PPP_NOT_FROM_LOOPBACK deny 10
 match ip address FROM_LOOPBACK
 match interface Serial0/1
!
route-map NAT_OUT_PPP_NOT_FROM_LOOPBACK permit 20
 match interface Serial0/1
!
route-map NAT_OUT_FRAME_RELAY_FROM_LOOPBACK permit 10
 match ip address FROM_LOOPBACK
 match interface Serial0/0
!
route-map NAT_OUT_FRAME_RELAY_NOT_FROM_LOOPBACK deny 10
 match ip address FROM_LOOPBACK
 match interface Serial0/0
!
route-map NAT_OUT_FRAME_RELAY_NOT_FROM_LOOPBACK permit 20
 match interface Serial0/0
!
ip nat pool PPP_LOOPBACK_POOL 155.1.23.200 155.1.23.200 prefix-length
24
!
ip nat inside source route-map NAT_OUT_PPP_FROM_LOOPBACK pool
PPP_LOOPBACK_POOL overload
!
ip nat inside source route-map NAT_OUT_PPP_NOT_FROM_LOOPBACK interface
Serial0/1 overload
!
ip nat pool FRAME_RELAY_LOOPBACK_POOL 150.1.2.200 150.1.2.200 prefix-
length 24
!
ip nat inside source route-map NAT_OUT_FRAME_RELAY_FROM_LOOPBACK pool
FRAME_RELAY_LOOPBACK_POOL overload
!
ip nat inside source route-map NAT_OUT_FRAME_RELAY_NOT_FROM_LOOPBACK
interface Serial0/0 overload
!
interface Serial0/0
 ip nat outside
!
interface Serial0/1
 ip nat outside
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> To verify this configuration telnet from R2 to destinations out the point-to-point and Frame Relay links while changing the source addresses.

```
Rack1R2#telnet 155.1.23.3 /source-interface FastEthernet0/0
Trying 155.1.23.3 ... Open

User Access Verification

Password:
Rack1R3>show users
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                 00:00:17
* 66 vty 0                 idle                 00:00:00 155.1.23.2

  Interface    User               Mode         Idle     Peer Address

Rack1R2#telnet 155.1.23.3 /source-interface Loopback0
Trying 155.1.23.3 ... Open

User Access Verification

Password:
Rack1R3>show users
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                 00:00:27
* 66 vty 0                 idle                 00:00:00 155.1.23.200

  Interface    User               Mode         Idle     Peer Address
```

Copyright © 2010 Internetwork Expert

```
Rack1R2#telnet 155.1.108.10 /source-interface FastEthernet0/0
Trying 155.1.108.10 ... Open

User Access Verification

Password:
Rack1SW4>show users
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                    3d21h
*  1 vty 0                 idle                 00:00:00 155.1.0.2

  Interface      User        Mode                      Idle     Peer
Address

Rack1R2#telnet 155.1.108.10 /source-interface Loopback0
Trying 155.1.108.10 ... Open


User Access Verification

Password:
Rack1SW4>show users
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                    3d21h
*  1 vty 0                 idle                 00:00:00 150.1.2.200

  Interface      User        Mode                      Idle     Peer
Address

Queued Packets: 0
```

When using route-maps for NAT configuration (not just simple access-lists) the NAT process creates “extendable” NAT entries (fully-extended, with local/global IP addresses and port numbers). This allows for NAT multi-homing, since a particular inside local source IP address is not bound to just one inside global IP address. Also, reverse connections to inside hosts using temporary global IP address mappings are not possible with default route-map configuration.

```
Rack1R2#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
tcp 150.1.2.200:37611  150.1.2.2:37611    155.1.108.10:23    155.1.108.10:23
udp 155.1.0.2:520      155.1.0.2:520      224.0.0.9:520      224.0.0.9:520
udp 155.1.23.2:520     155.1.23.2:520     224.0.0.9:520      224.0.0.9:520
tcp 155.1.0.2:35504    192.10.1.2:35504   155.1.108.10:23    155.1.108.10:23
```

Copyright © 2010 Internetwork Expert

13.21 Static NAT

- Configure R5 so that R4 can reach SW2's VLAN 58 IP address using the address 155.X.45.8.
- Additionally SW2 should be able reach R4's point-to-point link using the VLAN 58 IP address 155.X.58.4.

Configuration

```
R5:
interface Serial0/1/0
 ip nat outside
!
interface FastEthernet0/0
 ip nat inside
!
ip nat inside source static 155.1.58.8 155.1.45.8
ip nat outside source static 155.1.45.4 155.1.58.4
!
ip route 155.1.58.4 255.255.255.255 155.1.45.4
```

Verification

> **Note**
>
> When a packet arrives at the NAT "inside" interface its destination IP address is first used for a routing lookup before it is translated via the NAT table; this is why the static route is needed. When SW2 sends a packet to the address 155.X.58.4, R5 first attempts to route it before consulting the NAT table. Without the static route R5 ends up routing the packet back out the interface connecting to VLAN 58, which was the interface the packet came in on.

```
Rack1SW2#telnet 155.1.58.4
Trying 155.1.58.4 ... Open

User Access Verification

Password:
Rack1R4>sh users
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                 01:08:09
* 66 vty 0                 idle                 00:00:00 155.1.45.8

  Interface    User               Mode         Idle     Peer Address
```

```
Rack1R5#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
--- ---                ---                155.1.58.4         155.1.45.4
tcp 155.1.45.8:47235   155.1.58.8:47235   155.1.58.4:23      155.1.45.4:23
--- 155.1.45.8         155.1.58.8         ---                ---
```

To verify translation, route-cache is disabled on R5 to allow for transit packets to be process switched.

```
R5:
access-list 100 permit icmp any any
!
interface FastEthernet 0/0
 no ip route-cache
!
interface Serial 0/1/0
 no ip route-cache

Rack1R5#debug ip packet detail 100
IP packet debugging is on (detailed) for access list 100

Rack1R5#debug ip nat detail
IP NAT detailed debugging is on

Rack1SW2#ping 155.1.58.4 repeat 2

Type escape sequence to abort.
Sending 2, 100-byte ICMP Echos to 155.1.58.4, timeout is 2 seconds:
!!
Success rate is 100 percent (2/2), round-trip min/avg/max = 42/42/42 ms
```

The NAT inside translation is denoted by the lowercase “i”. The packet from SW2 reaches R5 and gets translated due to the static route out to R4.

```
Rack1R5#
IP: tableid=0, s=155.1.58.8 (FastEthernet0/0), d=155.1.58.4
(Serial0/1/0), routed via RIB
NAT: i: icmp (155.1.58.8, 3) -> (155.1.58.4, 3) [12]
NAT: s=155.1.58.8->155.1.45.8, d=155.1.58.4 [12]
NAT: s=155.1.45.8, d=155.1.58.4->155.1.45.4 [12]
IP: s=155.1.45.8 (FastEthernet0/0), d=155.1.45.4 (Serial0/1/0),
g=155.1.45.4, len 100, forward
    ICMP type=8, code=0
```

Copyright © 2010 Internetwork Expert

The NAT outside translation is denoted by the lowercase "o". For the "outside" interface the destination IP is first used for translation, and only then for the routing lookup. Based on this there is no need for a static "fixup" route on R5.

```
NAT*: o: icmp (155.1.45.4, 3) -> (155.1.45.8, 3) [12]
NAT*: s=155.1.45.4->155.1.58.4, d=155.1.45.8 [12]
NAT*: s=155.1.58.4, d=155.1.45.8->155.1.58.8 [12]
IP: tableid=0, s=155.1.58.4 (Serial0/1/0), d=155.1.58.8
(FastEthernet0/0), routed via FIB
IP: s=155.1.58.4 (Serial0/1/0), d=155.1.58.8 (FastEthernet0/0),
g=155.1.58.8, len 100, forward
    ICMP type=0, code=0

<snip>
```

Try removing the static route from R5 and see what happens.

```
Rack1R5(config)#no ip route 155.1.58.4 255.255.255.255 155.1.45.4

Rack1SW2#ping 155.1.58.4 repeat 2

Type escape sequence to abort.
Sending 2, 100-byte ICMP Echos to 155.1.58.4, timeout is 2 seconds:
!!
Success rate is 100 percent (2/2), round-trip min/avg/max = 8/8/9 ms

Rack1SW2#telnet 155.1.58.4
Trying 155.1.58.4 ...
% Connection refused by remote host
```

NAT never triggers at R5, but R5 responds to the ICMP echo packets due to the local alias. Note that you can disable the local aliasing (proxy ARP entries creation) for NAT entries using the **no-alias** option for the ip nat command.

```
Rack1R5#show arp | include 155.1.58.(4|5)
Internet  155.1.58.4              -   000c.ce60.9fe0  ARPA   FastEthernet0/0
Internet  155.1.58.5              -   000c.ce60.9fe0  ARPA   FastEthernet0/0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.22 Static PAT

- Remove the previous static NAT entries on R5.
- Configure R5 so that when R4 telnets to 155.X.45.44 at port 8023 they get connected to SW2's VLAN 8 interface at port 23.
- Configure R5 so that when R4 telnets to 155.X.45.44 at port 10023 they get connected to SW4's VLAN 10 interface at port 23.

Configuration

```
R5:
ip nat inside source static tcp 155.1.8.8 23 155.1.45.44 8023
ip nat inside source static tcp 155.1.10.10 23 155.1.45.44 10023
```

Verification

Note

Static PAT allows mapping ports on a global IP address to the ports on a local IP address. To verify this configuration, telnet from R4 to R5 at ports 8023 and 10023.

```
Rack1R4#telnet 155.1.45.44 8023
Trying 155.1.45.44, 8023 ... Open

User Access Verification

Password:
Rack1SW2>

Rack1R4#telnet 155.1.45.44 10023
Trying 155.1.45.44, 10023 ... Open

User Access Verification

Password:
Rack1SW4>
```

Copyright © 2010 Internetwork Expert

Note that the host responding to ICMP echo packets is R5, since it created the IP alias entry for the translated IP address.

```
Rack1R4#ping 155.1.45.44

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.45.44, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/29/32 ms

Rack1R5#show ip alias
Address Type              IP Address      Port
Interface                 150.1.5.5
Interface                 155.1.5.5
Dynamic                   155.1.45.44
Interface                 155.1.0.5
Interface                 155.1.58.5
Interface                 155.1.45.5
```

13.23 Static NAT and IP Aliasing

- Configure R5 so that it does not add a local alias for the static mapping of the IP address 155.X.45.44.

Configuration

```
R5:
ip nat inside source static tcp 155.1.8.8 23 155.1.45.44 8023
extendable no-alias
ip nat inside source static tcp 155.1.10.10 23 155.1.45.44 10023
extendable no-alias
```

Verification

> **Note**
>
> The no-alias feature will prevent a router running NAT from installing a local IP alias entry. However, the router will still respond to ARP requests for the global translated IP address, it just won't terminate any connection on itself. To verify this, telnet to 155.X.45.44 at port 8023 from R4, and ping this address.

```
Rack1R4#telnet 155.1.45.44 10023
Trying 155.1.45.44, 10023 ... Open

User Access Verification

Password:
Rack1SW4>
```

Now R5 does not respond to the ICMP echo packets since it has no local alias generated for the IP address 155.X.45.44.

```
Rack1R4#ping 155.1.45.44

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.45.44, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)

Rack1R5#show ip aliases
Address Type             IP Address      Port
Interface                150.1.5.5  Interface
155.1.5.5  Interface                155.1.0.5
Interface                155.1.58.5  Inter-
face                155.1.45.5
```

Copyright © 2010 Internetwork Expert

13.24 Static Policy NAT

- Remove all previous NAT configurations from R2.
- Configure a static translation on R3 so that when traffic is received from SW1's VLAN 7 IP address and it is going out the point-to-point link to R1 it is translated to the address 155.X.13.7.
- Configure a static translation on R3 so that when traffic is received from SW1's VLAN 7 IP address and it is going out the point-to-point link to R2 it is translated to the address 155.X.23.7.

Configuration

```
R3:
route-map LINK_TO_R1
 match interface Serial1/2
!
route-map LINK_TO_R2
 match interface Serial1/3
!
ip nat inside source static 155.1.7.7 155.1.13.7 route-map LINK_TO_R1
ip nat inside source static 155.1.7.7 155.1.23.7 route-map LINK_TO_R2
!
interface FastEthernet0/0
 ip nat inside
!
interface Serial1/2
 ip nat outside
!
interface Serial1/3
 ip nat outside
```

Verification

Note

Static policy NAT (static NAT with route-maps) allows the use of mappings to different inside global IP addresses for the same inside local IP. The route-maps are used to classify traffic to be used with which respective static NAT translation. Commonly this type of mapping is used on multi-homed routers where the same inside local server address maps to different inside global IP addresses out to multiple ISPs.

To verify this configuration telnet from SW1 with a source address of VLAN 7 out to R1 and R2, and issue the **`show users`** command.

Copyright © 2010 Internetwork Expert

```
Rack1SW1#telnet 155.1.13.1 /source-interface Vlan7
Trying 155.1.13.1 ... Open

User Access Verification

Password:
Rack1R1>show users
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                 00:31:47
* 66 vty 0                 idle                 00:00:00 155.1.13.7

  Interface    User               Mode         Idle     Peer Address

Rack1R1>exit

[Connection to 155.1.13.1 closed by foreign host]
Rack1SW1#telnet 155.1.23.2 /source-interface Vlan7
Trying 155.1.23.2 ... Open


User Access Verification

Password:
Rack1R2>show user
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                 00:00:12
* 66 vty 0                 idle                 00:00:00 155.1.23.7

  Interface    User               Mode         Idle     Peer Address
```

Next remove the static statements that call the route-maps and replace them with normal static extendable translations.

```
Rack1R3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#no ip nat inside source static 155.1.7.7 155.1.13.7
route-map LINK_TO_R1
Rack1R3(config)#no ip nat inside source static 155.1.7.7 155.1.23.7
route-map LINK_TO_R2
Rack1R3(config)#ip nat inside source static 155.1.7.7 155.1.13.7
extendable
Rack1R3(config)#ip nat inside source static 155.1.7.7 155.1.23.7
extendable
Rack1R3(config)#end
Rack1R3#
```

Reverify the operation of the translations by telneting from SW1 to R1 and R2 again, while sourcing the packets from VLAN 7.

Copyright © 2010 Internetwork Expert

```
Rack1SW1#telnet 155.1.13.1 /source-interface Vlan7
Trying 155.1.13.1 ... Open


User Access Verification

Password:
Rack1R1>show users
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                 00:40:52
* 66 vty 0                 idle                 00:00:00 155.1.13.7

  Interface    User               Mode         Idle     Peer Address

Rack1R1>exit

[Connection to 155.1.13.1 closed by foreign host]
Rack1SW1#telnet 155.1.23.2 /source-interface Vlan7
Trying 155.1.23.2 ... Open


User Access Verification

Password:
Rack1R2>show users
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                 00:09:19
* 66 vty 0                 idle                 00:00:00 155.1.13.7

  Interface    User               Mode         Idle     Peer Address
```

Note that in the above output both sessions are translated to the address 155.1.13.7, where in the previous output they were translated based on the exit interface when the route-maps were used for classification.

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.25 NAT with Overlapping Subnets

- Create a new Loopback1 interface on R1 and R2 using the IP addresses 10.0.0.1/24 and 10.0.0.2/24 respectively.
- Advertise the 10.0.0.0/24 network into RIP on R2.
- Configure NAT on R1 so that when R2 telnets to the address 11.0.0.Z from its Loopback1 address it is redirected to the address 10.0.0.Z attached to R1, where "Z" is any number.
- The host 10.0.0.Z on R1 should see the packets from R2 as if they were sourced from the network 22.0.0.0/24.

Configuration

```
R1:
interface Loopback 1
 ip address 10.0.0.1 255.255.255.0
 ip nat inside
!
ip nat pool R2_MASQUARADE 22.0.0.1 22.0.0.254 prefix-length 24
!
ip access-list extended R2_LOOPBACK1
 permit ip 10.0.0.0 0.0.0.255 any
!
ip nat outside source list R2_LOOPBACK1 pool R2_MASQUARADE
ip nat inside source static network 10.0.0.0 11.0.0.0 /24
!
ip route 11.0.0.0 255.255.255.0 Null 0
ip route 22.0.0.0 255.255.255.0 Serial 0/1
!
router rip
 redistribute static

R2:
interface Loopback 1
 ip address 10.0.0.2 255.255.255.0
!
router rip
 network 10.0.0.0
```

Copyright © 2010 Internetwork Expert

Verification

Note

This particular lab shows how to deal with overlapping subnets by configuring just one router. A better approach in a real-world design would be to create static network-to-network mappings on both R1 and R2 to mask their overlapping 10.0.0.0/24 segments, and never advertise the overlapping subnet in the first place.

In this case R2 still advertises its subnet 10.0.0.0/24 to the rest of the network, but R1 hides its Loopback1 subnet using a static NAT rule. Based on this R1 needs to be configured so that it never sees packets sourced from the 10.0.0.0/24 subnet on R2.

To verify the scenario enable NAT debugging on R1 and source ping packets off the Loopback1 interface of R2. Note that **ip nat inside** command on the Loopback1 interface of R1 is not necessary in this design, but it is needed in a real design where the are actual hosts behind the masqueraded segment.

```
Rack1R1#debug ip nat detailed
IP NAT detailed debugging is on
Rack1R1#

Rack1R2#ping 11.0.0.1 source loopback 1 repeat 2

Type escape sequence to abort.
Sending 2, 100-byte ICMP Echos to 11.0.0.1, timeout is 2 seconds:
Packet sent with a source address of 10.0.0.2
!!
Success rate is 100 percent (2/2), round-trip min/avg/max = 60/62/64 ms
```

Per the NAT outside rule, 10.0.0.2 is translated to 22.0.0.2 using the dynamic pool configured. Note that 11.0.0.1 translates to 10.0.0.1 thanks to the static subnet mapping. Eventually, R1 sees the original packet from 10.0.0.2 to 11.0.0.1 as a packet from 22.0.0.2 to 10.0.0.1.

```
Rack1R1#
NAT*: o: icmp (10.0.0.2, 20) -> (11.0.0.1, 20) [127]
NAT*: o: icmp (10.0.0.2, 20) -> (11.0.0.1, 20) [127]
NAT*: s=10.0.0.2->22.0.0.2, d=11.0.0.1 [127]
NAT*: s=22.0.0.2, d=11.0.0.1->10.0.0.1 [127]
```

Copyright © 2010 Internetwork Expert

The response packet from 10.0.0.1 to 22.0.0.2 is translated using two rules. First, 10.0.0.1 translates back to 11.0.0.1, and secondly, 22.0.0.2 translates to 10.0.0.2. Note that before the translation occurs R1 will try to look up a route for 22.0.0.2; this is why you need a static route on R1. The rest of the network does not need this static route as they know how to get to the 10.0.0.0/24 subnet on R2 because of the dynamic RIP route.

```
NAT: i: icmp (10.0.0.1, 20) -> (22.0.0.2, 20) [127]
NAT: s=10.0.0.1->11.0.0.1, d=22.0.0.2 [127]
NAT: s=11.0.0.1, d=22.0.0.2->10.0.0.2 [127]
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.26 TCP Load Distribution with NAT

- Remove the previous NAT configurations on R5.
- Configure rotary NAT R5 so that when SW2 telnets to the address 155.X.58.55 it is redirected to R1, R2, and R3 in an even distribution.

Configuration

```
R5:
interface FastEthernet0/0
 ip nat outside
!
interface Serial0/0/0
 ip nat inside
!
ip nat pool ROTARY prefix-length 24 type rotary
 address 155.1.0.1 155.1.0.1
 address 155.1.0.2 155.1.0.2
 address 155.1.0.3 155.1.0.3
!
ip access-list extended LOAD_BALANCE
 permit tcp any host 155.1.58.55 eq telnet
!
ip nat inside destination list LOAD_BALANCE pool ROTARY
!
ip alias 155.1.58.55 23
```

Verification

Note

NAT supports simple load-distribution for TCP based applications using the concept of rotary NAT pools. In this particular case the inside destination translation checks traffic as it comes in the outside interface. If the traffic is matched byt access-list LOAD_BALANCE, which equates to telnet traffic going to 155.1.58.55, the destination address is changed to 155.1.0.1, 155.1.0.2, and 155.1.0.3 in a round-robin (rotary) fashion per flow.

A more scalable approach to this same design is to use IOS Server Load Balancing (SLB) feature and NAT.

To verify the configuration telnet from SW2 to R5 three times in a row and see how the destination changes every time.

Copyright © 2010 Internetwork Expert

```
Rack1SW2#telnet 155.1.58.55
Trying 155.1.58.55 ... Open


User Access Verification

Password: cisco
Rack1R1>exit

[Connection to 155.1.58.55 closed by foreign host]

Rack1SW2#telnet 155.1.58.55
Trying 155.1.58.55 ... Open


User Access Verification

Password: cisco
Rack1R2>exit

[Connection to 155.1.58.55 closed by foreign host]

Rack1SW2#telnet 155.1.58.55
Trying 155.1.58.55 ... Open


User Access Verification

Password: cisco
Rack1R3>exit

[Connection to 155.1.58.55 closed by foreign host]
Rack1SW2#

Rack1R5#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
tcp 155.1.58.55:23     155.1.0.1:23       155.1.58.8:11004   155.1.58.8:11004
tcp 155.1.58.55:23     155.1.0.2:23       155.1.58.8:11005   155.1.58.8:11005
tcp 155.1.58.55:23     155.1.0.3:23       155.1.58.8:11006   155.1.58.8:11006
```

Copyright © 2010 Internetwork Expert

13.27 Stateful NAT with HSRP

- Remove all previous NAT configurations on R4 and R6.
- Configure HSRP on R4 and R6 with the group name "VLAN146" so that hosts on VLAN 146 can use the virtual IP address 155.X.146.254 as their default gateway.
- Confgure R6 to be the active physical gateway unless the line protocol of the link connecting to BB1 goes down.
- Set the hello interval to 1 second and dead interval to 3 seconds for faster failed gateway detection.
- Authenticate HSRP message exchanges using an MD5 hash of the key CISCO.
- Create a single NAT pool on R4 and R6 with the IP addresses in range 155.X.254.1-155.X.254.254.
- Configure BGP AS 100 on R4 and R6, and peer these routers with BB3 and BB1 respectively, who are in AS 54.
- Advertise the NAT pool via EBGP to BB1 and BB3.
- Redistribute all BGP routes into RIP on R4 and R6, and ensure that R6 is the preferred exit point.
- Configure dynamic NAT on R4 and R6 so that traffic from the 155.X.0.0/16 network is port translated to the previously defined NAT pool.
- Ensure R4 and R6 synchronize their NAT entries using HSRP so that TCP sessions from the inside network connected to devices in AS 54 are not dropped if R6 loses connectivity to BB1.

Configuration

```
R4:
interface FastEthernet 0/1
 standby 146 ip 155.1.146.254
 standby 146 timers 1 3
 standby 146 preempt
 standby 146 authentication md5 key-string CISCO
 standby 146 name VLAN146
 ip nat inside
!
interface FastEthernet0/0
 ip nat outside
!
ip nat stateful id 2
 redundancy VLAN146
  mapping-id 1
!
ip nat pool SHARED_POOL 155.1.254.1 155.1.254.254 prefix-length 24
!
ip route 155.1.254.0 255.255.255.0 Null 0
!
```

Copyright © 2010 Internetwork Expert

```
router bgp 100
 neighbor 204.12.1.254 remote-as 54
 network 155.1.254.0 mask 255.255.255.0
!
router rip
 redistribute bgp 100 metric 5
!
ip access-list standard NAT_LIST
 permit 155.1.0.0 0.0.255.255
!
ip nat inside source list NAT_LIST pool SHARED_POOL mapping 1

R6:
interface FastEthernet 0/0.146
 standby 146 ip 155.1.146.254
 standby 146 timers 1 3
 standby 146 preempt
 standby 146 authentication md5 key-string CISCO
 standby 146 name VLAN146
 standby 146 priority 110
 standby 146 track Serial 0/0/0
 ip nat inside
!
interface Serial0/0/0
 ip nat outside
!
ip nat stateful id 1
 redundancy VLAN146
  mapping-id 1
!
ip nat pool SHARED_POOL 155.1.254.1 155.1.254.254 prefix-length 24
!
ip route 155.1.254.0 255.255.255.0 Null 0
!
router bgp 100
 neighbor 54.1.1.254 remote-as 54
 network 155.1.254.0 mask 255.255.255.0
!
router rip
 redistribute bgp 100 metric 1
!
ip access-list standard NAT_LIST
 permit 155.1.0.0 0.0.255.255
!
ip nat inside source list NAT_LIST pool SHARED_POOL mapping 1
```

Copyright © 2010 Internetwork Expert

Verification

Note

The idea of stateful NAT is to allow two redundant exit points using the same NAT policy to exchange their NAT translation tables. This allows for stateful failover when the active forwarder fails.

To verify this check the Stateful NAT status first.

```
Rack1R6#sh ip snat distributed verbose

Stateful NAT Connected Peers

SNAT: Mode IP-REDUNDANCY :: ACTIVE
    : State READY
    : Local Address 155.1.146.6
    : Local NAT id 1
    : Peer Address 155.1.146.4
    : Peer NAT id 2
    : Mapping List 1
    : InMsgs 5, OutMsgs 0, tcb 0x84835C00, listener 0x0

User Access Verification

Rack1R4#sh ip snat distributed verbose

Stateful NAT Connected Peers

SNAT: Mode IP-REDUNDANCY :: STANDBY
    : State READY
    : Local Address 155.1.146.4
    : Local NAT id 2
    : Peer Address 155.1.146.6
    : Peer NAT id 1
    : Mapping List 1
    : InMsgs 39, OutMsgs 0, tcb 0x6408D040, listener 0x0
```

Now verify that SNAT peers change roles as R6 loses its active HSRP status:

```
Rack1R4#show standby
Ethernet0/1 - Group 146
  State is Active
    8 state changes, last state change 00:01:29
  Virtual IP address is 155.1.146.254
  Active virtual MAC address is 0000.0c07.ac92
    Local virtual MAC address is 0000.0c07.ac92 (v1 default)
  Hello time 1 sec, hold time 3 sec
    Next hello sent in 0.576 secs
  Authentication MD5, key-string "CISCO"
  Preemption enabled
```

Copyright © 2010 Internetwork Expert

```
  Active router is local
  Standby router is 155.1.146.6, priority 90 (expires in 2.716 sec)
  Priority 100 (default 100)
  IP redundancy name is "VLAN146" (cfgd)

Rack1R4#show ip snat distributed verbose

Stateful NAT Connected Peers

SNAT: Mode IP-REDUNDANCY :: ACTIVE
    : State READY
    : Local Address 155.1.146.4
    : Local NAT id 2
    : Peer Address 155.1.146.6
    : Peer NAT id 1
    : Mapping List 1
    : InMsgs 6, OutMsgs 0, tcb 0x6408D040, listener 0x0
```

Open a telnet session to a backbone network (BB1) and make sure the translation appears in the NAT table of R4.

```
Rack1R3#telnet 112.0.0.1
Trying 112.0.0.1 ... Open

Rack1R4#show ip nat translations | inc 155.1.0.3
tcp 155.1.254.4:65148  155.1.0.3:65148    112.0.0.1:23       112.0.0.1:23
```

Ensure R4 installed this NAT entry in R6's translation table using SNAT

```
Rack1R6#show ip snat peer 155.1.146.4

Show NAT Entries created by peer: 155.1.146.4

Pro Inside global      Inside local       Outside local      Outside global
tcp 155.1.254.4:65148  155.1.0.3:65148    112.0.0.1:23       112.0.0.1:23
```

Make R6 the active router now by enabling its backbone connection and shutting down the backbone connection of R4. Ensure that R3 never loses its telnet connection, but just experiences a temporary delay.

```
Rack1R6#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#interface serial 0/0/0
Rack1R6(config-if)#no shutdown

Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface Fastethernet 0/0
Rack1R4(config-if)#shutdown
```

Copyright © 2010 Internetwork Expert

```
Rack1AS>3
[Resuming connection 3 to R3 ... ]

RS.1.1.BB1>
RS.1.1.BB1>
```

Verify the NAT translation table of R6 and ensure it's now identical to the table used by R4.

```
Rack1R6#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
tcp 155.1.254.4:65148  155.1.0.3:65148    112.0.0.1:23       112.0.0.1:23
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.28 Stateful NAT with Primary/Backup

- Remove the HSRP configuration on R4 and R6.
- Modify the SNAT configuration on these devices so it does not rely on HSRP status for tracking of the active SNAT router.
- R6 should be the primary and R4 should be the backup router.

Configuration

```
R6:
interface FastEthernet0/0.146
 no standby 146
!
ip nat Stateful id 1
 no redundancy VLAN146
     primary 155.1.146.6
         peer 155.1.146.4
         mapping-id 1

R4:
interface FastEthernet0/1
 no standby 146
!
ip nat Stateful id 2
 no redundancy VLAN146
     backup 155.1.146.4
         peer 155.1.146.6
         mapping-id 1
```

Verification

> **Note**
>
> Stateful NAT with Primary/Backup replaces the need for HSRP tracking by using a built in SNAT keepalive. To verify the operation of this without HSRP source a telnet connection from R3 to BB1, and ensure that R6 installs the translation entry in the NAT table of R4.

```
Rack1R6#show ip snat distributed verbose

Stateful NAT Connected Peers

SNAT: Mode PRIMARY
    : State READY
    : Local Address 155.1.146.6
    : Local NAT id 1
    : Peer Address 155.1.146.4
    : Peer NAT id 2
    : Mapping List 1
    : InMsgs 5, OutMsgs 0, tcb 0x857FE654, listener 0x0
```

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ip snat distributed verbose

Stateful NAT Connected Peers

SNAT: Mode BACKUP
    : State READY
    : Local Address 155.1.146.4
    : Local NAT id 2
    : Peer Address 155.1.146.6
    : Peer NAT id 1
    : Mapping List 1
    : InMsgs 15, OutMsgs 0, tcb 0x650FE6E0, listener 0x650FE22C
```

```
Rack1R6#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
tcp 155.1.254.1:26887  155.1.146.1:26887  112.0.0.1:23       112.0.0.1:23
```

```
Rack1R4#show ip snat peer 155.1.146.6

Show NAT Entries created by peer: 155.1.146.4

Pro Inside global      Inside local       Outside local      Outside global
tcp 155.1.254.1:26887  155.1.146.1:26887  112.0.0.1:23       112.0.0.1:23
```

Copyright © 2010 Internetwork Expert

13.29 NAT Virtual Interface

- Remove all previous NAT configurations on R5.
- Configure R5 so that traffic sourced from SW2's VLAN 8 network is translated to the source address range 155.X.188.0/24.
- Do not use the **ip nat inside** or **ip nat outside** commands or static routing to accomplish this.

Configuration

```
R5:
interface Serial 0/0/0
 ip nat enable
!
interface Serial 0/1/0
 ip nat enable
!
interface FastEthernet 0/0
 ip nat enable
!
ip access-list standard VLAN8
 permit 155.1.8.0 0.0.0.255
!
ip nat pool NVI_POOL 155.1.188.1 155.1.188.254 prefix 24 add-route
!
ip nat source list VLAN8 pool NVI_POOL
!
router rip
 redistribute static metric 1
```

Verification

> **Note**
>
> To verify translation using the NVI disable CEF on R5's interfaces (no ip route-cache), generate traffic from SW2's VLAN 8 interface, and issue the following debugs.

```
Rack1R5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#access-list 100 permit icmp any any
Rack1R5(config)#end
Rack1R5#
Rack1R5#debug ip packet detail 100
IP packet debugging is on (detailed) for access list 100
Rack1R5#debug ip nat detail
IP NAT detailed debugging is on
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#ping 155.1.45.4 source vlan8 repeat 1

Rack1R5#
IP: tableid=0, s=155.1.8.8 (FastEthernet0/0), d=155.1.45.4
(Serial0/1/0), routed via FIB
NAT: i: icmp (155.1.8.8, 10) -> (155.1.45.4, 10) [41]
NAT: s=155.1.8.8->155.1.188.2, d=155.1.45.4 [41]
IP: tableid=0, s=155.1.188.2 (FastEthernet0/0), d=155.1.45.4
(Serial0/1/0), routed via RIB
IP: s=155.1.188.2 (FastEthernet0/0), d=155.1.45.4 (Serial0/1/0),
g=155.1.45.4, len 100, forward
    ICMP type=8, code=0
```

Note the NAT direction is always "inside" for NVI based NAT. This means a routing lookup is always performed before the translation. After the routing decision is made the packet source is translated, then the packet is forwarded down the Serial interface.

```
IP: tableid=0, s=155.1.45.4 (Serial0/1/0), d=155.1.188.2 (NVI0), routed
via RIB
NAT: i: icmp (155.1.45.4, 10) -> (155.1.188.2, 10) [41]
NAT: s=155.1.45.4, d=155.1.188.2->155.1.8.8 [41]
IP: tableid=0, s=155.1.45.4 (Serial0/1/0), d=155.1.8.8
(FastEthernet0/0), routed via RIB
```

When the return packet comes back it is routed using the NVI-based static route that is automatically. Note that NAT direction is "inside" again - this is why the NVI route is needed in this situation. With normal NAT outside to inside translation (the return traffic flow in this case) the packet destination would first get untranslated, and then routed using the true destination.

```
Rack1R5#show ip route static
     155.1.0.0/24 is subnetted, 17 subnets
S       155.1.188.0 [0/0] via 0.0.0.0, NVI0
```

Note that the NVI based static route eliminates the need for a statically configured route to Null0, but it does not remove the need of advertising the static route into the dynamic routing domain with the redistribute static statement.

To verify translation issue the show ip nat nvi translations command, not the normal show ip nat translations command.

```
Rack1R5#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global

Rack1R5#show ip nat nvi translations
Pro Source global      Source local       Destin  local      Destin  global
--- 155.1.188.2        155.1.8.8          ---                ---
```

Copyright © 2010 Internetwork Expert

13.30 NAT Default Interface

- Remove all previous NAT configurations on R5.
- Disable R5's connection to the Frame Relay network.
- Configure R5's point-to-point link to R4 as a passive-interface under RIP.
- Configure R5 so that all connections destined to its point-to-point Serial link are redirected to SW2's Loopback 0 interface.
- All other inside to outside traffic on R5 should be dynamically overloaded to the point-to-point Serial link's IP address.

Configuration

```
R5:
interface FastEthernet 0/0
 ip nat inside
!
interface Serial 0/1/0
 ip nat outside
!
interface Serial 0/0/0
 shutdown
!
router rip
 passive-interface Serial0/1/0
!
ip access-list standard ALL
 permit any
!
ip nat inside source list ALL interface Serial 0/1/0 overload
ip nat inside source static 150.1.8.8 interface Serial 0/1/0
```

Copyright © 2010 Internetwork Expert

Verification

Note

The NAT Default Interface feature allows all traffic received on the outside interface that does not already match an existing dynamic translation to be statically forwarded to an inside host. This feature is useful for applications that use different outbound versus inbound traffic flows.

To verify this configuration telnet to any outside destination from the inside devices, and view the translation table on R5.

```
Rack1SW2#telnet 150.1.1.1
Trying 150.1.1.1 ... Open

User Access Verification

Password:
Rack1R1>show users
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                 01:51:29
* 66 vty 0                 idle                 00:00:00 155.1.45.5

  Interface    User               Mode         Idle     Peer Address

Rack1SW4#telnet 155.1.146.6
Trying 155.1.146.6 ... Open

User Access Verification

Password:
Rack1R6>show users
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                 01:51:25
* 66 vty 0                 idle                 00:00:00 155.1.45.5

  Interface    User               Mode         Idle     Peer Address
```

Copyright © 2010 Internetwork Expert

All inside traffic sent outbound is translated to R5's interface address 155.1.45.5.

```
Rack1R5#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
--- 155.1.45.5         150.1.8.8          ---                ---
tcp 155.1.45.5:15418   155.1.58.8:15418   150.1.1.1:23       150.1.1.1:23
tcp 155.1.45.5:19938   155.1.108.10:19938 155.1.146.6:23     155.1.146.6:23
```

All outside traffic send inbound is redirected to the default host of SW2.

```
Rack1R4#telnet 155.1.45.5
Trying 155.1.45.5 ... Open

User Access Verification

Password:
Rack1SW2>
```

```
Rack1R5#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
tcp 155.1.45.5:23      150.1.8.8:23       155.1.45.4:32493   155.1.45.4:32493
--- 155.1.45.5         150.1.8.8          ---                ---
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.31 Reversible NAT

- Remove all previous NAT configurations on R5.
- Configure R5 so that all traffic received on the inside interface connecting to VLAN 58 is translated to the pool 155.X.45.100 – 155.X.45.200 as it exits out the point-to-point link to R4.
- Use route-map based NAT for this configuration to allow outside hosts to initiate connections back to hosts on the inside network once an inside to outside dynamic entry is created in the NAT table.

Configuration

```
R5:
interface FastEthernet 0/0
 ip nat inside
!
interface Serial 0/1/0
 ip nat outside
!
ip nat pool POOL 155.1.45.100 155.1.45.200 prefix-length 24
!
ip access-list standard INSIDE_HOSTS
 permit 155.1.0.0 0.0.255.255
 permit 150.1.0.0 0.0.255.255
!
route-map CREATE_EXTENDABLE_ENTRIES
 match ip address INSIDE_HOSTS
 match interface Serial0/1/0
!
ip nat inside source route-map CREATE_EXTENDABLE_ENTRIES pool POOL
reversible
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> By default when you use route-maps with NAT rules, extendable entries are created. This disallows an external user to open a reverse connection back to an inside host since no one-to-one mapping exists in translation table. Reversible NAT allows creating of extendable entries along with reversible one-to-one mappings.
>
> To verify this configuration, attempt to open a connection from the outside network to hosts on the inside network using the inside global NAT pool.

```
Rack1R6#telnet 155.1.45.101
Trying 155.1.45.101 ...
% Connection timed out; remote host not responding

Rack1R6#telnet 155.1.45.102
Trying 155.1.45.102 ...
% Connection timed out; remote host not responding

Rack1R6#telnet 155.1.45.103
Trying 155.1.45.103 ...
% Connection timed out; remote host not responding

Rack1R6#telnet 155.1.45.104
Trying 155.1.45.104 ...
% Connection timed out; remote host not responding
```

These connections are denied because R5 has no entries installed in the NAT table.

```
Rack1R5#show ip nat translations
```

Copyright © 2010 Internetwork Expert

Next, initiate a connection from the inside network to the outside network and verify R5's translation table.

```
Rack1SW4#telnet 155.1.146.6
Trying 155.1.146.6 ... Open

User Access Verification

Password:
Rack1R6>show users
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                 00:05:26
* 66 vty 0                 idle                 00:00:00 155.1.45.102

  Interface    User               Mode         Idle     Peer Address

Rack1R5#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
tcp 155.1.45.103:33927 155.1.108.10:33927 155.1.146.6:23     155.1.146.6:23
--- 155.1.45.103       155.1.108.10       ---                ---
```

Note that R5 creates both an extendable entry and a one-to-one entry. Once this one-to-one entry is created, the connection is reversible, and hosts on the outside network can initiate connections to hosts on the inside.

```
Rack1R1#telnet 155.1.45.103
Trying 155.1.45.103 ... Open

User Access Verification

Password:
Rack1SW4>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.32 Static Extendable NAT

- Configure R5 so that SW4's Loopback0 address is reachable from the outside of the network using the IP addresses 155.1.45.201 and 155.1.45.202 at the same time.

Configuration

```
R5:
ip nat inside source static 150.1.10.10 155.1.45.201 extendable
ip nat inside source static 150.1.10.10 155.1.45.202 extendable
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Extendable static NAT allows you to configure multiple static mappings for the same local or global IP address. In this example the same local IP address is mapped to multiple global IP addresses. The extendable keyword allows every new translation to be fully extended, without binding a local IP address to a fixed global IP.
>
> To verify this telnet to both of the global IP addresses mapped to the local IP addresses of SW4.

```
Rack1R5#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
--- 155.1.45.201       150.1.10.10        ---                ---
--- 155.1.45.202       150.1.10.10        ---                ---
```

> Before traffic is initiated R5 does not install extendable entries for the static mappings.

```
Rack1R6#telnet 155.1.45.201
Trying 155.1.45.201 ... Open

User Access Verification

Password:
Rack1SW4>exit

[Connection to 155.1.45.201 closed by foreign host]

Rack1R6#telnet 155.1.45.202
Trying 155.1.45.202 ... Open

User Access Verification

Password:
Rack1SW4>exit
```

Copyright © 2010 Internetwork Expert

Once traffic is sent the translations are fully extended.

```
Rack1R5#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
tcp 155.1.45.201:23    150.1.10.10:23     155.1.146.6:57575  155.1.146.6:57575
tcp 155.1.45.202:23    150.1.10.10:23     155.1.146.6:58011  155.1.146.6:58011
--- 155.1.45.201       150.1.10.10        ---                ---
--- 155.1.45.202       150.1.10.10        ---                ---
```

The rule that appears first in the configuration is used to resolve the ambiguity when translating from one to many addresses. This can be verified from initiating connections from the inside out.

```
Rack1SW4#telnet 150.1.1.1 /source-interface Loopback0
Trying 150.1.1.1 ... Open

User Access Verification

Password:
Rack1R1>show user
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                 00:04:39
* 66 vty 0                 idle                 00:00:00 155.1.45.201

  Interface    User               Mode         Idle     Peer Address

Rack1SW4#telnet 150.1.6.6 /source-interface Loopback0
Trying 150.1.6.6 ... Open

User Access Verification

Password:
Rack1R6>show user
    Line       User       Host(s)              Idle       Location
   0 con 0                 idle                 00:04:24
* 66 vty 0                 idle                 00:00:00 155.1.45.201

  Interface    User               Mode         Idle     Peer Address
```

Static policy NAT with route-maps can be used to explicitly specify which NAT rules to use and when they are used (i.e. per interface).

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

> **Note**
>
> Revert all devices to the *IP Services* initial configurations prior to continuing.

13.33 IP Precedence Accounting

- Configure R6 to account for the number of input and output packets based on their IP Precedence values on it's connection to BB1.

Configuration

```
R6:
interface Serial 0/0/0
 ip accounting precedence input
 ip accounting precedence output
```

Verification

> **Note**
>
> Routing protocols by default use an IP precedence value of 6. ICMP ping uses a default IP precedence of 0. These packets can be used to test the precedence accounting.

```
Rack1R6#ping 212.18.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 212.18.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/33 ms
Rack1R6#
```

Copyright © 2010 Internetwork Expert

Extended ping can be used to change the IP precedence of the packets generated.

```
Rack1R6#ping
Protocol [ip]:
Target IP address: 212.18.0.1
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface:
```

The first three bits of of the Type of Service (TOS) field are IP precedence. Therefore a precedence of 3 in binary would be 01100000, or 96 in decimal. A shortcut for this is to take the precedence value and multiple it by 32 to get its decimal representation (i.e. 3 X 32 = 96).

```
Type of service [0]: 96
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 112.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/36 ms
```

To verify these statistics accumulated by the Serial interface issue the **show interface [*interface*] precedence** command. Use the **clear counters** command to clear these statistics.

```
Rack1R6#show interfaces serial 0/0/0 precedence
Serial0/0/0
  Input
    Precedence 0:  5 packets, 520 bytes
    Precedence 3:  5 packets, 520 bytes
    Precedence 6:  712 packets, 58455 bytes
  Output
    Precedence 0:  5 packets, 520 bytes
    Precedence 3:  5 packets, 520 bytes
    Precedence 6:  62 packets, 3634 bytes
```

Copyright © 2010 Internetwork Expert

13.34 IP Output Packet Accounting

- Configure R1 to account for the number of output packets on all physical interfaces.
- Only account for packets going to subnets of 155.X.0.0/16.
- Limit the accounting database size to 4096 entries.
- The number of records for packets not matching the account list should be set to 1.

Configuration

```
R1:
ip accounting-threshold 4096
ip accounting-transits 1
ip accounting-list 155.1.0.0 0.0.255.255
!
interface FastEthernet 0/0
 ip accounting output-packets
!
interface Serial 0/0
 ip accounting output-packets
!
interface Serial 0/1
 ip accounting output-packets
```

Verification

Note

IP accounting for output packets keeps local statistics of source & destination pairs, packet counts, and byte counts for packets leaving a particular link. The accounting-list matches either sources or destinations of outgoing packets that should be accounted for. Transit-records are accounting records created for packets not matching any of the accounting lists.

Those records do not contain detailed information, but instead just a notification of the first packet in the flow. By default if there is no accounting-list configured the value of the transit records is ignored. The default number of transit records is 0, so when you enable an accounting-list all non-matching packets are not accounted for.

Copyright © 2010 Internetwork Expert

To test this configuration, generate some traffic to populate the accounting database.

```
Rack1R3#ping 155.1.146.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/31/32 ms

Rack1R3#ping 150.1.6.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.6.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/31/32 ms

Rack1R3#ping 150.1.6.6 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.6.6, timeout is 2 seconds:
Packet sent with a source address of 150.1.3.3
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/45/108 ms

Rack1R3#ping 150.1.4.4 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
Packet sent with a source address of 150.1.3.3
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/29/32 ms
```

Copyright © 2010 Internetwork Expert

Issue the **show ip accounting** command to view the accounting database. In this case we should see entries for flows that match the accounting-list (from or to the 155.X.0.0/16 subnet). Addtionally there should be just one entry for the packet exchange between R3's Loopback 0 and R6's Loopback 0. This is because the transit-record limit is set to 1. The second traffic flow from R3's Loopback 0 to R4's Loopback0 will exceed the transit-record limit and therefore won't be recorded.

```
Rack1R1#show ip accounting
   Source           Destination           Packets               Bytes
 150.1.3.3        150.1.6.6                   1                 100
 150.1.6.6        155.1.13.3                  5                 500
 155.1.13.3       150.1.6.6                   5                 500
 155.1.146.4      155.1.13.3                  5                 500
 155.1.13.3       155.1.146.4                 5                 500
```

When you clear the active accounting database it is copied to the "checkpoint" database, and accounting starts from scratch. You can retrieve the checkpoint database separately, as seen below. Every time you clear the active database it is copied to the checkpoint.

```
Rack1R1#clear ip accounting
Rack1R1#show ip accounting
   Source           Destination           Packets               Bytes

Accounting data age is 0

Rack1R1#show ip accounting checkpoint
   Source           Destination           Packets               Bytes
 150.1.3.3        150.1.6.6                   1                 100
 150.1.6.6        155.1.13.3                  5                 500
 155.1.13.3       150.1.6.6                   5                 500
 155.1.146.4      155.1.13.3                  5                 500
 155.1.13.3       155.1.146.4                 5                 500

Accounting data age is 8
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.35 IP Access Violation Accounting

- Configure R6 to deny packets with an IP precedence of 4 coming from BB1.
- Account for the packets denied by this policy.

Configuration

```
R6:
access-list 101 deny ip any any precedence 4
access-list 101 permit ip any any
!
interface Serial 0/0/0
 ip access-group 101 in
 ip accounting access-violations
```

Verification

> **Note**
>
> To test this configuration generate packets from R6 to BB1 using a TOS of 128 (IP Precedence value of 4).

```
Rack1R6#sh ip accounting access-violations
   Source          Destination             Packets               Bytes    ACL

Accounting data age is 01:34

Rack1R6#ping
Protocol [ip]:
Target IP address: 212.18.0.1
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface:
Type of service [0]: 128
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 112.0.0.1, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip accounting access-violations
   Source           Destination                Packets                 Bytes    ACL
 112.0.0.1        54.1.1.6                         5                   500    101

Accounting data age is 01:35
```

Pitfall

Access-violation accounting does not work with named access-lists.

13.36 MAC Address Accounting

- Configure R1 to account for the MAC addresses in IP packets entering or leaving its connection to VLAN 146.

Configuration

```
R1:
interface FastEthernet0/0
 ip accounting mac-address input
 ip accounting mac-address output
```

Verification

> **Note**
>
> MAC address accounting has fixed size of 512 entries for input and output frames. The below output shows source MAC addresses for input packets, and destination MAC addresses for output packets, as well as packet counts and their cumulative size.

```
Rack1R1#show interfaces fastEthernet 0/0 mac-accounting
FastEthernet0/0
  Input   (510 free)
    0006.d7bc.ed82(2  ):  21 packets, 8526 bytes, last: 6551ms ago
    0011.9221.da80(248):  21 packets, 9786 bytes, last: 4275ms ago
                  Total:  42 packets, 18312 bytes
  Output  (511 free)
    0100.5e00.0009(86 ):  21 packets, 6006 bytes, last: 1831ms ago
                  Total:  21 packets, 6006 bytes
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.37 TCP Optimization

- Modify R1's TCP configuration to meet the following requirements:
  - Avoid the TCP "silly window syndrome".
  - Enable high-performance TCP options.
  - Set the TCP window size to twice the standard 16 bit maximum value.
  - Limit the wait time for a TCP SYN response to the minimum.
  - Enable the feature to avoiding fragmentation with TCP sessions.
  - Hold no more that 16 packets in outgoing TCP queue.

Configuration

```
R1:
service nagle
ip tcp ecn
ip tcp selective-ack
ip tcp timestamp
ip tcp window-size 131070
ip tcp queuemax 16
ip tcp synwait-time 5
ip tcp path-mtu-discovery
```

Verification

> **Note**
>
> For more information on **`service nagle`** refer to RFC 896, *Congestion control in IP/TCP internetworks*. Refer to RFC 1323 for *TCP Extensions for High Performance*.
>
> To verify this configuration enable the same set of options in R6 and telnet from R6 to R1. Once the session is open examine the TCP connection status on R6 and R1.

```
Rack1R6#telnet 150.1.1.1
Trying 150.1.1.1 ... Open


User Access Verification

Password:
Rack1R1>
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#show tcp brief all
TCB        Local Address               Foreign Address         (state)
856D54A0   54.1.1.6.60615              54.1.1.254.179          ESTAB
856BE248   155.1.146.6.33209           150.1.1.1.23            ESTAB
849057EC   150.1.6.6.23                155.1.146.4.60166       ESTAB
856D4FEC   *.179                       54.1.1.254.*            LISTEN
854E4AFC   *.80                        *.*                     LISTEN

Rack1R6#show tcp tcb 856BE248
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Enabled, Mininum incoming TTL 0, Outgoing TTL 255
Local host: 155.1.146.6, Local port: 33209
Foreign host: 150.1.1.1, Foreign port: 23

Enqueued packets for retransmit: 0, input: 0  mis-ordered: 0 (0 bytes)

Event Timers (current time is 0x1080595):
Timer          Starts    Wakeups            Next
Retrans             6          0             0x0
TimeWait            0          0             0x0
AckHold             3          1             0x0
SendWnd             0          0             0x0
KeepAlive           0          0             0x0
GiveUp              0          0             0x0
PmtuAger            0          0             0x0
DeadWait            0          0             0x0

iss: 1568200946  snduna: 1568200984  sndnxt: 1568200984     sndwnd:
131032
irs: 1153885002  rcvnxt: 1153885082  rcvwnd:     130991  delrcvwnd:
79

SRTT: 165 ms, RTTO: 1172 ms, RTV: 1007 ms, KRTT: 0 ms
minRTT: 4 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: active open, retransmission timeout, retransmitting, nagle
  path mtu capable, SACK option permitted, Timestamp option used
  win-scale
IP Precedence value : 6

Datagrams (max data segment is 1448 bytes):
Rcvd: 10 (out of order: 0), with data: 7, total data bytes: 79
Sent: 11 (retransmit: 0, fastretransmit: 0, partialack: 0, Second
Congestion: 0), with data: 7, total data bytes: 37
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#show tcp brief all
TCB        Local Address               Foreign Address         (state)
848F416C   150.1.1.1.23                155.1.146.6.33209       ESTAB
854BE9AC   *.80                        *.*                     LISTEN

Rack1R1#show tcp tcb 848F416C
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Enabled, Mininum incoming TTL 0, Outgoing TTL 255
Local host: 150.1.1.1, Local port: 23
Foreign host: 155.1.146.6, Foreign port: 33209

Enqueued packets for retransmit: 0, input: 0  mis-ordered: 0 (0 bytes)

Event Timers (current time is 0x10CB15B):
Timer          Starts    Wakeups            Next
Retrans             4          0             0x0
TimeWait            0          0             0x0
AckHold             5          2             0x0
SendWnd             0          0             0x0
KeepAlive           0          0             0x0
GiveUp              0          0             0x0
PmtuAger            0          0             0x0
DeadWait            0          0             0x0

iss: 1153885002  snduna: 1153885082  sndnxt: 1153885082     sndwnd:
130990
irs: 1568200946  rcvnxt: 1568200984  rcvwnd:     131033  delrcvwnd:
37

SRTT: 124 ms, RTTO: 1405 ms, RTV: 1281 ms, KRTT: 0 ms
minRTT: 4 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: passive open, active open, retransmission timeout,
retransmitting
  nagle, SACK option permitted, Timestamp option used, win-scale
IP Precedence value : 0

Datagrams (max data segment is 1448 bytes):
Rcvd: 11 (out of order: 0), with data: 7, total data bytes: 37
Sent: 10 (retransmit: 0, fastretransmit: 0, partialack: 0, Second
Congestion: 0), with data: 7, total data bytes: 79
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.38 IOS Small Services & Finger

- Configure R1 to meet the following requirements:
  - Users who telnet or send UDP packets to R1 at port 7 should have any typed characters echoed back to them.
  - Users who telnet or send UDP packets to R1 at port 9 should have any typed characters thrown away.
  - Users who telnet to R1 at port 13 should be returned the date and time.
  - Users who telnet or send UDP packets to R1 at port 19 should be sent a stream of ASCII characters.
  - Users who telnet to R1 at port 79 should be sent a list of currently logged in users.

Configuration

```
R1:
service udp-small-servers
service tcp-small-servers
ip finger
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Connect to the well-know TCP ports on R1 to verify the small services. Use the escape sequence “Ctrl-Shift-6-6 X” to escape the session and disconnect it.

```
Rack1R1#telnet 150.1.1.1 echo
Trying 150.1.1.1, 7 ... Open
tteesstt

Rack1R1#telnet 150.1.1.1 discard
Trying 150.1.1.1, 9 ... Open
test

Rack1R1#telnet 150.1.1.1 daytime
Trying 150.1.1.1, 13 ... Open
Thursday, October 9, 2008 11:03:56-UTC

Rack1R1#telnet 150.1.1.1 chargen
Trying 150.1.1.1, 19 ... Open
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefg
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefgh
"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghi
#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghij
$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijk
%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijkl
&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklm
<snip>

Rack1R1#telnet 150.1.1.1 finger
Trying 150.1.1.1, 79 ... Open

    Line       User       Host(s)              Idle       Location
   0 con 0                 150.1.1.1            00:00:00
* 66 vty 0                 idle                 00:00:00 150.1.1.1

  Interface    User               Mode         Idle     Peer Address
```

13.39 Directed Broadcasts & UDP Forwarding

- Configure R5 so that if it receives packets for the destination address 155.X.58.255 they are forwarded out the link to SW2, but the destination should not be changed to 255.255.255.255.
- Configure R5 to listen to UDP broadcasts on the point-to-point link to R4, and forward them to the address 155.X.58.255.
- This forwarding should only occur for DNS packets.
- Verify this configuration by sending ICMP pings to the address 155.X.58.255 and DNS broadcast requests from R4.

Configuration

```
R5:
no ip forward-protocol udp bootps
no ip forward-protocol udp tftp
no ip forward-protocol udp time
no ip forward-protocol udp netbios-ns
no ip forward-protocol udp netbios-dgm
no ip forward-protocol udp tacacs
!
interface Serial 0/1/0
 ip helper-address 155.1.58.255
!
interface FastEthernet 0/0
 ip broadcast-address 155.1.58.255
 ip directed-broadcast
```

Verification

Note

A directed broadcast packet is a packet whose destination address matches the last IP address in a subnet assigned to the local device. For example the network 10.0.0.0/8 has a directed broadcast address, or subnet broadcast address, of 10.255.255.255. The directed broadcast feature, which is disabled by default, allows remote devices to send a broadcast packet to a particular link by sending the packet to the directed broadcast address. Normally this feature is not allowed due to security risks inherent to broadcasts, such as smurf and fraggle DoS attacks.

When the router receives a directed broadcast, and the **`ip directed-broadcast`** command is enabled on the outgoing interface where the address match occurs, the router changes the destination address in the packet from the directed broadcast address to the all subnet broadcast address of 255.255.255.255. This behavior can be overridden with the **`ip broadcast-address`** command, which it is in the above case.

Copyright © 2010 Internetwork Expert

The UDP forwarding feature, controlled by the **ip helper-address**, is used to take broadast UDP packets and change the destination address to a unicast or directed broadcast address. By default the router will forward UDP packets only for the following protocols: TACACS (not TACACS+), TFTP, BOOTP, Time, NetBIOS Name Server and NetBIOS Datagram Services and DNS. In the above case all of these protocols are disabled with the exception of DNS.

To verify this configuration send packets from R4 to broadcast IP address of VLAN 58. Enable IP packet debugging for ICMP packets on R5 and SW2 to view the results.

```
R5, SW2:
access-list 100 permit icmp any any

Rack1R5#debug ip packet detail 100
Rack1SW2#debug ip packet detail 100

Rack1R4#ping 155.1.58.255

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.58.255, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/33/36 ms

Rack1R5#
IP: =155.1.45.4 (Serial0/1/0), d=155.1.58.255 (Ethernet0/0), routed via
RIB
IP: s=155.1.45.4 (Serial0/1/0), d=155.1.58.255 (Ethernet0/0),
g=255.255.255.255, len 100, forward directed broadcast
    ICMP type=8, code=0
IP: s=155.1.45.4 (Serial0/1/0), d=155.1.58.255 (Ethernet0/0), len 100,
rcvd 5
    ICMP type=8, code=0
IP: tableid=0, s=155.1.45.5 (local), d=155.1.45.4 (Serial0/1/0), routed
via FIB
IP: s=155.1.45.5 (local), d=155.1.45.4 (Serial0/1/0), len 100, sending
    ICMP type=0, code=0
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#
IP: s=155.1.45.4 (Vlan58), d=155.1.58.255 (Vlan58), len 100, rcvd 3
    ICMP type=8, code=0
IP: s=155.1.45.4 (Vlan58), d=155.1.58.255, len 100, stop process pak
for forus packet
    ICMP type=8, code=0
IP: s=155.1.58.8 (local), d=155.1.45.4 (Vlan58), len 100, sending
    ICMP type=0, code=0
```

R5 receives the directed broadcast and forwards it onto VLAN 58. SW2 receives the packet, processes the packet locally, and sends an ICMP echo reply back to R4. This behavior is considered a security risk and is exploited in the smurf and fraggle attacks. These attacks send ICMP or UDP echo packets to a directed broadcast address with a spoofed source. When the destination segment receives the broadcast, all hosts on the segment reply back to the spoofed source (the victim). If the link is a /24 subnet that is fully populated with 253 hosts, it means that for every one request sent from the attacker 253 replies are sent back to the victim. To prevent this the **ip directed-broadcast** command is simply disabled at the interface level.

To test the UDP forwarding enable DNS name lookup on R4 without configuring a DNS server. This causes the router to send DNS requests to the all subnet broadcast address of 255.255.255.255 out all links. When R5 hears this packet it changes the destination address to the helper, 155.1.58.255, and forwards it as a directed broadcast to VLAN 58.

In a real network design this behavior can be useful for DNS and DHCP forwarding. If there are multiple DNS or DHCP servers on a segment, traffic can be load shared to the fastest server by sending the name or address request to both at the same time. Whichever server replies first will be the one that is used for that particular application.

```
Rack1R4#conf t
Rack1R4#ip domain-lookup
Rack1R4(config)#access-list 100 permit udp any any eq 53
Rack1R4#debug ip packet detail 100

Rack1R4#testname
Translating "testname"...domain server (255.255.255.255)
IP: s=204.12.1.4 (local), d=255.255.255.255 (Ethernet0/0), len 54,
sending broad/multicast
UDP src=54655, dst=53
IP: s=155.1.0.4 (local), d=255.255.255.255 (Serial0/0), len 54, sending
broad/multicast
UDP src=54655, dst=53
IP: s=155.1.146.4 (local), d=255.255.255.255 (Ethernet0/1), len 54,
sending broad/multicast
UDP src=54655, dst=53
```

Copyright © 2010 Internetwork Expert

SW2 receives the DNS request sent from R4 as a directed broadcast through R5.

```
Rack1SW2#
IP: s=155.1.45.4 (Vlan58), d=155.1.58.255 (Vlan58), len 54, rcvd 3
    UDP src=54655, dst=53
IP: s=155.1.45.4 (Vlan58), d=155.1.58.255, len 54, stop process pak for
forus packet
    UDP src=54655, dst=53
IP: s=155.1.45.4 (Vlan58), d=155.1.58.255 (Vlan58), len 54, rcvd 3
    UDP src=54655, dst=53
IP: s=155.1.45.4 (Vlan58), d=155.1.58.255, len 54, stop process pak for
forus packet
```

Copyright © 2010 Internetwork Expert

13.40 DRP Server Agent

- Configure R1 to support reporting of information to a Distributed Director.
- Only allow connections from Directors from VLAN 146, authenticate the communication using the key value CISCO.

Configuration

```
R1:
ip drp access-group 99
ip drp authentication key-chain DRP
ip drp server
!
key chain DRP
 key 1
  key-string CISCO
!
access-list 99 permit 155.1.146.0 0.0.0.255
```

Verification

```
Rack1R1#show ip drp
Director Responder Protocol Agent is enabled
0 director requests:
0 successful route table lookups
0 successful measured lookups
0 no route in table
0 nortt
0 DRP packet failures returned
0 successful echos
0 Boomerang requests
0 Boomerang-raced DNS responses
Authentication is enabled, using "DRP" key-chain
Director requests filtered by access-list 99
rttprobe source port is     : 53
rttprobe destination port is: 53
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.41 WCCPv1 Web-Cache

- Configure R4 and R6 for WCCPv1 web-cache redirection.
- R4 should redirect outgoing packets on its VLAN 43 interface, and R6 should redirect incoming packets on its VLAN 146 connection.
- Exclude packets coming in R4's Serial connection to R5 from redirection.
- Exclude connections sourced from the VLAN 146 subnet from redirection.
- Do not redirect flows excluded by outbound ACLs.

Configuration

```
R4:
access-list 199 deny ip 155.1.146.0 0.0.0.255 any
access-list 199 permit ip any any
!
ip wccp web-cache redirect-list 199
ip wccp outbound-acl-check
!
interface FastEthernet 0/0
 ip wccp web-cache redirect out
!
interface Serial 0/1/0
 ip wccp redirect exclude in

R6:
access-list 199 deny ip 155.1.146.0 0.0.0.255 any
access-list 199 permit ip any any
!
ip wccp version 1
ip wccp web-cache redirect-list 199
ip wccp outbound-acl-check
!
interface FastEthernet 0/0.146
 ip wccp web-cache redirect in
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Once a WCCP cache server is configured to register with the router, the router starts redirecting HTTP connections to the cache server. Note that the direction of the redirection indicates which traffic flows are redirected, either inbound or outbound flows, not where the cache engine is located. Verification of this feature is not available without an actual server that supports the WCCP protocol, however the basic WCCP configuration status can be verified as follows.

```
Rack1R4#show ip wccp
Global WCCP information:
    Router information:
      Router Identifier:                   150.1.4.4
      Protocol Version:                    1.0

    Service Identifier: web-cache
      Number of Service Group Clients:     0
      Number of Service Group Routers:     1
      Total Packets s/w Redirected:        0
        Process:                           0
        Fast:                              0
        CEF:                               0
      Redirect access-list:                199
      Total Packets Denied Redirect:       0
      Total Packets Unassigned:            0
      Group access-list:                   -none-
      Total Messages Denied to Group:      0
      Total Authentication failures:       0

Rack1R4#show ip wccp interface
WCCP interface configuration:
    Ethernet0/0
      Output services: 1
      Input services:  0
      Mcast services:  0
      Exclude In:      FALSE

    Serial0/1/0
      Output services: 0
      Input services:  0
      Mcast services:  0
      Exclude In:      TRUE
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip wccp
Global WCCP information:
    Router information:
      Router Identifier:                   150.1.6.6
      Protocol Version:                    1.0

    Service Identifier: web-cache
      Number of Service Group Clients:     0
      Number of Service Group Routers:     1
      Total Packets s/w Redirected:        0
        Process:                           0
        Fast:                              0
        CEF:                               0
      Redirect access-list:                199
      Total Packets Denied Redirect:       0
      Total Packets Unassigned:            0
      Group access-list:                   -none-
      Total Messages Denied to Group:      0
      Total Authentication failures:       0

Rack1R6#show ip wccp  interface
WCCP interface configuration:
    FastEthernet0/0.146
      Output services: 0
      Input services:  1
      Mcast services:  0
      Exclude In:      FALSE
```

Copyright © 2010 Internetwork Expert

13.42 WCCPv2 Services

- Configure R5 to listen for WCCPv2 cache engines on its FastEthernet0/0 link using a dynamic service-group numbered 50.
- The cache engines will be sending packets to the multicast address 224.0.1.100.
- Authenticate them with the password CISCO.
- Only accept packets from the cache engines 155.X.58.100 and 155.X.58.200.
- Redirect traffic from hosts in the VLAN 58 subnet to these cache engines.

Configuration

```
R5:
access-list 10 permit 155.1.58.200
access-list 10 permit 155.1.58.100
!
access-list 20 permit 155.1.58.0 0.0.0.255
!
ip wccp version 2
ip wccp 50 group-address 224.0.1.100 redirect-list 20 group-list 10
password CISCO
!
interface FastEthernet 0/0
 ip wccp 50 redirect in
 ip wccp 50 group-listen
```

Verification

Note

WCCPv2 allows the cache engine to support “dynamic service-groups” with the router. During the registration process the router is told which protocols and ports it should redirect to the cache engines. The router must be configured for the service-group ID before the cache engines register the service-group.

Additionally, WCCPv2 enhancements allow many routers to communicate with a cache cluster, and use multicast packets for more effective signaling. WCCPv2 also allows authenticating communications using an MD5 hashing scheme.

To verify the configuration check the WCCPv2 service group 50 status.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip wccp 50
Global WCCP information:
    Router information:
      Router Identifier:                   -not yet determined-
      Protocol Version:                    2.0

    Service Identifier: 50
      Number of Service Group Clients:     0
      Number of Service Group Routers:     0
      Total Packets s/w Redirected:        0
        Process:                           0
        Fast:                              0
        CEF:                               0
      Redirect access-list:                20
      Total Packets Denied Redirect:       0
      Total Packets Unassigned:            0
      Group access-list:                   10
      Total Messages Denied to Group:      0
      Total Authentication failures:       0
      Total Bypassed Packets Received:     0

Rack1R5#show ip wccp interfaces
WCCP interface configuration:
    FastEthernet0/0
      Output services: 0
      Input services:  1
      Mcast services:  1
      Exclude In:      FALSE
```

Copyright © 2010 Internetwork Expert

> **Note**
>
> Revert all devices to the *IP Services* initial configurations prior to continuing.

13.43 NBAR Protocol Discovery

- Configure R4 and R6 to collect protocol statistics on their connections to backbone routers using NBAR.
- Both routers should be able to classify connections to HTTP proxy ports 3128 and 8080.
- Change the SOCKS protocol mapping to port 2080.
- Create a new custom protocol mapping called TEST that matches the ASCII character "A" in the beginning of a TCP segment flowing to the destination port 3001.

Configuration

```
R4:
interface FastEthernet 0/0
 ip nbar protocol-discovery
!
ip nbar custom HTTP_PROXY destination tcp 3128 8080
ip nbar port-map socks tcp 2080
ip nbar custom TEST 0 ascii A destination tcp 3001

R6:
interface Serial 0/0/0
 ip nbar protocol-discovery
!
ip nbar custom HTTP_PROXY destination tcp 3128 8080
ip nbar port-map socks tcp 2080
ip nbar custom TEST 0 ascii A destination tcp 3001
```

Verification

> **Note**
>
> NBAR protocol discovery is used for network traffic monitoring. It displays the protocols flowing across a particular interface along with their statistics. As shown in this example you can also define new protocols using port-mapping or a low-level byte string match.

Copyright © 2010 Internetwork Expert

Rack1R4#show ip nbar protocol-discovery

```
 FastEthernet0/0
                             Input                     Output
                             -----                     ------
   Protocol                  Packet Count              Packet Count
                             Byte Count                Byte Count
                             5min Bit Rate (bps)       5min Bit Rate (bps)
                             5min Max Bit Rate (bps)   5min Max Bit Rate (bps)
   ------------------------- ------------------------- -------------------------
   rip                       52                        154
                             10712                     60984
                             0                         0
                             0                         0
   bgp                       10                        5
                             600                       300
                             0                         0
                             0                         0
   HTTP_PROXY                0                         0
<snip>
```

Rack1R1#ping 30.0.0.1

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/5/8 ms
Rack1R1#
```

Rack1R4#show ip nbar protocol-discovery

```
 FastEthernet0/0
                             Input                     Output
                             -----                     ------
   Protocol                  Packet Count              Packet Count
                             Byte Count                Byte Count
                             5min Bit Rate (bps)       5min Bit Rate (bps)
                             5min Max Bit Rate (bps)   5min Max Bit Rate (bps)
   ------------------------- ------------------------- -------------------------
   rip                       98                        292
                             20188                     115632
                             0                         0
                             0                         0
   bgp                       22                        11
                             1320                      660
                             0                         0
                             0                         0
   icmp                      5                         5
                             570                       570
                             0                         0
                             0                         0
   HTTP_PROXY                0                         0
                             0                         0
                             0                         0
                             0                         0
```

Copyright © 2010 Internetwork Expert

To verify a byte-string match on R4, configure R1 to listen on port 3001 and initiate a TCP connection to it.

```
Rack1R1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R1(config)#line vty 0
Rack1R1(config-line)#rotary 1

BB3>telnet 150.1.1.1 3001
Trying 150.1.1.1, 3001 ... Open

User Access Verification

Password: cisco
Rack1R1>AAAAA
Translating "AAAAA"

Translating "AAAAA"
% Unknown command or computer name, or unable to find computer address

Rack1R4#show ip nbar protocol-discovery protocol TEST

 FastEthernet0/0
                         Input                     Output
                         -----                     ------
   Protocol              Packet Count              Packet Count
                         Byte Count                Byte Count
                         5min Bit Rate (bps)       5min Bit Rate (bps)
                         5min Max Bit Rate (bps)   5min Max Bit Rate (bps)
   ----------------------- ------------------------ ------------------------
   TEST                  9                         7
                         540                       509
                         0                         0
                         0                         0
```

The port values that NBAR uses for matching can be listed as follows.

```
Rack1R1#show ip nbar port-map
port-map HTTP_PROXY               tcp 3128 8080
port-map TEST                     tcp 3001
port-map bgp                      udp 179
port-map bgp                      tcp 179
port-map citrix                   udp 1604
port-map citrix                   tcp 1494
port-map cuseeme                  udp 7648 7649 24032
port-map cuseeme                  tcp 7648 7649
port-map dhcp                     udp 67 68
port-map dns                      udp 53
port-map dns                      tcp 53
port-map edonkey                  tcp 4662
port-map exchange                 tcp 135
port-map fasttrack                tcp 1214
port-map finger                   tcp 79
<snip>
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.44 Netflow Ingress & Egress

- Configure R4 and R6 to export Netflow version 5 information to a host 155.X.146.100 in VLAN 146 using port 9999.
- R4 should collect statistics on incoming flows on all interfaces, while R6 should collect statistics on outgoing flows on all interfaces.
- Limit the size of the flow cache to 4096 entries and export the BGP AS of Origin if available.
- Collect additional information about ICMP message information as well as layer 2 VLAN numbers where applicable.

Configuration

```
R4:
ip flow-capture vlan-id
ip flow-capture icmp
ip flow-export version 5
ip flow-export destination 155.1.146.100 9999
ip flow-export version 5 origin-as
ip flow-cache entries 4096
!
interface Serial 0/0/0
 ip flow ingress
!
interface Serial 0/1/0
 ip flow ingress
!
interface FastEthernet 0/0
 ip flow ingress
!
interface FastEthernet 0/1
 ip flow ingress

R6:
ip flow-capture vlan-id
ip flow-capture icmp
ip flow-export version 5
ip flow-export version 5 origin-as
ip flow-export destination 155.1.146.100 9999
ip flow-cache entries 4096
!
interface Serial 0/0/0
 ip flow egress
!
interface FastEthernet 0/0.67
 ip flow egress
!
interface FastEthernet 0/0.146
 ip flow egress
```

Copyright © 2010 Internetwork Expert

Verification

Note

Netfow evolved from a route-caching (switching path) technology to a powerful statistics-gathering tool. Flow collection on interfaces can be enabled for either inbound (ingress) or outbound (egress) flow. The difference between them is that ingress flow monitoring will capture all flows entering the router, including the packets destined to the router itself. Egress flow collects packets switched across the router that are leaving via an egress interface, but does not include packets originated from the router itself.

Egress flow collection has some performance penalties, but is useful when you need to collect Netflow information on MPLS traffic, which leaves the router untagged. This is needed since ingress Netflow can't look inside the MPLS tagged packets. You may distinguish egress flows in the show command output by recognizing a "*" sign near the destination interface. Another difference between ingress and egress Netflow is their interaction with other IOS features, such as NAT, rate-limiting, and encryption. Ingress Netflow is applied before ACLs and rate-limiting, while egress Netflow is applied after rate-limiting and encryption features.

Different versions of Netflow are available, with the most basic, widely available, and default version being 5. Version 8 supports aggregation caches, and version 9, which is incompatible with previous versions, allows for the most flexibility with choosing the export template format. Aggregation caches are also supported by version 9, so version 8 is becoming deprecated.

Copyright © 2010 Internetwork Expert

To verify this configuration, generate some flows and check verbose flow-cache output as follows.

```
Rack1R1#telnet 212.18.1.1
Trying 212.18.1.1 ... Open

<snip>

Rack1R6#show ip cache verbose flow
IP packet size distribution (30 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
   .000 .566 .000 .333 .033 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .000 .000 .066 .000 .000 .000 .000 .000 .000 .000 .000

IP Flow Switching Cache, 278544 bytes
  2 active, 4094 inactive, 4 added
  50 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes
  Inactive flows timeout in 15 seconds
IP Sub Flow Cache, 21640 bytes
  4 active, 1020 inactive, 8 added, 4 added to flow
  0 alloc failures, 0 force free
  1 chunk, 1 chunk added
  last clearing of statistics never
Protocol         Total    Flows   Packets Bytes  Packets Active(Sec) Idle(Sec)
--------         Flows     /Sec     /Flow  /Pkt     /Sec     /Flow     /Flow
ICMP                 2      0.0         5   100      0.0       0.1      15.9
Total:               2      0.0         5   100      0.0       0.1      15.9

SrcIf          SrcIPaddress    DstIf          DstIPaddress    Pr TOS Flgs  Pkts
Port Msk AS                    Port Msk AS    NextHop              B/Pk  Active
Fa0/0.146      155.1.146.1     Se0/0*         212.18.1.1      06 C0  1A      10
2AF8 /24 0                     0017 /24 0     54.1.1.254             43     0.4
FFlags: 01
VLAN id:       146                            000
Min plen:       40                            Max plen:       52

Se0/0          212.18.1.1      Fa0/0.146*     155.1.146.1     06 C0  1A      10
0017 /24 0                     2AF8 /24 0     155.1.146.1           160     0.6
FFlags: 01
VLAN id:       000                            146
Min plen:       40                            Max plen:      576
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#ping 30.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/5/8 ms

Rack1R4#show ip cache verbose flow
IP packet size distribution (190 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
   .000 .015 .000 .105 .000 .052 .000 .000 .168 .000 .000 .000 .494 .000 .000

    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .000 .163 .000 .000 .000 .000 .000 .000 .000 .000 .000

IP Flow Switching Cache, 278544 bytes
  6 active, 4090 inactive, 21 added
  1389 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes
  Inactive flows timeout in 15 seconds
IP Sub Flow Cache, 21640 bytes
  12 active, 1012 inactive, 42 added, 21 added to flow
  0 alloc failures, 0 force free
  1 chunk, 1 chunk added
  last clearing of statistics never
Protocol         Total    Flows   Packets Bytes  Packets Active(Sec) Idle(Sec)
--------         Flows     /Sec     /Flow  /Pkt     /Sec     /Flow     /Flow
TCP-BGP              3      0.0         1    44      0.0       0.0      15.7
UDP-other           10      0.0         1   192      0.0       0.0      15.5
ICMP                 2      0.0         5   100      0.0       0.0      15.5
Total:              15      0.0         1   132      0.0       0.0      15.5

SrcIf          SrcIPaddress    DstIf          DstIPaddress    Pr TOS Flgs  Pkts
Port Msk AS                    Port Msk AS    NextHop              B/Pk  Active
Et0/1          155.1.146.1     Et0/0          30.0.0.1        01 00  10       5
0000 /24 0                     0800 /16 0     204.12.1.254           100    0.0
VLAN id:       000                            000
Min plen:      100                            Max plen:       100
ICMP type:       8                            ICMP code:        0

Se0/1/0        155.1.45.5      Null           224.0.0.9       11 C0  10      32
0208 /24 0                     0208 /24 0     0.0.0.0                412   284.7
VLAN id:       000                            000
Min plen:      412                            Max plen:       412

<snip>

Et0/0          30.0.0.1        Et0/1          155.1.146.1     01 00  10       5
0000 /16 0                     0000 /24 0     155.1.146.1            100    0.0
VLAN id:       000                            000
Min plen:      100                            Max plen:       100
ICMP type:       0                            ICMP code:        0
```

Copyright © 2010 Internetwork Expert

13.45 Netflow Top Talkers

- Configure R4 and R6 to collect information about 10 most active ICMP flows.
- R4 should count packets, while R6 counts bytes.
- Match only TCP and ICMP flows coming from the subnet 155.X.0.0/16.

Configuration

```
R4:
ip flow-top-talkers
 top 10
 sort-by packets
 match source address 155.1.0.0 255.255.0.0
 match protocol 1

R6:
ip flow-top-talkers
 top 10
 sort-by bytes
 match source address 155.1.0.0 255.255.0.0
 match protocol 1
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> The top talkers feature collects information about the most packet or traffic-intense flows. The information is displayed only about the active, non-expired flows, so you may miss some short and bursty transmission. A flexible filtering system allows customizing the view per your requirements. The feature can be verified as follows.

```
Rack1R1#ping 212.18.1.1 repeat 50

Type escape sequence to abort.
Sending 50, 100-byte ICMP Echos to 212.18.1.1, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (50/50), round-trip min/avg/max = 32/35/44 ms

Rack1R1#ping 30.0.0.1 repeat 50

Type escape sequence to abort.
Sending 50, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (50/50), round-trip min/avg/max = 4/5/8 ms

Rack1R4#show ip flow top-talkers

SrcIf         SrcIPaddress    DstIf         DstIPaddress    Pr SrcP DstP  Pkts
Et0/1         155.1.146.1     Et0/0         30.0.0.1        01 0000 0800    50
1 of 10 top talkers shown. 1 of 7 flows matched.

Rack1R6#show ip flow top-talkers

SrcIf         SrcIPaddress    DstIf         DstIPaddress    Pr SrcP DstP Bytes
Fa0/0.146     155.1.146.1     Se0/0*        212.18.1.1      01 0000 0800  5000
1 of 10 top talkers shown. 1 of 2 flows matched.
```

Copyright © 2010 Internetwork Expert

13.46 Netflow Aggregation Cache

- Create an aggregation cache that is destination-prefix based on R4 and source-based on R6.
- Export the new information to 155.X.146.100 at port 9998 with Netflow version 8.
- Do not summarize addresses to more than a /8 prefix length.
- Store no more than 1024 aggregated entries.

Configuration

```
R4:
ip flow-aggregation cache destination-prefix
 cache entries 1024
 export destination 155.1.146.100 9998
 mask destination minimum 8
 enabled

R6:
ip flow-aggregation cache source-prefix
 cache entries 1024
 export destination 155.1.146.100 9998
 mask source minimum 8
 enabled
```

Verification

Note

The Netflow aggregation cache is an additional flow cache used to store aggregated prefix information. A special Netflow packet format that was first supported in version 8, and more recently in version 9, is used to export the aggregated information to a collector.

The aggregation cache is based on the main (normal) flow cache, and flow aggregation is based on the routing table contents. For example when a flow is exported from the main cache to the aggregation cache, the flow's source and/or destination IP addresses (based on the configured scheme) are logically ANDed bitwise with the network masks of its corresponding route in the routing table. The resulting "masked" (less-specific) flow entry is merged with the aggregation cache. This will result in multiple flow entries with source or destination IP addresses belonging to the same prefix in IP routing table looking the same in aggregation cache, with the bytes and/or packet count being the sum of all specific flows. Additionally the number of merged flows for each aggregated entry is also stored in aggregation cache.

Copyright © 2010 Internetwork Expert

An additional feature, called the minimum aggregation mask, allows preserving a specified level of granularity even when widely scoped summary routes are installed in the routing table. For example if the router has the prefix 10.0.0.0/8 in its routing table, you may still want to know detailed flow information information about the subnets 10.X.X.0. By setting the minimum aggregation mask to /24 the aggregation cache will choose the longer of the two masks when creating an entry. If a flow source with the IP address 10.1.2.3 is about to be aggregated, the routing table mask of /8 is compared against the minimum aggregation mask of /24, and the result is that the router installs the aggregate entry as 10.1.2.0.

```
Rack1R1#ping 30.2.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 30.2.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/6/8 ms

Rack1R1#ping 30.2.0.254 time 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 30.2.0.254, timeout is 0 seconds:
 ...
Success rate is 0 percent (0/5)

Rack1R1#ping 30.1.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 30.1.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/5/8 ms

Rack1R1#ping 30.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/5/8 ms
```

In the above case is to be summarized based on destination addresses. On R1 all prefixes in routing table for the subnets of 30.0.0.0 have prefix length of 16. Since max(minimum-mask, routing-prefix) = max(8,16) = 16, the prefix length in the aggregation table is /16.

Note that prefix 30.2.0.0/16 has twice as many packets as the other due to the two pings to destinations in that range.

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ip route 30.0.0.0
Routing entry for 30.0.0.0/16, 4 known subnets
  Redistributing via rip

R       30.2.0.0 [120/1] via 204.12.1.254, 00:00:23, Ethernet0/0
R       30.3.0.0 [120/1] via 204.12.1.254, 00:00:23, Ethernet0/0
R       30.0.0.0 [120/1] via 204.12.1.254, 00:00:23, Ethernet0/0
R       30.1.0.0 [120/1] via 204.12.1.254, 00:00:23, Ethernet0/0
```

```
Rack1R4#show ip cache flow aggregation destination-prefix

IP Flow Switching Cache, 69636 bytes
  5 active, 1019 inactive, 150 added
  2336 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes
  Inactive flows timeout in 15 seconds
IP Sub Flow Cache, 5896 bytes
  5 active, 251 inactive, 150 added, 150 added to flow
  0 alloc failures, 0 force free
  1 chunk, 1 chunk added

Minimum destination mask is configured to /8

Dst If         Dst Prefix      Msk  AS     Flows   Pkts B/Pk  Active
Null           224.0.0.0       /24  0          1      1   192     0.0
Fa0/1          155.1.146.0     /24  0          3     15   100    14.7
Fa0/0          30.2.0.0        /16  0          2     10   100     8.2
Fa0/0          30.0.0.0        /16  0          1      5   100     0.0
Fa0/0          30.1.0.0        /16  0          1      5   100     0.0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

13.47 Netflow Random Sampling

- Modify R6's Netflow configuration so that it only samples every 10th packet in each flow.

Configuration

```
R6:
flow-sampler-map SAMPLER
 mode random one-out-of 10
!
policy-map NETFLOW_MAP
 class class-default
   netflow-sampler SAMPLER
!
interface Serial 0/0/0
 no ip flow egress
 service-policy output NETFLOW_MAP
!
interface FastEthernet 0/0.146
 no ip flow egress
 service-policy output NETFLOW_MAP
!
interface FastEthernet 0/0.67
 no ip flow egress
 service-policy output NETFLOW_MAP
```

Verification

Note

Flow sampling allows you to lower the amount of Netflow information collected, which lowers the performance impact on the router collecting statistics. Commonly random sampling is used for the purpose of capacity planning, where precise numbers are not required, but statistical information is required. The random flow sampler randomly picks every 1 of Nth packets for each flow and uses it to build flow information. When a flow sampler is set to sample 1 of 1, it essentially behaves in normal flow-collecting mode.

The application of the flow samplers is through an MQC policy-map. An outgoing service policy will implement egress Netflow, and an incoming service policy implements ingress Netflow. The interface-level netflow configuration will override the MQC settings, so always remove the **`ip flow ingress`** or **`ip flow egress`** commands when applying Netflow samplers.

When a sampler is applied in the class-default of a policy-map it effectively applies to all traffic entering or leaving the respective interface, provided that traffic does not match any other classes in the policy-map.

Copyright © 2010 Internetwork Expert

```
Rack1R1#ping 212.18.2.1 repeat 500

Type escape sequence to abort.
Sending 500, 100-byte ICMP Echos to 212.18.2.1, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!
<snip>

Rack1R6#show ip cache flow
IP packet size distribution (1631 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
   .000 .068 .000 .925 .001 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .000 .000 .003 .000 .000 .000 .000 .000 .000 .000 .000

IP Flow Switching Cache, 278544 bytes
  2 active, 4094 inactive, 28 added
  452 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes
  Inactive flows timeout in 15 seconds
IP Sub Flow Cache, 21640 bytes
  4 active, 1020 inactive, 56 added, 28 added to flow
  0 alloc failures, 0 force free
  1 chunk, 1 chunk added
  last clearing of statistics never
Protocol         Total    Flows   Packets Bytes  Packets Active(Sec) Idle(Sec)
--------         Flows     /Sec     /Flow  /Pkt     /Sec     /Flow     /Flow
TCP-Telnet          14      0.0         8    72      0.0       1.4       9.6
ICMP                12      0.0       117   100      0.0       4.5      15.4
Total:              26      0.0        58    97      0.0       2.8      12.3

SrcIf         SrcIPaddress    DstIf         DstIPaddress    Pr SrcP DstP  Pkts
Fa0/0.146     155.1.146.1     Se0/0*        212.18.2.1      01 0000 0800    57
Se0/0/0       212.18.2.1      Fa0/0.146*    155.1.146.1     01 0000 0000    43
```

Note that the number of packets accounted is approximately 10 times less than the number of packets actually sent and received. As the number of packets grows, the relation will tend to be more precise, so for the purpose of statistical analysis it is possible to simply scale the numeric values collected by 10 to get realistic results.

Copyright © 2010 Internetwork Expert

```
Rack1R6#show policy-map interface serial 0/0/0

      netflow-sampler: SAMPLER
 Serial0/0/0

  Service-policy output: NETFLOW_MAP

    Class-map: class-default (match-any)
      1503 packets, 287842 bytes
      5 minute offered rate 2000 bps, drop rate 0 bps
      Match: any

Rack1R6#show policy-map int fastEthernet 0/0.146
      netflow-sampler: SAMPLER
 FastEthernet0/0.146

  Service-policy output: NETFLOW_MAP

    Class-map: class-default (match-any)
      1066 packets, 136236 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Copyright © 2010 Internetwork Expert

13.48 Netflow Input Filters

- Modify R6's Netflow sampling so that every packet is sampled for sources on VLAN 146.
- Other packets should still be randomly sampled.

Configuration

```
R6:
ip access-list extended VLAN146
 permit ip 155.1.146.0 0.0.0.255 any
 permit ip any 155.1.146.0 0.0.0.255
!
class-map VLAN146
 match access-group name VLAN146
!
flow-sampler-map NORMAL
 mode random one-out-of 1
!
policy-map NETFLOW_MAP
 class VLAN146
   netflow-sampler NORMAL
```

Verification

> **Note**
>
> Netflow input filters allow you to attach a flow sampler to any class in a policy-map. In this example class VLAN146 samples 1/1 packets, while class-default (everything else) still samples 1/10 packets.

```
Rack1R1#ping 212.18.2.1 repeat 500

Type escape sequence to abort.
Sending 500, 100-byte ICMP Echos to 212.18.2.1, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!
Success rate is 100 percent (500/500), round-trip min/avg/max =
32/37/408 ms
```

```
Rack1R6#show ip cache flow
IP packet size distribution (2631 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
   .000 .042 .000 .954 .001 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .000 .000 .002 .000 .000 .000 .000 .000 .000 .000 .000

IP Flow Switching Cache, 278544 bytes
  2 active, 4094 inactive, 30 added
  528 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes
  Inactive flows timeout in 15 seconds
IP Sub Flow Cache, 21640 bytes
  4 active, 1020 inactive, 60 added, 30 added to flow
  0 alloc failures, 0 force free
  1 chunk, 1 chunk added
  last clearing of statistics never
Protocol         Total    Flows   Packets Bytes  Packets Active(Sec) Idle(Sec)
--------         Flows     /Sec     /Flow  /Pkt     /Sec     /Flow     /Flow
TCP-Telnet          14      0.0         8    72      0.0       1.4       9.6
ICMP                14      0.0       107   100      0.0       6.5      15.4
Total:              28      0.0        58    97      0.0       3.9      12.5

SrcIf         SrcIPaddress    DstIf         DstIPaddress    Pr SrcP DstP  Pkts
Se0/0/0       212.18.2.1      Fa0/0.146*    155.1.146.1     01 0000 0000   500
Fa0/0.146     155.1.146.1     Se0/0*        212.18.2.1      01 0000 0800   500
```

Generate traffic from a subnet that is classified by class-default in the policy-map. Note that this traffic is statistically sampled.

```
Rack1R1#ping 212.18.2.1 repeat 500 source loopback 0

Type escape sequence to abort.
Sending 500, 100-byte ICMP Echos to 212.18.2.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.1.1
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!
Success rate is 100 percent (500/500), round-trip min/avg/max =
32/37/400 ms
Rack1R1#
```

```
Rack1R6#show ip cache flow
IP packet size distribution (2731 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
   .000 .041 .000 .955 .001 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .000 .000 .002 .000 .000 .000 .000 .000 .000 .000 .000

IP Flow Switching Cache, 278544 bytes
  2 active, 4094 inactive, 32 added
  594 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes
  Inactive flows timeout in 15 seconds
IP Sub Flow Cache, 21640 bytes
  4 active, 1020 inactive, 64 added, 32 added to flow
  0 alloc failures, 0 force free
  1 chunk, 1 chunk added
  last clearing of statistics never
Protocol         Total    Flows   Packets Bytes  Packets Active(Sec) Idle(Sec)
--------         Flows     /Sec     /Flow  /Pkt     /Sec     /Flow     /Flow
TCP-Telnet          14      0.0         8    72      0.0       1.4       9.6
ICMP                16      0.0       156   100      0.0       8.1      15.4
Total:              30      0.0        87    98      0.0       4.9      12.7

SrcIf         SrcIPaddress    DstIf         DstIPaddress    Pr SrcP DstP  Pkts
Fa0/0.146     150.1.1.1       Se0/0*        212.18.2.1      01 0000 0800    47
Se0/0/0       212.18.2.1      Fa0/0.146*    150.1.1.1       01 0000 0000    53
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

> **Note**
>
> Revert all devices to the *IP Services* initial configurations prior to continuing.

13.49 IOS Authoritative DNS Server

- Configure R4 and R6 as primary DNS servers for the domain cisco.com.
- The primary name server should be called ns.cisco.com with a contact mailbox of ccie.cisco.com.
- Both R4 and R6 should resolve the names R4.cisco.com and R6.cisco.com to all connected IP addresses of the respective router.
- R1 should use both R4 and R6 as its DNS servers in a round-robin fashion.
- R1's DNS client should complete all unqualified domain-names with the name "cisco.com".

Configuration

```
R1:
ip name-server 155.1.146.4 155.1.146.6
ip domain-lookup
ip domain name cisco.com
ip domain round-robin

R4:
ip dns server
ip dns primary cisco.com soa ns.cisco.com ccie.cisco.com
!
ip host cisco.com ns 155.1.146.4
ip host cisco.com ns 155.1.146.6
!
ip host R4.cisco.com 150.1.4.4 155.1.146.4 155.1.45.4 204.12.1.4
155.1.0.4
ip host R6.cisco.com 150.1.6.6 155.1.146.6 155.1.67.6 54.1.1.6

R6:
ip dns server
!
ip dns server
ip dns primary cisco.com soa ns.cisco.com ccie.cisco.com
!
ip host cisco.com ns 155.1.146.4
ip host cisco.com ns 155.1.146.6
!
ip host R4.cisco.com 150.1.4.4 155.1.146.4 155.1.45.4 204.12.1.4
155.1.0.4
ip host R6.cisco.com 150.1.6.6 155.1.146.6 155.1.67.6 54.1.1.6
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> IOS supports basic DNS server resolution, in addition to SOA, MX, NS, SRV resource records to act as an authoritative DNS server.
>
> The IOS DNS client ican load-balance between multiple A entries returned in a DNS response in a round-robin fashion. You can specify the default domain name to complete "unqualified" names (such as "R4") to fully-qualified DNS names ("R4.cisco.com").

```
Rack1R4#show ip dns primary
Primary for zone cisco.com:
  SOA information:
  Zone primary (MNAME): ns.cisco.com
  Zone contact (RNAME): ccie.cisco.com
  Refresh (seconds):    21600
  Retry (seconds):      900
  Expire (seconds):     7776000
  Minimum (seconds):    86400

Rack1R6#show ip dns primary
Primary for zone cisco.com:
  SOA information:
  Zone primary (MNAME): ns.cisco.com
  Zone contact (RNAME): ccie.cisco.com
  Refresh (seconds):    21600
  Retry (seconds):      900
  Expire (seconds):     7776000
  Minimum (seconds):    86400

Rack1R1#ping R4
Translating "R4"...domain server (155.1.146.4) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.45.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 100/100/100 ms

Rack1R1#ping R4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 204.12.1.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 100/100/100 ms
```

Copyright © 2010 Internetwork Expert

```
Rack1R1#ping R4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.0.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/116/124 ms

Rack1R1#show hosts
Default domain is cisco.com
Name/address lookup uses domain service
Name servers are 155.1.146.4, 155.1.146.6

Codes: UN - unknown, EX - expired, OK - OK, ?? - revalidate
       temp - temporary, perm - permanent
       NA - Not Applicable None - Not defined

Host                      Port  Flags      Age Type   Address(es)
R4.cisco.com              None  (temp, OK)  0   IP    155.1.0.4
                                                      150.1.4.4
                                                      155.1.146.4
                                                      155.1.45.4
                                                      204.12.1.4
```

```
Rack1R1#ping R6
Translating "R6"...domain server (155.1.146.4) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.67.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 100/100/104 ms

Rack1R1#ping R6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 54.1.1.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 100/100/104 ms

Rack1R1#ping R6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.6.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 100/100/104 ms

Rack1R1#ping R6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.146.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1R1#show hosts
Default domain is cisco.com
Name/address lookup uses domain service
Name servers are 155.1.146.4, 155.1.146.6

Codes: UN - unknown, EX - expired, OK - OK, ?? - revalidate
       temp - temporary, perm - permanent
       NA - Not Applicable None - Not defined

Host                      Port  Flags      Age Type   Address(es)
R6.cisco.com              None  (temp, OK)  0   IP    155.1.146.6
                                                      155.1.67.6
                                                      54.1.1.6
                                                      150.1.6.6
```

13.50 IOS Caching DNS Server

- Configure R3 as a DNS client of R1.
- When R1 receives a DNS query from R3 it should check its local cache.
- If the entry is not cached a request should be sent from R1 to R4 or R6 on behalf of R3.

Configuration

```
R1:
ip name-server 155.1.146.4 155.1.146.6
ip domain-lookup
ip dns server

R3:
ip name-server 150.1.1.1
ip domain-lookup
```

Verification

> **Note**
>
> To verify this configuration issue ping commands from R3 using R4 and R6's fully-qualified domain-names.

```
Rack1R3#ping R4.cisco.com.

Translating "R4.cisco.com."...domain server (150.1.1.1) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 128/128/132 ms

Rack1R3#ping R6.cisco.com

Translating "R6.cisco.com"...domain server (150.1.1.1) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.6.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 144/144/144 ms
```

Copyright © 2010 Internetwork Expert

Debug DNS server activities on R1 and send a DNS query from R3.

```
Rack1R1#debug domain
Domain Name System debugging is on

Rack1R3#ping R6.cisco.com

Translating "R6.cisco.com"...domain server (150.1.1.1) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.6.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 144/152/188 ms

Rack1R1#
DNS: Incoming UDP query (id#43)
DNS: Type 1 DNS query (id#43) for host 'R6.cisco.com' from
155.1.13.3(55196)
DNS: Re-sending DNS query (type 1, id#43) to 155.1.146.4
DNS: Incoming UDP query (id#43)
DNS: Type 1 response (id#43) for host <R6.cisco.com> from
155.1.146.4(53)
DOM: dom2cache: hostname is R6.cisco.com, RR type=1, class=1, ttl=10,
n=4
DOM: dom2cache: hostname is R6.cisco.com, RR type=1, class=1, ttl=10,
n=4
DOM: dom2cache: hostname is R6.cisco.com, RR type=1, class=1, ttl=10,
n=4
DOM: dom2cache: hostname is R6.cisco.com, RR type=1, class=1, ttl=10,
n=4
DNS: Forwarding back A response - no director required
DNS: Finished processing query (id#43) in 0.012 secs
DNS: Forwarding back reply to 155.1.13.3/55196
```

Copyright © 2010 Internetwork Expert

13.51 IOS DNS Spoofing

- Configure R5 as a DNS client of R4.
- Configure SW2 as a DNS client of R5.
- If R5 loses connectivity to R4 it should respond to all DNS queries with the IP address of its Loopback0 interface.

Configuration

```
R5:
ip dns spoofing 150.1.5.5
ip name-server 155.1.4.4
ip domain lookup
ip dns server

SW2:
ip domain lookup
ip name-server 155.1.58.5
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> A router acting as a DNS caching server with just one upstream interface will forward DNS requests to its upstream server as long as the connection is healthy. When the router loses reachability to the DNS servers, it will respond to all DNS queries with a pre-configured IP address, i.e. with the IP address of an internal HTTP server to display an informational message.

```
Rack1SW2#ping R4.cisco.com.

Translating "R4.cisco.com."...domain server (155.1.58.5) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 42/45/51 ms

Rack1SW2#ping TEST

Translating "TEST"...domain server (155.1.58.5)
% Unrecognized host or address, or protocol not running.
```

Copyright © 2010 Internetwork Expert

Shutdown R5's connections to make sure it can't reach the Loopback 0 subnet of R4 and enable DNS debugging on R5.

```
Rack1R5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface Serial 0/0/0
Rack1R5(config-if)#shutdown
Rack1R5(config-if)#interface Serial 0/1/0
Rack1R5(config-if)#shutdown

Rack1R5#debug domain
```

Ping any name from SW2 to see that it actually resolves to the IP address of R5's Loopback 0 interface.

```
Rack1SW2#ping TEST

Translating "TEST"...domain server (155.1.58.5) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.5.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/8 ms

Rack1SW2#ping R4.cisco.com.

Translating "R4.cisco.com."...domain server (155.1.58.5) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.5.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms

Rack1SW2#ping ANYNAME

Translating "ANYNAME"...domain server (155.1.58.5) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.5.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/4/9 ms
```

Note that queries for the hostname of R5 return the IP address of R5 interface that received the query.

Copyright © 2010 Internetwork Expert

```
Rack1SW2#ping Rack1R5

Translating "Rack1R5"...domain server (155.1.58.5) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.58.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/9 ms
```

Observe the debugging output at R5. The router responds with the configured spoofing address to any query with the exception of queries for its own hostname. For its own name the router returns the IP address of the interface that received the query.

```
Rack1R5#
DNS: Incoming UDP query (id#7)
DNS: Type 1 DNS query (id#7) for host 'ANYNAME' from 155.1.58.8(52461)
DNS: No name-servers are accessible
DNS: Spoofing reply to query (id#7)
DNS: Finished processing query (id#7) in 0.000 secs
DNS: Incoming UDP query (id#8)
DNS: Type 1 DNS query (id#8) for host 'Rack1R5' from 155.1.58.8(53293)
DNS: Query for my own hostname: Rack1R5
DNS: Spoofing reply to query (id#8)
DNS: Finished processing query (id#8) in 0.004 secs
```

Copyright © 2010 Internetwork Expert

13.52 IP Event Dampening

- Configure dampening on R6 so that after a reload its connection to BB1 is not advertised into IGP for 30 seconds.
- Additionally configure R6 so that when this connection flaps it does not disappear for more that 60 seconds from the routing table no matter how much penalty it accumulates.

Configuration

```
R6:
interface Serial 0/0/0
 dampening 30 1000 2000 60 restart 2000
```

Copyright © 2010 Internetwork Expert

Verification

Note

To find the default IP event dampening values configure the **dampening** command with no arguments at the interface level, and then issue the **show interface dampening** command.

Next, to suppress the interface after the router reloads, apply a restart penalty value. With a half-life time of 30 seconds and restart penalty of 2000, it means that 30 seconds after bootup the penalty will have decayed to 1000. If the reuse value is set to 1000 the end result is that the interface is installed in the routing table 30 seconds after bootup is complete.

```
Rack1R6#show dampening interface
1 interface is configured with dampening.
No interface is being suppressed.
Features that are using interface dampening:
  IP Routing

Rack1R6#show interfaces dampening
Serial0/0/0
  Flaps Penalty    Supp ReuseTm   HalfL  ReuseV   SuppV  MaxSTm    MaxP Restart
      5       0   FALSE       0      30    1000    2000      60    4000    2000
```

Copyright © 2010 Internetwork Expert

MPLS VPN Solutions

Note

Load the *MPLS VPN* initial configurations prior to starting.

14.1 VRF Lite

- Create a new VLAN 76 connecting R6 and SW1 using the subnet 155.X.76.0/24.
- On R6 and SW1 configure VLAN 67 so that it belongs to vrf group VPN_A and VLAN 76 should belong to a different vrf group named VPN_B.
- Create Loopback interfaces Lo 101 and Lo 102 on SW1 and place them in the VRF tables VPN_A and VPN_B respectively, using the IP addresses 172.16.7.7/24 for Lo 101 and 192.168.7.7/24 for Lo 102.
- Both VRFs on SW1 should use R6 as the default gateway.
- Configure R6 so that from SW1 you can ping interface Lo 101 from VPN_B and Lo 102 from VPN_A.

Configuration

Note

VRF (Virtual Routing and Forwarding) tables are the fundamental building blocks needed to turn a single router into multiple virtual routers. VRFs create multiple virtual, isolated networks, where each technically has its own separate RIB (Routing Information Base) and FIB (Forwarding Information Base). These structures are also known as Routing Tables and CEF tables.

Any router interface can be assigned to a VRF using the command **`ip vrf forwarding <VRF_NAME>`**. Notice that this command will erase all existing IP addresses configured on the interface in order to avoid potential address duplication in the new routing table. After this configuration, all packets received on this interface are routed and forwarded using the associated VRF table. This concept is very similar to the way VLAN trunking works at layer two. Packets entering a specific VRF can only follow routes in **that** specific VRF's routing table. Additionally, very much like a layer two trunk can span VLANS across multiple switches, VRFs can be extended across multiple devices as well.

Copyright © 2010 Internetwork Expert

You may extend VRFs beyond a single router by properly mapping the VRFs to the links connecting two routers. This results in "parallel" VPNs being run across multiple devices, also known as "VRF Lite" – the most basic way to build VPNs. This is the simplest way of creating non-overlapping VPNs in a network; however this solution has poor scalability, as you need to allocate a dedicated inter-router link for every VPN. Therefore, if you have two routers, and 100 VPNs, you will need to provision 100 connections between the two routers, one for every VPN. The connection could be either a separate interface, or some Layer 2 virtualization technique, such as Frame-Relay PVC or Ethernet VLAN.

By default, all interfaces (physical or sub-interfaces) are assigned to a VRF known as the **global**, which is the regular routing table used in non-VRF capable routers. In order to create a new VRF, issue the command **ip vrf <VRF_NAME>** which opens up the VRF configuration context mode. After this initial step, you need to define a route distinguisher (RD) for this particular VRF using the command **rd X:Y** where X and Y are 32-bit numbers. A Route Distinguisher is a special 64-bit prefix prepended to every route in the respective VRF routing table. This is done for the purpose of distinguishing the prefixes inside the router and avoiding collisions if two VRFs contain the same prefixes. The common format for a RD is using the combination **ASN:NN** where ASN is the autonomous system number and NN is the VRF number inside the router or more globally, the VPN number within the ASN. Alternatively, you may use the format **IP-Address:NN** where IP is the router's IP address and NN is the VRF name. The second format properly reflects the feature of RD being a local distinguisher, but using the format **ASN:NN** is more popular and common, as it easily associates a VRF with a particular VPN in the network.

It is possible to associate static routes or dynamic routing protocol processes with the VRFs. In this lab, we deal with static routing only. The syntax for a VRF-bound static route is:

```
ip route vrf <NAME> PREFIX MASK [interface] [next-hop]
```

Where **[next-hop]** is an IP address resolvable through the VRF. It is possible to use static routes for "inter-VRF" communications. If you are using a static route with the **[interface]** specification, the interface could belong to any VRF. Note that with multi-access interfaces you will also need to specify the next-hop associated with the interface subnet due to the fact that Cisco IOS will install a CEF entry in the "source" VRF using the information provided and will not attempt to resolve the next-hop recursively. Keep in mind that this trick only works with the non-recursive static routes that use directly connected interfaces.

Copyright © 2010 Internetwork Expert

R6:

```
!
! VRF for VPN_A
!
ip vrf VPN_A
 rd 100:1

!
! VRF for VPN_B
!
ip vrf VPN_B
 rd 100:2

!
! PE-CE Links
!
interface FastEthernet 0/0.67
 ip vrf forwarding VPN_A
 ip address 155.1.67.6 255.255.255.0
!
interface FastEthernet 0/0.76
 encapsulation dot1q 76
 ip vrf forwarding VPN_B
 ip address 155.1.76.6 255.255.255.0

!
! Inter-VRF static routes
!
ip route vrf VPN_A 192.168.7.0 255.255.255.0 FastEthernet0/0.76
155.1.76.7
!
ip route vrf VPN_B 172.16.7.0 255.255.255.0 FastEthernet0/0.67
155.1.67.7
```

SW1:

```
!
! Don't forget the following command
!
ip routing
!
! VRF for VPN_A
!
ip vrf VPN_A
 rd 100:1

!
! VRF for VPN_B
!
ip vrf VPN_B
 rd 100:2
```

Copyright © 2010 Internetwork Expert

```
!
! VPN Interfaces
!
interface Vlan 67
 ip vrf forwarding VPN_A
 ip address 155.1.67.7 255.255.255.0
!
interface Vlan 76
 ip vrf forwarding VPN_B
 ip address 155.1.76.7 255.255.255.0
!
interface Loopback 101
 ip vrf forwarding VPN_A
 ip address 172.16.7.7 255.255.255.0
!
interface Loopback 102
 ip vrf forwarding VPN_B
 ip address 192.168.7.7 255.255.255.0
!
! Default routes for each VRF
!
ip route vrf VPN_A 0.0.0.0 0.0.0.0 155.1.67.6
ip route vrf VPN_B 0.0.0.0 0.0.0.0 155.1.76.6
```

SW1, SW2, SW3, SW4:

```
vlan 76
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Start by checking the VRF interfaces and basic connectivity. Notice that the verification commands now use the "vrf" argument to select the specific routing table.

```
Rack1R6#show ip vrf
  Name                              Default RD          Interfaces
  VPN_A                             100:1               Fa0/0.67
  VPN_B                             100:2               Fa0/0.76

Rack1R6#ping vrf VPN_A 155.1.67.7

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.67.7, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/8 ms

Rack1R6#ping vrf VPN_B 155.1.76.7

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.76.7, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

Now, let's look into inter-VRF connectivity in detail. First, check the CEF table for VRF VPN_A in R6:

```
Rack1R6#show ip route vrf VPN_A 192.168.7.0
Routing entry for 192.168.7.0/24
  Known via "static", distance 1, metric 0
  Routing Descriptor Blocks:
  * 155.1.76.7, via FastEthernet0/0.76
      Route metric is 0, traffic share count is 1

Rack1R6#show ip cef vrf VPN_A 192.168.7.0 detail
192.168.7.0/24, epoch 0
  nexthop 155.1.76.7 FastEthernet0/0.76
```

Copyright © 2010 Internetwork Expert

It appears accurate and complete. Now ping 172.16.7.7 from within VRF VPN_B in SW1 and vice-versa:

```
Rack1SW1#ping vrf VPN_B 172.16.7.7 source loopback 102

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.7.7, timeout is 2 seconds:
Packet sent with a source address of 192.168.7.7
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/4/9 ms

Rack1SW1#ping vrf VPN_A 192.168.7.7 source loopback 101

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.7.7, timeout is 2 seconds:
Packet sent with a source address of 172.16.7.7
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/8 ms
```

Copyright © 2010 Internetwork Expert

14.2 MPLS LDP

- Configure R4, R5 and R6 for MPLS label exchange using an IETF standard protocol.
- Authenticate LDP peering sessions using an MD5 hash and the password value of CISCO.
- Use only a single command on R4 to enable LDP on all OSPF-enabled interfaces.
- Use the physical interface's IP addresses for the LDP session between R4 and R6.

Configuration

Note

The problem with the VRF Lite feature is its lack of scalability. When your VPNs span multiple routers, you need a separate link for every VPN between every pair of routers involved. Even worse, you cannot auto-provision these links using any standard signaling protocol.

A technique that allows for overcoming this limitation emerged from tag-switching technology. Originally developed as a way to accelerate IP packet switching tag-switching ultimately evolved into an advanced dynamic tunneling protocol. The name finally chosen for this technology was MPLS or Multi-Protocol Label Switching.

MPLS "tunnels", known as LSPs (Label Switching Paths), are similar to Frame-Relay or ATM PVCs. As you know, Frame-Relay packets are switched based on the DLCI value found in the header. This DLCI value is purely local to every switch and could be rewritten every time the packet is switched out. The DLCI value could be thought of as a local label used to "switch" the incoming packet. The switching decision is made based on the local table that maps incoming DLCIs to outgoing DLCIs – effectively the Frame-Relay routing table. However, keep in mind that Frame-Relay is a Layer-2 protocol, and the next header following the Frame-Relay header normally belongs to the standard IP, or any other Layer 3, protocol.

MPLS employs a similar principle; a stack of "labels" is inserted between the original Layer 2 and the Layer 3 header. Per the RFC standard, every MPLS label starts with a 20-bit value. However, the actual tag is 32 bits after the addition of all special and reserved fields. MPLS labeling acts as a shim-layer, introducing an extra set of virtual-paths deployed over the Layer 2 topology being used in the network. If a packet carries the stack of MPLS labels, every router performs switching based on the topmost label found in the stack. This is done using the Label FIB or LFIB, which maps incoming labels to their outgoing

Copyright © 2010 Internetwork Expert

equivalents. Thus, MPLS-labeled packets are switched based on a Label Lookup/Switch instead of a lookup in the IP table for the destination prefix. The first router on the edge of an MPLS cloud is known as a LER or Label Edge Router and is responsible for inserting (pushing) the initial label. The routers inside the MPLS cloud are known as LSRs or Label Switching Routers – they swap labels found in the packets (perform pop and push operations) and switch packets further. The last LER along the LSP is responsible for popping the label and switching the packets further using traditional prefix lookup mechanics.

The fundamental core of MPLS centers on how the labels are assigned. A tunnel path is established for every IGP prefix found in the network. This allows us to replace packet switching based on destination prefix lookup with switching based on label swapping. While it does not provide any real performance improvement, it allows for other important features, such as MPLS VPNs, which we are going to discuss in a separate task. Now let's see how MPLS routers exchange their label bindings.

In Frame-Relay, in order to provision a PVC you must manually program DLCI mappings along the path. However, MPLS is a dynamic technology, and uses a special protocol called "Label Distribution Protocol" or LDP to exchange label values. By default, every router will generate a local label for every prefix found in its routing table and advertise it via LDP to its neighbors. This is the label adjacent routers must use when switching for this prefix via the local router. LDP works similar to distance-vector protocols – in that it broadcasts all local prefixes with their respective labels. As soon as a local router learns the labels used by its neighbors for the same prefixes it will program the LFIB with respective label values: incoming label (which has been locally generated) replaced with an outgoing label (used by the neighbor routers).

LDP is a complicated protocol defined in RFC3036. We are not going to discuss all of its functionality in detail. In short, LDP works as follows:

1) In order to enable MPLS switching on an interface and start LDP on the same interface, you need to enter the interface-level command **`mpls ip`**. If you have too many interfaces to enable MPLS on, you may use MPLS LDP auto configuration, which is available when you run OSPF as your IGP protocol. Under the OSPF process, enter the command **`mpls ldp autoconfig`** to activate LDP/MPLS switching on all interfaces running OSPF.

After MPLS has been enabled on an interface, LDP will attempt to discover valid neighbors on the interface. Initially, LDP sends discovery UDP packets to the multicast IP 224.0.0.2 port 646. This corresponds to "all routers" on the local segment.

Copyright © 2010 Internetwork Expert

2) Upon hearing from the other LDP routers, LDP learns their LDP Routing ID's, which are by default the highest Loopback IP addresses. You may change the Router-ID using the command **mpls ldp router-id <interface> force**. If for some reason the Loopback IP addresses are unreachable, a TCP connection will not be established. If you want LDP to establish a TCP connection using the physical interface IP address, use the interface command **mpls ldp discovery transport-address interface** command at the interface level.

3) Using the Router-ID IP addresses as sources, two routers that heard from each other establish a TCP transport connection using the destination port of 646. This connection could be authenticated using an MD5 hash TCP option. The hashing key is defined per-neighbor using the command **mpls ldp neighbor <IP> password <password>.** The IP address here is the neighbor's LDP Router-ID. In order to make the use of passwords mandatory you need the global command **mpls ldp password required**.

4) After the transport connection has been established, the routers exchange prefix and label bindings, resulting in LFIB population. The exchange is performed over the TCP connection established previously.

After the LFIB databases have been populated, label switching may occur. Notice that it is important for all routers within the MPLS domain to have the same prefixes in their routing tables – otherwise the label bindings will not match. This results in the inability to use route summarization with MPLS and traditional IGPs.

The predecessor to LDP was a Cisco proprietary protocol known as Tag Switching Protocol or TDP. It was the default label exchange protocol on Cisco routers until recent IOS releases. If for some reason you are required to use the legacy and proprietary protocol, you may do so using the command **mpls label protocol ldp** in the global configuration mode.

Copyright © 2010 Internetwork Expert

R4:

```
mpls ip
!
mpls ldp router-id Loopback0 force
!
interface FastEthernet 0/1
 mpls ldp discovery transport-address interface
!
router ospf 1
 mpls ldp autoconfig
!
mpls ldp password required
mpls ldp neighbor 150.1.5.5 password CISCO
mpls ldp neighbor 150.1.6.6 password CISCO
```

R6:

```
mpls ip
!
mpls ldp router-id Loopback0 force
!
interface FastEthernet 0/0.146
 mpls ldp discovery transport-address interface
 mpls ip
!
mpls ldp password required
mpls ldp neighbor 150.1.4.4 password CISCO
```

R5:

```
mpls ip
!
mpls ldp router-id Loopback0 force
!
interface Serial 0/1/0
 mpls ip
!
interface Serial 0/0/0
 mpls ip
!
mpls ldp password required
mpls ldp neighbor 150.1.4.4 password CISCO
```

Copyright © 2010 Internetwork Expert

Verification

Note

Check the MPLS LDP peering sessions first. Notice the IP addresses used for the TCP peering sessions. R4 and R6 use physical interface IP addresses to accomplish this.

```
Rack1R4#show mpls ldp neighbor
    Peer LDP Ident: 150.1.5.5:0; Local LDP Ident 150.1.4.4:0
        TCP connection: 150.1.5.5.42010 - 150.1.4.4.646
        State: Oper; Msgs sent/rcvd: 27/27; Downstream
        Up time: 00:11:22
        LDP discovery sources:
          Serial0/0/0.1, Src IP addr: 155.1.0.5
        Addresses bound to peer LDP Ident:
          155.1.58.5      155.1.5.5       155.1.0.5       155.1.45.5
          150.1.5.5
    Peer LDP Ident: 150.1.6.6:0; Local LDP Ident 150.1.4.4:0
        TCP connection: 155.1.146.6.49606 - 155.1.146.4.646
        State: Oper; Msgs sent/rcvd: 12/12; Downstream
        Up time: 00:00:03
        LDP discovery sources:
          FastEthernet0/1, Src IP addr: 155.1.146.6
        Addresses bound to peer LDP Ident:
          155.1.146.6     150.1.6.6
```

Confirm authentication for the neighbors:

```
Rack1R4#show mpls ldp neighbor password
    Peer LDP Ident: 150.1.5.5:0; Local LDP Ident 150.1.4.4:0
        TCP connection: 150.1.5.5.42010 - 150.1.4.4.646
        Password: required, neighbor, in use
        State: Oper; Msgs sent/rcvd: 28/28
    Peer LDP Ident: 150.1.6.6:0; Local LDP Ident 150.1.4.4:0
        TCP connection: 155.1.146.6.49606 - 155.1.146.4.646
        Password: required, neighbor, in use
        State: Oper; Msgs sent/rcvd: 12/12
```

Note

Check the MPLS forwarding tables of all three routers. Notice the labels assigned to the Loopback 0 interfaces. For example, R6 advertises label 17 for 150.X.5.5/32 and receives label 16 for this prefix from R4. Many prefixes have the “Pop Label” action as a result of implicit-null binding.

Copyright © 2010 Internetwork Expert

```
Rack1R6#show mpls forwarding-table
Local  Outgoing      Prefix           Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id     Switched      interface
16     16            150.1.5.5/32     0             Fa0/0.146  155.1.146.4
17     Pop Label     150.1.4.4/32     0             Fa0/0.146  155.1.146.4
18     Pop Label     204.12.1.0/24    0             Fa0/0.146  155.1.146.4
19     18            155.1.58.0/24    0             Fa0/0.146  155.1.146.4
20     Pop Label     155.1.45.0/24    0             Fa0/0.146  155.1.146.4
21     17            155.1.5.0/24     0             Fa0/0.146  155.1.146.4
22     Pop Label     155.1.0.0/24     0             Fa0/0.146  155.1.146.4

Rack1R4#show mpls forwarding-table
Local  Outgoing      Prefix           Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id     Switched      interface
16     Pop Label     150.1.5.5/32     0             Se0/0/0.1  point2point
       No Label      150.1.5.5/32     0             Se0/1/0    point2point
17     Pop Label     155.1.5.0/24     0             Se0/0/0.1  point2point
       No Label      155.1.5.0/24     0             Se0/1/0    point2point
18     Pop Label     155.1.58.0/24    0             Se0/0/0.1  point2point
       No Label      155.1.58.0/24    0             Se0/1/0    point2point
19     Pop Label     150.1.6.6/32     0             Fa0/1      155.1.146.6
Rack1R4#

Rack1R5#show mpls forwarding-table
Local  Outgoing      Prefix           Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id     Switched      interface
16     Pop Label     150.1.4.4/32     0             Se0/0/0    155.1.0.4
       No Label      150.1.4.4/32     0             Se0/1/0    point2point
17     Pop Label     155.1.146.0/24   0             Se0/0/0    155.1.0.4
       No Label      155.1.146.0/24   0             Se0/1/0    point2point
18     Pop Label     204.12.1.0/24    0             Se0/0/0    155.1.0.4
       No Label      204.12.1.0/24    0             Se0/1/0    point2point
19     19            150.1.6.6/32     0             Se0/0/0    155.1.0.4
       No Label      150.1.6.6/32     0             Se0/1/0    point2point
```

Note

Do a traceroute from R6 to R5 and notice the MPLS label popping in the output.

```
Rack1R6#traceroute 150.1.5.5

Type escape sequence to abort.
Tracing the route to 150.1.5.5

  1 155.1.146.4 [MPLS: Label 16 Exp 0] 48 msec 4 msec 48 msec
  2 155.1.45.5 0 msec *  0 msec
```

Copyright © 2010 Internetwork Expert

14.3 MPLS Label Filtering

- Configure R4, R5 and R6 so that the only labels advertised by LDP are the ones for the Loopback 0 interfaces of the mentioned routers.

Configuration

> **Note**
>
> By default, LDP will generate and advertise labels for every prefix found in the local routing table. If you want to change this behavior and generate labels only for specific prefixes, you may use an access-list to select the prefixes eligible for label generation.

R4, R5, R6:

```
access-list 10 permit 150.1.0.0 0.0.255.255
!
no mpls ldp advertise-labels
mpls ldp advertise-labels for 10
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the MPLS forwarding tables in R4, R5 and R6 and notice that only the Loopback 0 prefixes now have labels assigned.

```
Rack1R6#show mpls forwarding-table
Local  Outgoing      Prefix           Bytes Label   Outgoing    Next Hop
Label  Label or VC   or Tunnel Id     Switched      interface
16     16            150.1.5.5/32     0             Fa0/0.146   155.1.146.4
17     Pop Label     150.1.4.4/32     0             Fa0/0.146   155.1.146.4
18     No Label      204.12.1.0/24    0             Fa0/0.146   155.1.146.4
19     No Label      155.1.58.0/24    0             Fa0/0.146   155.1.146.4
20     No Label      155.1.45.0/24    0             Fa0/0.146   155.1.146.4
21     No Label      155.1.5.0/24     0             Fa0/0.146   155.1.146.4
22     No Label      155.1.0.0/24     0             Fa0/0.146   155.1.146.4

Rack1R4#show mpls forwarding-table
Local  Outgoing      Prefix           Bytes Label   Outgoing    Next Hop
Label  Label or VC   or Tunnel Id     Switched      interface
16     Pop Label     150.1.5.5/32     0             Se0/0/0.1   point2point
       No Label      150.1.5.5/32     96            Se0/1/0     point2point
17     No Label      155.1.5.0/24     0             Se0/0/0.1   point2point
       No Label      155.1.5.0/24     0             Se0/1/0     point2point
18     No Label      155.1.58.0/24    0             Se0/0/0.1   point2point
       No Label      155.1.58.0/24    0             Se0/1/0     point2point
19     Pop Label     150.1.6.6/32     0             Fa0/1       155.1.146.6

Rack1R5#show mpls forwarding-table
Local  Outgoing      Prefix           Bytes Label   Outgoing    Next Hop
Label  Label or VC   or Tunnel Id     Switched      interface
16     Pop Label     150.1.4.4/32     0             Se0/0/0     155.1.0.4
       No Label      150.1.4.4/32     0             Se0/1/0     point2point
17     No Label      155.1.146.0/24   0             Se0/0/0     155.1.0.4
       No Label      155.1.146.0/24   0             Se0/1/0     point2point
18     No Label      204.12.1.0/24    0             Se0/0/0     155.1.0.4
       No Label      204.12.1.0/24    0             Se0/1/0     point2point
19     19            150.1.6.6/32     0             Se0/0/0     155.1.0.4
       No Label      150.1.6.6/32     0             Se0/1/0     point2point
```

Copyright © 2010 Internetwork Expert

14.4 MP-BGP VPNv4

- Create two new VRFs named VPN_A and VPN_B on R5 and assign the VLAN 58 and VLAN 5 interfaces into these VRFs repectively.
- Enable VPN route exchange between R5 and R6 use R4 as the BGP route-reflector.
- Make sure IPv4 peering sessions are not activated by default.
- By the end of this task you should be able to ping the connected interfaces inside the VPNs.

Configuration

Note

MPLS technology is a perfect candidate for a dynamic tunneling solution to resolve the scalability issue associated with VRF Lite. The idea of MPLS VPNs is to establish a full-mesh of dynamic MPLS LSRs between the PE (Provider Edge) routers and using those for tunneling VPN packets across the network core. In order to select the proper VRF instance on the endpoint PE router, an additional label is needed that selects the proper FIB entry associated with the target VRF. This requires two labels in the MPLS stack – one label (the topmost) is the transport label, which is being swapped along the entire path between the PEs and the other label (innermost) is the VPN label, which is used to select the proper outgoing VRF CEF entry.

When a tunneling solution was found, a way to distribute VPN routes between the sites was needed. You cannot normally establish IGP protocol adjacencies across MPLS LSRs because they are unidirectional. And even if a bi-directional tunneling solution like mGRE were in use, establishing hundreds of adjacencies for OSPF across a network core will not work well from a scaling perspective. Due to these factors, BGP has been chosen as a universal prefix redistribution protocol.

In order to support these new features, BGP functionality has been enhanced to handle the “VRF” specific routes. A new special MP-BGP (multiprotocol BGP) address family named “VPNv4” (VPN IPv4) has been added to BGP along with a new NLRI format. Every VPNv4 prefix has the RD associated with it and the corresponding MPLS label, in addition to the normal BGP attributes. This allows for transporting different VPN routes together and performing best-path selection independently for each different RD. The VPNv4 address-family capability is activated per-neighbor using the respective address-family configuration. By default, when you create a new BGP neighbor using the command **`neighbor <IP> remote-as N`** the default IPv4 unicast address-family is activated for this neighbor. If for some reason you don’t want this behavior and only need the

Copyright © 2010 Internetwork Expert

VPNv4 prefixes to be sent, you may disable the default behavior via the command **no bgp default ipv4-unicast**.

There are special limitations for iBGP peering sessions that you want to enable for VPNv4 prefix exchange. First of all, they must be sourced from a Loopback 0 interface and secondly, this interface must have a /32 mask. This is needed because the BGP peering IP address is used as the NEXT_HOP for the locally originated VPNv4 prefixes. When the remote BGP router receives those prefixes, it performs a recursive routing lookup for the NEXT_HOP value and finds a label in the LFIB. This label is used as the transport label in the receiving router. Effectively, the NEXT_HOP is used to build the "tunnel" or the transport LSP between the PEs. The VPN label is generated by the BGP process on the advertising router and directly corresponds to the local VRF route. As for the /32 restriction, it is needed to guarantee that the transport LSP terminates on the particular PE router, and not some shared network segment.

In order to inject a particular VRF's routes into BGP, you need to activate the respective address-family under the BGP process and enable route redistribution (e.g. static or connected). All the respective routes belonging to that particular VRF will be injected into the BGP table with their RDs and have their VPN labels generated. The import process is a bit more complicated and based on the concept of "Route Targets".

A Route Target is a BGP extended community attribute. These BGP attributes are transitive, and encoded as 64-bit values (as opposed to normal 32-bit communities). They are used for "enhanced" tagging of VPNv4 prefixes. The need for route-target arises from the fact that one cannot just use Route Distinguishers for prefix importing/exporting, because routes with the same RD may eventually belong to multiple VRFs, when you share their routes.

Here is how route-target based import works. By default, all prefixes redistributed from a VRF into a BGP process are tagged with the extended community **X:Y** specified under the VRF configuration via the command **route-target export X:Y**. You may specify as many export commands as you want to tag prefixes with multiple attributes. On the receiving side, the VRF will import the BGP VPNv4 prefixes with the route-targets matching the local command **route-target import X:Y**. The import process is based entirely on the route-targets, and not the RDs. If the imported routes used to have RDs different from the one used by the local VRF, they are "naturalized" by having the RD changed to the local value. Theoretically, you may assign a route-target to every VPN site, and specify fine-tune import policies, to select the remote site routes accepted locally. Finally, notice that the use of the command **route-target both X:Y** means import and export statements at the same time.

Copyright © 2010 Internetwork Expert

R4:

```
router bgp 100
!
! Disable automatic IPv4 neighbor activation
!
no bgp default ipv4-unicast
 neighbor 150.1.5.5 remote-as 100
 neighbor 150.1.5.5 update-source Loopback0
 neighbor 150.1.6.6 remote-as 100
 neighbor 150.1.6.6 update-source Loopback0
address-family vpnv4 unicast
 neighbor 150.1.5.5 activate
 neighbor 150.1.6.6 activate
!
! Enable sending of ext. communities with VPNv4 routes
!
 neighbor 150.1.5.5 send-community extended
 neighbor 150.1.6.6 send-community extended
 neighbor 150.1.5.5 route-reflector-client
 neighbor 150.1.6.6 route-reflector-client
```

R5:

```
ip vrf VPN_A
 rd 100:1
 route-target both 100:1
!
ip vrf VPN_B
 rd 100:2
 route-target both 100:2
!
interface FastEthernet 0/0
 ip vrf forwarding VPN_A
 ip address 155.1.58.5 255.255.255.0
!
interface FastEthernet 0/1
 ip vrf forwarding VPN_B
 ip address 155.1.5.5 255.255.255.0
```

R6:

```
ip vrf VPN_A
 rd 100:1
 route-target both 100:1
!
ip vrf VPN_B
 rd 100:2
 route-target both 100:2
```

Copyright © 2010 Internetwork Expert

R5 & R6:

```
router bgp 100
 no bgp default ipv4-unicast
 neighbor 150.1.4.4 remote-as 100
 neighbor 150.1.4.4 update-source Loopback0
 address-family vpnv4 unicast
 neighbor 150.1.4.4 activate
 neighbor 150.1.4.4 send-community extended
!
! Activate the address families for VRFs and redistribute routes
!
 address-family ipv4 vrf VPN_A
  redistribute connected
  redistribute static
 address-family ipv4 vrf VPN_B
  redistribute connected
  redistribute static
```

Copyright © 2010 Internetwork Expert

Verification

Note

Check the PE route and look for BGP-learned prefixes in every VRF's routing table:

```
Rack1R5#show ip route vrf VPN_A

Routing Table: VPN_A
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     155.1.0.0/24 is subnetted, 2 subnets
C       155.1.58.0 is directly connected, FastEthernet0/0
B       155.1.67.0 [200/0] via 150.1.6.6, 00:03:08
```

```
Rack1R5#show ip route vrf VPN_B

Routing Table: VPN_B
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     155.1.0.0/24 is subnetted, 2 subnets
C       155.1.5.0 is directly connected, FastEthernet0/1
B       155.1.76.0 [200/0] via 150.1.6.6, 00:04:03
```

Now check the route-reflector BGP table for VPNv4 prefixes. Notice how prefixes are grouped based on the RD.

```
Rack1R4#show bgp vpnv4 unicast all
BGP table version is 5, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 100:1
*>i155.1.58.0/24    150.1.5.5                0    100      0 ?
*>i155.1.67.0/24    150.1.6.6                0    100      0 ?
Route Distinguisher: 100:2
*>i155.1.5.0/24     150.1.5.5                0    100      0 ?
*>i155.1.76.0/24    150.1.6.6                0    100      0 ?
```

Note

Check the route-target values associated with the various prefixes in R4's VPNv4 BGP table:

```
Rack1R4#show bgp vpnv4 unicast all 155.1.76.0 255.255.255.0
BGP routing table entry for 100:2:155.1.76.0/24, version 5
Paths: (1 available, best #1, no table)
  Advertised to update-groups:
        2
  Local, (Received from a RR-client)
    150.1.6.6 (metric 2) from 150.1.6.6 (150.1.6.6)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Extended Community: RT:100:2
      mpls labels in/out nolabel/19

Rack1R4#show bgp vpnv4 unicast all 155.1.67.0 255.255.255.0
BGP routing table entry for 100:1:155.1.67.0/24, version 3
Paths: (1 available, best #1, no table)
  Advertised to update-groups:
        2
  Local, (Received from a RR-client)
    150.1.6.6 (metric 2) from 150.1.6.6 (150.1.6.6)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Extended Community: RT:100:1
      mpls labels in/out nolabel/21
```

Note

Check the end-to-end connectivity across each VPN using ping and traceroute Notice the label stack used to tunnel the VPN packets. The multiple IP addresses on the first hop are due to MPLS multi-path load balancing over the two links connecting R4 and R5.

```
Rack1R5#ping vrf VPN_A 155.1.67.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.67.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 60/60/64 ms

Rack14R5#traceroute vrf VPN_A 155.1.67.6
Type escape sequence to abort.
Tracing the route to 155.14.67.6
  1 155.1.0.4 [MPLS: Labels 18/23 Exp 0] 64 msec
    155.1.45.4 [MPLS: Labels 18/23 Exp 0] 48 msec
    155.1.0.4 [MPLS: Labels 18/23 Exp 0] 60 msec
  2 155.1.67.6 0 msec *  0 msec
Rack14R5#

Rack1R5#ping vrf VPN_B 155.1.76.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 155.1.76.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 60/60/64 ms
```

Copyright © 2010 Internetwork Expert

```
Rack1R5#traceroute vrf VPN_B 155.1.76.7

Type escape sequence to abort.
Tracing the route to 155.1.76.7

  1 155.1.0.4 [MPLS: Labels 18/24 Exp 0] 16 msec
    155.1.45.4 [MPLS: Labels 18/24 Exp 0] 0 msec
    155.1.0.4 [MPLS: Labels 18/24 Exp 0] 16 msec
  2 155.1.76.6 4 msec 12 msec 4 msec
  3 155.1.76.7 12 msec *  12 msec

Rack1R5#traceroute vrf VPN_B 155.1.76.6

Type escape sequence to abort.
Tracing the route to 155.1.76.6

  1 155.1.45.4 [MPLS: Labels 18/24 Exp 0] 4 msec
    155.1.0.4 [MPLS: Labels 18/24 Exp 0] 16 msec
    155.1.45.4 [MPLS: Labels 18/24 Exp 0] 0 msec
  2 155.1.76.6 16 msec *  12 msec
```

Copyright © 2010 Internetwork Expert

Note

Check the MPLS labels (VPN labels) assigned to the VPN prefixes at the PE routers. There are two labels in the stack, one is the VPN label and the other is the transport label. You may find the VPN label in the BGP table and the transport label can be found by looking up the VPNv4 BGP next-hop in the MPLS forwarding table.

```
Rack1R5#show ip bgp vpnv4 vrf VPN_A 155.1.67.0
BGP routing table entry for 100:1:155.1.67.0/24, version 61
Paths: (1 available, best #1, table VPN_A)
  Not advertised to any peer
  Local
    150.1.6.6 (metric 66) from 150.1.4.4 (150.1.4.4)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Extended Community: RT:100:1
      Originator: 150.1.6.6, Cluster list: 150.1.4.4
      mpls labels in/out nolabel/21

Rack1R5#show mpls forwarding-table 150.1.6.6
Local  Outgoing      Prefix            Bytes Label   Outgoing    Next
Hop
Label  Label or VC   or Tunnel Id      Switched      interface
19     19            150.1.6.6/32      0             Se0/0/0
155.1.0.4
       No Label      150.1.6.6/32      0             Se0/1/0
point2point
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

14.5 MP-BGP Prefix Filtering

- Create a new Loopback 101 interface in R5's VRF VPN_A with the IP address of 172.16.5.5/24.
- Create a new Loopback 102 interface in R6's VRF VPN_B with the IP address of 192.168.6.6/24.
- Configure the network to provide bi-directional connectivity between the two new subnets.
- Make sure R6's VPN_A does not see the prefix 172.16.5.0/24 and R5's VPN_B does not see the prefix 192.168.6.0/24.

Configuration

Note

As we found in the previous task, route-target tagging applies to all routes injected from VRF into a BGP process; sometimes we need more granular control over route tagging. In order to accomplish this, Cisco IOS has a feature known as export/import maps. They are configured under the VRF configuration context using the command **`{import|export} map <ROUTE_MAP_NAME>.`** The special export route-map associated with the VRF could match the prefixes based on the prefix-lists, access-lists or extended-communities. All routes not permitted in an export route-map are not exported into the BGP process. The export route-map may also be used to set the extended-community attribute selectively, using the command **`set extcommunity rt`**. This allows for selective tagging of VPN routes.

The import map is used less often than the export map, but still has some good uses. First of all, it allows controlling all routes imported into VRF from BGP based on prefix-lists, access-lists, or extended/standard communities (or any other applicable BGP attribute). This might be helpful when, for example, you want to import all other site's routes with exception to the default route. Notice that by default all prefixes not permitted with the import-map are implicitly denied and not imported.

Copyright © 2010 Internetwork Expert

R5:

```
interface Loopback101
 ip vrf forwarding VPN_A
 ip address 172.16.5.5 255.255.255.0
!
ip prefix-list LO101 permit 172.16.5.0/24
!
route-map VPN_A_EXPORT permit 10
 match ip address prefix-list LO101
 set extcommunity rt 100:55
!
route-map VPN_A_EXPORT permit 20
 set extcommunity rt 100:1
!
ip vrf VPN_A
 export map VPN_A_EXPORT
 route-target import 100:66
```

R6:

```
interface Loopback102
 ip vrf forwarding VPN_B
 ip address 192.168.6.6 255.255.255.0
!
ip prefix-list LO102 permit 192.168.6.0/24
!
route-map VPN_B_EXPORT permit 10
 match ip address prefix-list LO102
 set extcommunity rt 100:66
!
route-map VPN_B_EXPORT permit 20
 set extcommunity rt 100:2
!
ip vrf VPN_B
 export map VPN_B_EXPORT
 route-target import 100:55
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the prefix 172.16.5.0/24 in R6's VPN_A and VPN_B routing tables.

```
Rack1R6#show ip route vrf VPN_B 172.16.5.0
Routing entry for 172.16.5.0/24
  Known via "bgp 100", distance 200, metric 0, type internal
  Last update from 150.1.5.5 00:00:59 ago
  Routing Descriptor Blocks:
  * 150.1.5.5 (Default-IP-Routing-Table), from 150.1.4.4, 00:00:59 ago
      Route metric is 0, traffic share count is 1
      AS Hops 0

Rack1R6#show ip route vrf VPN_A 172.16.5.0
% Network not in table
```

> **Note**
>
> Perform the same checks on R5's VRFs for the prefix 192.168.6.0/24

```
Rack1R5#show ip route vrf VPN_A 192.168.6.0
Routing entry for 192.168.6.0/24
  Known via "bgp 100", distance 200, metric 0, type internal
  Last update from 150.1.6.6 00:02:27 ago
  Routing Descriptor Blocks:
  * 150.1.6.6 (Default-IP-Routing-Table), from 150.1.4.4, 00:02:27 ago
      Route metric is 0, traffic share count is 1
      AS Hops 0

Rack1R5#show ip route vrf VPN_B 192.168.6.0
% Network not in table
```

Copyright © 2010 Internetwork Expert

Check the route-target values associated with the new prefixes.

```
Rack1R4#show ip bgp vpnv4 rd 100:1 172.16.5.0
BGP routing table entry for 100:1:172.16.5.0/24, version 7
Paths: (1 available, best #1, no table)
  Advertised to update-groups:
        2
  Local, (Received from a RR-client)
    150.1.5.5 (metric 65) from 150.1.5.5 (150.1.5.5)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Extended Community: RT:100:55
      mpls labels in/out nolabel/22

Rack1R4#show ip bgp vpnv4 rd 100:2 192.168.6.0
BGP routing table entry for 100:2:192.168.6.0/24, version 9
Paths: (1 available, best #1, no table)
  Advertised to update-groups:
        2
  Local, (Received from a RR-client)
    150.1.6.6 (metric 2) from 150.1.6.6 (150.1.6.6)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Extended Community: RT:100:66
      mpls labels in/out nolabel/25
```

Copyright © 2010 Internetwork Expert

14.6 PE-CE Routing with RIP

- Use RIP as the PE-CE routing protocol for VPN_B sites.
- Remove all static routing used to provide VPN_B connectivity.
- Configure R4 so that the VLAN 43 connection belongs to VPN_B as well.
- Preserve the RIP metric values learned from the CE routers.

Configuration

Note

Even though MP-BGP is the "carrier" routing protocol for MPLS VPNs, a wide variety of PE-CE routing protocols could be used. The PE-CE routing protocols run independently at every site, relying on MP-BGP to exchange their routing information. The route exchange is performed via prefix redistribution between the MP-BGP process and the respective PE-CE routing protocol. A number of features have been added to MP-BGP in order to preserve some valuable routing protocol information that otherwise might be lost during redistribution. These features vary from one PE-CE routing protocol to another.

We start with the simplest of all PE-CE protocols, RIPv2. With RIPv2 you may enable the respective address family under the process configuration mode using the command **`address-family ipv4 vrf <VRF_NAME>.`** You may then specify the network commands as usual. Notice that the timer settings, version and auto-summary settings are global for all VRFs with this approach. When you need to redistribute the MP-BGP routes into RIP, use the command **`redistribute bgp <N> metric {X|transparent}`** under the respective address family. Here N is the BGP process number (AS#) and X is the metric assigned to the RIP routes. If you are using the keyword **`transparent`**, the RIP metrics will be recovered from the BGP MED attribute, which in turn is copied from RIP metrics learned at the remote site. This allows for transparent preservation of RIPv2 metric values across the VPN and better path selection in case of backdoor links.

Naturally, if you want to inject the respective VRF's RIP routes into BGP, you simply redistribute them under the corresponding address-family. The MED value for the new VPNv4 prefixes will be initialized from the RIP metrics of the redistributed routes.

Copyright © 2010 Internetwork Expert

R4:

```
ip vrf VPN_B
 rd 100:2
 no route-target both 100:1
 route-target both 100:2
!
interface FastEthernet0/0
 ip vrf forwarding VPN_B
 ip address 204.12.1.4 255.255.255.0
!
router rip
 version 2
 no auto-summary
 address-family ipv4 vrf VPN_B
!
! The keyword "transparent" carries RIP hop count in
! BGP metric attribute
!
 redistribute bgp 100 metric transparent
 network 204.12.1.0
!
router bgp 100
 address-family ipv4 vrf VPN_B
  redistribute rip
```

R6:

```
router rip
 version 2
 no auto-summary
 address-family ipv4 vrf VPN_B
 redistribute bgp 100 metric transparent
 network 155.1.0.0
!
no ip route vrf VPN_B 172.16.7.0 255.255.255.0 FastEthernet0/0.67
155.1.67.7
```

SW1:

```
no ip route vrf VPN_A 0.0.0.0 0.0.0.0 155.1.76.6
!
router rip
 version 2
 no auto-summary
 address-family ipv4 vrf VPN_B
 network 155.1.0.0
 network 192.168.7.0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the VPN_A routes on SW1 and make sure you see the RIP prefixes advertised by BB3:

```
Rack1SW1#show ip route vrf VPN_B

Routing Table: VPN_B
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

R    204.12.1.0/24 [120/1] via 155.1.76.6, 00:00:04, Vlan76
     155.1.0.0/24 is subnetted, 2 subnets
R       155.1.5.0 [120/1] via 155.1.76.6, 00:00:04, Vlan76
C       155.1.76.0 is directly connected, Vlan76
     172.16.0.0/24 is subnetted, 1 subnets
R       172.16.5.0 [120/1] via 155.1.76.6, 00:00:04, Vlan76
C    192.168.7.0/24 is directly connected, Loopback102
     31.0.0.0/16 is subnetted, 4 subnets
R       31.3.0.0 [120/2] via 155.1.76.6, 00:00:04, Vlan76
R       31.2.0.0 [120/2] via 155.1.76.6, 00:00:04, Vlan76
R       31.1.0.0 [120/2] via 155.1.76.6, 00:00:04, Vlan76
R       31.0.0.0 [120/2] via 155.1.76.6, 00:00:05, Vlan76
     30.0.0.0/16 is subnetted, 4 subnets
R       30.2.0.0 [120/2] via 155.1.76.6, 00:00:05, Vlan76
R       30.3.0.0 [120/2] via 155.1.76.6, 00:00:05, Vlan76
R       30.0.0.0 [120/2] via 155.1.76.6, 00:00:05, Vlan76
R       30.1.0.0 [120/2] via 155.1.76.6, 00:00:05, Vlan76
```

> **Note**
>
> Check the routing table for VPN_B on R4 and look for the same prefixes:

Copyright © 2010 Internetwork Expert

```
Rack1R4#show ip route vrf VPN_B

Routing Table: VPN_A
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

C    204.12.1.0/24 is directly connected, FastEthernet0/0
     155.1.0.0/24 is subnetted, 2 subnets
B       155.1.58.0 [200/0] via 150.1.5.5, 00:32:29
B       155.1.67.0 [200/0] via 150.1.6.6, 00:32:29
     31.0.0.0/16 is subnetted, 4 subnets
R       31.3.0.0 [120/1] via 204.12.1.254, 00:00:25, FastEthernet0/0
R       31.2.0.0 [120/1] via 204.12.1.254, 00:00:25, FastEthernet0/0
R       31.1.0.0 [120/1] via 204.12.1.254, 00:00:25, FastEthernet0/0
R       31.0.0.0 [120/1] via 204.12.1.254, 00:00:25, FastEthernet0/0
     30.0.0.0/16 is subnetted, 4 subnets
R       30.2.0.0 [120/1] via 204.12.1.254, 00:00:25, FastEthernet0/0
R       30.3.0.0 [120/1] via 204.12.1.254, 00:00:00, FastEthernet0/0
R       30.0.0.0 [120/1] via 204.12.1.254, 00:00:00, FastEthernet0/0
R       30.1.0.0 [120/1] via 204.12.1.254, 00:00:00, FastEthernet0/0
```

Note

Look at R6's BGP table to see the RIP prefixes being carried in BGP updates. Notice the "metric" field value:

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip bgp vpnv4 vrf VPN_B
BGP table version is 210, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 100:2 (default for vrf VPN_B)
*>i30.0.0.0/16      150.6.4.4                1    100      0 ?
*>i30.1.0.0/16      150.6.4.4                1    100      0 ?
*>i30.2.0.0/16      150.6.4.4                1    100      0 ?
*>i30.3.0.0/16      150.6.4.4                1    100      0 ?
*>i31.0.0.0/16      150.6.4.4                1    100      0 ?
*>i31.1.0.0/16      150.6.4.4                1    100      0 ?
*>i31.2.0.0/16      150.6.4.4                1    100      0 ?
*>i31.3.0.0/16      150.6.4.4                1    100      0 ?
*>i155.6.5.0/24     150.6.5.5                0    100      0 ?
*> 155.6.76.0/24    0.0.0.0                  0         32768 ?
*>i172.16.5.0/24    150.6.5.5                0    100      0 ?
*> 192.168.6.0      0.0.0.0                  0         32768 ?
*>i204.12.6.0       150.6.4.4                0    100      0 ?
```

Note

Test end-to-end connectivity across VPN_B:

```
Rack1SW1#traceroute vrf VPN_B 30.0.0.1

Type escape sequence to abort.
Tracing the route to 30.0.0.1

  1 155.1.76.6 0 msec 0 msec 8 msec
  2 204.12.1.4 0 msec 0 msec 0 msec
  3 204.12.1.254 9 msec *  0 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

14.7 PE-CE Routing with OSPF

- Use OSPF as the PE-CE routing protocol for the VPN_A sites and configure PE-CE routers in area 1.
- Use the same OSPF process IDs at R6 and R5 but ensure SW1 and SW2 can reach each other.
- Create a new Loopback interface on SW2 with the IP address 172.16.8.8/24 and make sure R6 sees only a /16 summary for this prefix.

Configuration

Note

OSPF is a link-state routing protocol, which relies on building a network topology table prior to calculating the optimal routes. However, as we remember MPLS VPNs rely on using MP-BGP to transport the routing information across the MPLS backbone. This means no OSPF adjacencies can be established across the core. Please note that OSPF uses distance-vector like updates sent across Area 0 to redistribute routing information between different areas. This allows for interpreting the MP-BGP cloud as one "super area 0" that is used to link all OSPF areas at different sites. This special "virtual area" is called the OSPF super-backbone and it is emulated by passing OSPF VRF routing information in MP-BGP updates. The use of the super-backbone allows us to avoid using area 0 at all, as the super-backbone performs the same function. Thus we can have non-zero OSPF areas at different VPN sites connecting via the MP-BGP mesh without the need for an area 0 at any site. However, it's perfectly okay to have area 0 at different sites along with non-zero areas attached to the super-backbone as well. The main design principle that should be followed is that all areas are connected to the super-backbone in a loop-less start-like manner.

We briefly discussed BGP extended communities before. With OSPF, special extended communities are used to propagate additional OSPF prefix information. All OSPF routes redistributed into MP-BGP are treated pretty much like Type 3 summary LSAs, as they enter the super-backbone from other areas. When injected into BGP, OSPF prefixes have two extended-community attributes attached to them. One of the attributes is known as domain-id, which is equal to the OSPF process numbers on the local router OR explicitly configured using the command **`domain-id`** under the OSPF process. The purpose of this attribute is to identify OSPF processes belonging to different VPNs. It is assumed that you configured all OSPF processes within the same VPN using the same domain-id (e.g. the same process number). If for some reason you exchange routes between two different VPNs using different domain-ids, the OSPF process will interpret all such prefixes as if they are Type-5 External LSAs, effectively external routes.

Copyright © 2010 Internetwork Expert

The other extended community attribute is known as the OSPF route-type which has three significant fields: source area, route-type and option. They are usually depicted as triple X:Y:Z. Here Y=2 for intra-area learned prefix, Y=3 for inter-area routes, Y=5 for external prefixes and Y=7 for NSSA routes. The last value, Z, is the metric type for Y=5 or 7 - it's either 1 for E1 or 2 for E2. Notice that all routes redistributed from BGP into OSPF appear like *inter-area* routes even if they belong to the same area number at different sites. This effect is due to the fact that the LSAs cross the super-backbone and essentially are inter-area routes.

The last BGP attribute used to carry OSPF information is MED or metric, which copies the original route's metric from the routing table. The route-type attribute is needed to allow for proper OSPF best-path calculation when routes are inserted into the OSPF database based on redistribution from BGP. Notice that the routes travelling the MP-BGP cloud do not increment their metric unless you manually change the MED attribute for incoming BGP prefixes. This might be needed sometimes for proper OSPF best-path selection.

All the above mechanics are implemented automatically once you configure route redistribution between a VRF process and BGP. Notice that you create a separate OSPF process for every VRF using the command **`router ospf X vrf NAME`**, unlike RIP, where the whole process is shared among VRFs.

The fact that there is an additional backbone area could introduce the possibility of routing loops if for some reason the VPN sites are not connected in a star-like manner. To reduce this risk, OSPF implements some basic loop prevention rules.

First of all, all summary LSAs generated from the routes redistributed from BGP have a special "Down" bit set in the LSA headers. If a router receives a summary-LSA with the down bit set on an interface that belongs to a VRF, it simply drops this LSA. This is to prevent the case of routing loops for multi-homed sites, when a summary LSA is flooded across the CE site and delivered back to another PE. However, this feature may have an undesirable effect when you have a CE router configured with multiple VRFs. In this case, you may want to enter the OSPF process command **`capability vrf-lite`** on the CE router. This will disable the default loop-prevention capability. Be advised that some IOS versions do not support this feature (e.g. older IOSs or some Catalyst IOS revisions). If you have such a router configured for multi-VRF and experience route black holing, configure the PE routers with different domain-IDs – this will force all redistributed routes to become external and bypass the down-bit check. This is the trick used in the solution for this particular scenario, as the Catalyst switches do not support the OSPF "VRF-lite" feature.

Copyright © 2010 Internetwork Expert

The other feature is based on the route tagging. All routes redistributed via a particular PE will carry the OSPF route tag with the BGP AS number encoded inside. The receiving router that has VRFs enabled will compare the AS number in the tag with the local BGP AS number. If they match, this could mean the LSA has looped back to another PE connecting the same site to the MPLS backbone. If for some reason you face this issue and need to get rid of it, simply apply a proper tag to the redistributed routes, for instance using the OSPF command **`redistribute bgp N subnets tag Y`**.

R5:

```
router ospf 100 vrf VPN_A
 domain-id 0.0.0.5
 log-adjacency-changes
 redistribute bgp 100 subnets
 network 0.0.0.0 255.255.255.255 area 1
!
router bgp 100
address-family ipv4 vrf VPN_A
 redistribute ospf 100 vrf VPN_A
```

R6:

```
router ospf 100 vrf VPN_A
 domain-id 0.0.0.6
 log-adjacency-changes
 redistribute bgp 100 subnets
 network 0.0.0.0 255.255.255.255 area 1
 !
 ! We use the summary address because the routes are generated
 ! using type 5 external LSAs.
 !
 summary-address 172.16.0.0 255.255.0.0
!
router bgp 100
address-family ipv4 vrf VPN_A
 redistribute ospf 100 vrf VPN_A
```

SW1:

```
no ip route vrf VPN_A 0.0.0.0 0.0.0.0 155.14.76.6
!
router ospf 1 vrf VPN_A
 network 0.0.0.0 255.255.255.255 area 1
```

SW2:

```
ip routing
!
router ospf 1
 network 0.0.0.0 255.255.255.255 area 1
!
interface Loopback100
 ip address 172.16.8.8 255.255.255.0
```

Copyright © 2010 Internetwork Expert

Verification

Note

Check the routing tables of SW2 and SW1 VRF_A. Notice that SW1 sees the /16 summary for the 172.16.8.8/24 prefix (actually it was /32 due to the OSPF loopback network type).

```
Rack1SW2#show ip route ospf
     155.1.0.0/24 is subnetted, 3 subnets
O E2    155.1.67.0 [110/1] via 155.1.58.5, 00:17:22, Vlan58
     172.16.0.0/16 is variably subnetted, 4 subnets, 3 masks
O E2    172.16.7.7/32 [110/2] via 155.1.58.5, 00:17:05, Vlan58
O       172.16.5.5/32 [110/2] via 155.1.58.5, 00:17:34, Vlan58
O E2    172.16.0.0/16 [110/2] via 155.1.58.5, 00:05:12, Vlan58
O E2 192.168.6.0/24 [110/1] via 155.1.58.5, 00:17:34, Vlan58

Rack1SW1#show ip route vrf VPN_A

Routing Table: VPN_A
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     155.1.0.0/24 is subnetted, 3 subnets
O E2    155.1.8.0 [110/2] via 155.1.67.6, 00:17:53, Vlan67
O E2    155.1.58.0 [110/1] via 155.1.67.6, 00:17:53, Vlan67
C       155.1.67.0 is directly connected, Vlan67
     172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
C       172.16.7.0/24 is directly connected, Loopback101
O E2    172.16.0.0/16 [110/2] via 155.1.67.6, 00:06:06, Vlan67
     150.1.0.0/32 is subnetted, 1 subnets
O E2    150.1.8.8 [110/2] via 155.1.67.6, 00:17:53, Vlan67
```

Copyright © 2010 Internetwork Expert

Note

Check that the OSPF processes on R5 and R6 are connected to the OSPF super-backbone. Notice that both routers are ASBRs since they redistribute BGP routes.

```
Rack1R5#show ip ospf 100
 Routing Process "ospf 100" with ID 172.16.5.5
   Domain ID type 0x0005, value 0.0.0.5
 Start time: 1d07h, Time elapsed: 16:48:19.640
 Supports only single TOS(TOS0) routes
 Supports opaque LSA
 Supports Link-local Signaling (LLS)
 Supports area transit capability
 Connected to MPLS VPN Superbackbone, VRF VPN_A
 It is an area border and autonomous system boundary router
 Redistributing External Routes from,
    bgp 100, includes subnets in redistribution
 Router is not originating router-LSAs with maximum metric
 Initial SPF schedule delay 5000 msecs
 Minimum hold time between two consecutive SPFs 10000 msecs
 Maximum wait time between two consecutive SPFs 10000 msecs
 Incremental-SPF disabled
 Minimum LSA interval 5 secs
 Minimum LSA arrival 1000 msecs
 LSA group pacing timer 240 secs
 Interface flood pacing timer 33 msecs
 Retransmission pacing timer 66 msecs
 Number of external LSA 4. Checksum Sum 0x02404D
 Number of opaque AS LSA 0. Checksum Sum 0x000000
 Number of DCbitless external and opaque AS LSA 0
 Number of DoNotAge external and opaque AS LSA 0
 Number of areas in this router is 1. 1 normal 0 stub 0 nssa
 Number of areas transit capable is 0
 External flood list length 0
 IETF NSF helper support enabled
 Cisco NSF helper support enabled
    Area 1
        Number of interfaces in this area is 2 (1 loopback)
        Area has no authentication
        SPF algorithm last executed 16:48:16.004 ago
        SPF algorithm executed 2 times
        Area ranges are
        Number of LSA 3. Checksum Sum 0x01063B
        Number of opaque link LSA 0. Checksum Sum 0x000000
        Number of DCbitless LSA 0
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0
```

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip ospf 100
 Routing Process "ospf 100" with ID 155.1.67.6
   Domain ID type 0x0005, value 0.0.0.6
 Start time: 1w1d, Time elapsed: 17:18:03.756
 Supports only single TOS(TOS0) routes
 Supports opaque LSA
 Supports Link-local Signaling (LLS)
 Supports area transit capability
 Connected to MPLS VPN Superbackbone, VRF VPN_A
 It is an area border and autonomous system boundary router
 Redistributing External Routes from,
    bgp 100, includes subnets in redistribution
 Router is not originating router-LSAs with maximum metric
 Initial SPF schedule delay 5000 msecs
 Minimum hold time between two consecutive SPFs 10000 msecs
 Maximum wait time between two consecutive SPFs 10000 msecs
 Incremental-SPF disabled
 Minimum LSA interval 5 secs
 Minimum LSA arrival 1000 msecs
 LSA group pacing timer 240 secs
 Interface flood pacing timer 33 msecs
 Retransmission pacing timer 66 msecs
 Number of external LSA 4. Checksum Sum 0x021F17
 Number of opaque AS LSA 0. Checksum Sum 0x000000
 Number of DCbitless external and opaque AS LSA 0
 Number of DoNotAge external and opaque AS LSA 0
 Number of areas in this router is 1. 1 normal 0 stub 0 nssa
 Number of areas transit capable is 0
 External flood list length 0
 IETF NSF helper support enabled
 Cisco NSF helper support enabled
    Area 1
        Number of interfaces in this area is 1
        Area has no authentication
        SPF algorithm last executed 17:17:59.820 ago
        SPF algorithm executed 2 times
        Area ranges are
        Number of LSA 3. Checksum Sum 0x0118E5
        Number of opaque link LSA 0. Checksum Sum 0x000000
        Number of DCbitless LSA 0
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0
```

Note

Now check the extended community attributes associated with the OSPF prefixes carried from R5 to R6. Take any prefix, for example 172.16.8.8. Pay attention to the Domain ID encoded in 0x00000**05**0200 as 0x05 and the route-type attribute value of 0.0.0.1:2:0 which stands for intra-area route received from area 1.

Copyright © 2010 Internetwork Expert

```
Rack1R6#show bgp vpnv4 unicast vrf VPN_A 172.16.8.8
BGP routing table entry for 100:1:172.16.8.8/32, version 236
Paths: (1 available, best #1, table VPN_A)
  Not advertised to any peer
  Local
    150.1.5.5 (metric 66) from 150.1.4.4 (150.1.4.4)
      Origin incomplete, metric 2, localpref 100, valid, internal, best
      Extended Community: RT:100:1 OSPF DOMAIN ID:0x0005:0x000000050200
        OSPF RT:0.0.0.1:2:0 OSPF ROUTER ID:172.16.5.5:0
      Originator: 150.1.5.5, Cluster list: 150.1.4.4
      mpls labels in/out nolabel/25
```

Copyright © 2010 Internetwork Expert

> **Note**
>
> Now you may want to check the OSPF database on SW1 to see what type of LSA it has for the subnet 172.16.8.0/24

```
Rack1SW1#show ip ospf database external 172.16.8.8

            OSPF Router with ID (150.1.7.7) (Process ID 1)

Rack1SW1#show ip ospf database external 172.16.0.0

            OSPF Router with ID (150.1.7.7) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 779
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 172.16.0.0 (External Network Number )
  Advertising Router: 155.1.67.6
  LS Seq Number: 80000001
  Checksum: 0x32DD
  Length: 36
  Network Mask: /16
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 2
        Forward Address: 0.0.0.0
        External Route Tag: 0
```

> **Note**
>
> Now verify end-to-end connectivity across the VPN:

```
Rack1SW1#traceroute vrf VPN_A 172.16.8.8

Type escape sequence to abort.
Tracing the route to 172.16.8.8

  1 155.1.67.6 0 msec 0 msec 0 msec
  2 155.1.146.4 101 msec 50 msec 101 msec
  3 155.1.58.5 0 msec 8 msec 0 msec
  4 155.1.58.8 9 msec *  0 mses
```

Copyright © 2010 Internetwork Expert

14.8 OSPF Sham-Link

- Reconfigure SW1 to a pure CE router in VRF VPN_A; be sure to remove the VRF-lite configuration.
- Preserve the OSPF routing between VPN_A sites at R5 and R6.
- Provision a back-door link between SW1 and SW2 using any of the interconnections as a L3 interface.
- Configure VPN_A OSPF processes on R5 and R6 for the same domain-id.
- Ensure SW1 and SW2 prefer the path across the MPLS core to the backdoor link.

Configuration

Note

As we learned in the previous scenario, OSPF prefixes are transported via the MPLS VPN core using MP-BGP and interpreted either as type-3 summary LSAs or type-5 external LSAs on the receiving end. This may pose some difficulties in the event there is a backdoor link connecting two VPN sites directly. Many times the backdoor link is supposed to be used as a backup path (i.e. it's a slow leased line) and the MPLS VPN cloud should be used as the primary way between the two sites. However, if the link is in the same area as the PE/CE routers, the PE routers will prefer the path across the back-door link, as OSPF treats all paths across it as intra-area, and the prefixes received via MP-BGP are interpreted as inter-area.

This is a fundamental issue, which requires the MPLS core to be a part of the same OSPF area to resolve the conflict. The solution is called an OSPF sham-link, which is a special tunnel similar to a virtual-link connecting two PE routers and configured in the same area as the PE routers. This link is used to establish an OSPF adjacency and for the exchange of LSAs. The LSAs are then loaded in the OSPF database and the sham-link is used for intra-area path computations. However, when the routes are being installed in the respective VRF RIB, the forwarding information is based on looking up the MP-BGP learned routes based on exact prefix-match. The corresponding VPN and transport labels are then used for actual packet forwarding across the MPLS core. Thus the information loaded across the sham-link is used only for SPF calculations and best-path selection – the actual forwarding is being done based on the information learned via MP-BGP.

Copyright © 2010 Internetwork Expert

Configuring sham-links is a bit different from configuring virtual-links. First and foremost, sham-links are sourced off actual interfaces configured in the respective VRF. Commonly these are Loopback interfaces, used as endpoints for the sham-link tunnel. Notice that the IP addresses for these interfaces should be advertised into the VRF routing table by means other than OSPF, most commonly via BGP. After you have the endpoints reachable across the MPLS VPN core, you may configure the sham-link using the command **`area 1 sham-link <SRC> <DST> cost X`**. Here <SRC> and <DST> are the IP addresses for the sham-link source and destination. The cost is the OSPF metric value associated with traversing the MPLS core. If the endpoints are reachable, OSPF will establish an adjacency on the link treating it as a point-to-point connection and re-calculate the shortest paths.

In the solution below, we change the OSPF network command to cover only the interfaces that should establish OSPF adjacencies. The sham-link's endpoints should not be advertised into OSPF. Also note that the OSPF summary command we configured in the previous task no longer has any effect. SW1 and SW2 see each other's routes as intra-area even though they are sent across the MPLS core network.

Copyright © 2010 Internetwork Expert

R5:

```
router ospf 100 vrf VPN_A
 no domain-id 0.0.0.5
 area 1 sham-link 150.1.55.55 150.1.66.66 cost 1
 !
 ! We need to make sure we're not advertising the sham-link
 ! endpoints into OSPF
 !
 no network 0.0.0.0 255.255.255.255 area 1
 network 155.1.58.5 0.0.0.0 area 1
!
interface Loopback 200
 ip vrf forwarding VPN_A
 ip address 150.1.55.55 255.255.255.255
!
router bgp 100
address-family ipv4 vrf VPN_A
 network 150.1.55.55 mask 255.255.255.255
```

R6:

```
router ospf 100 vrf VPN_A
 no domain-id 0.0.0.5
 area 1 sham-link 150.1.66.66 150.1.55.55 cost 1
 !
 ! We need to make sure we're not advertising the sham-link
 ! endpoints into OSPF
 !
 no network 0.0.0.0 255.255.255.255 area 1
 network 155.1.67.6 0.0.0.0 area 1
!
interface Loopback 200
 ip vrf forwarding VPN_A
 ip address 150.1.66.66 255.255.255.255
!
router bgp 100
address-family ipv4 vrf VPN_A
 network 150.1.66.66 mask 255.255.255.255
```

Copyright © 2010 Internetwork Expert

SW1:

```
!
! Adjust the cost to make sure the backdoor link is used for backup
!
interface FastEthernet 0/15
 no switchport
 ip address 155.1.78.7 255.255.255.0
 ip ospf cost 9999
!
! Remove the old VRF configurations
!
interface Vlan 67
 no ip vrf forwarding VPN_A
 ip address 155.1.67.7 255.255.255.0
!
interface Loopback 101
 ip address 172.16.7.7 255.255.255.0
!
no router ospf 1
router ospf 1
 network 0.0.0.0 255.255.255.255 area 1
```

SW2:

```
interface FastEthernet 0/15
 no switchport
 ip address 155.1.78.8 255.255.255.0
 ip ospf cost 9999
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the sham-link status:

```
Rack1R5#show ip ospf sham-links
Sham Link OSPF_SL0 to address 150.1.66.66 is up
Area 1 source address 150.1.55.55
  Run as demand circuit
  DoNotAge LSA allowed. Cost of using 1 State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40,
    Hello due in 00:00:06
    Adjacency State FULL (Hello suppressed)
    Index 2/2, retransmission queue length 0, number of retransmission
0
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 0, maximum is 0
    Last retransmission scan time is 0 msec, maximum is 0 msec
```

> **Note**
>
> Confirm that the endpoint addresses were learned via BGP:

```
Rack1R5# sh ip route vrf VPN_A 150.1.66.66
Routing entry for 150.1.66.66/32
  Known via "bgp 100", distance 200, metric 0, type internal
  Redistributing via ospf 100
  Advertised by ospf 100 subnets
  Last update from 150.1.6.6 00:05:56 ago
  Routing Descriptor Blocks:
  * 150.1.6.6 (Default-IP-Routing-Table), from 150.1.4.4, 00:05:56 ago
      Route metric is 0, traffic share count is 1
      AS Hops 0

Rack1R6#show ip route vrf VPN_A 150.1.55.55
Routing entry for 150.1.55.55/32
  Known via "bgp 100", distance 200, metric 0, type internal
  Redistributing via ospf 100
  Advertised by ospf 100 subnets
  Last update from 150.1.5.5 00:04:06 ago
  Routing Descriptor Blocks:
  * 150.1.5.5 (Default-IP-Routing-Table), from 150.1.4.4, 00:04:06 ago
      Route metric is 0, traffic share count is 1
      AS Hops 0
```

Copyright © 2010 Internetwork Expert

Note

Check that both SW1 and SW2 prefer reaching each other across the MPLS core, and all the prefixes are seen as OSPF intra-area.

```
Rack1SW1#show ip route ospf
     155.1.0.0/24 is subnetted, 7 subnets
O       155.1.8.0 [110/4] via 155.1.67.6, 00:01:47, Vlan67
O       155.1.58.0 [110/3] via 155.1.67.6, 00:01:47, Vlan67
     172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
O       172.16.8.8/32 [110/4] via 155.1.67.6, 00:01:47, Vlan67
O E2 192.168.6.0/24 [110/1] via 155.1.67.6, 00:01:47, Vlan67
     150.1.0.0/16 is variably subnetted, 4 subnets, 2 masks
O E2    150.1.66.66/32 [110/1] via 155.1.67.6, 00:01:47, Vlan67
O E2    150.1.55.55/32 [110/1] via 155.1.67.6, 00:01:47, Vlan67
O       150.1.8.8/32 [110/4] via 155.1.67.6, 00:01:47, Vlan67

Rack1SW2#show ip route ospf
     155.1.0.0/24 is subnetted, 7 subnets
O       155.1.7.0 [110/4] via 155.1.58.5, 00:02:05, Vlan58
O       155.1.37.0 [110/4] via 155.1.58.5, 00:02:05, Vlan58
O       155.1.79.0 [110/4] via 155.1.58.5, 00:02:05, Vlan58
O       155.1.67.0 [110/3] via 155.1.58.5, 00:02:05, Vlan58
     172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
O       172.16.7.7/32 [110/4] via 155.1.58.5, 00:02:05, Vlan58
O E2 192.168.6.0/24 [110/1] via 155.1.58.5, 00:02:05, Vlan58
     150.1.0.0/16 is variably subnetted, 4 subnets, 2 masks
O E2    150.1.66.66/32 [110/1] via 155.1.58.5, 00:02:06, Vlan58
O E2    150.1.55.55/32 [110/1] via 155.1.58.5, 00:02:06, Vlan58
O       150.1.7.7/32 [110/4] via 155.1.58.5, 00:02:06, Vlan58
```

Note

Check the end-to-end connectivity between SW1 and SW2:

```
Rack1SW1#traceroute 150.1.8.8

Type escape sequence to abort.
Tracing the route to 150.1.8.8

  1 155.1.67.6 0 msec 8 msec 0 msec
  2 155.1.146.4 50 msec 101 msec 50 msec
  3 155.1.58.5 17 msec 17 msec 17 msec
  4 155.1.58.8 25 msec *  17 msec
```

Copyright © 2010 Internetwork Expert

Note

Check the data-plane information for prefix 150.1.8.8 in R5 to confirm it's being switched using the MPLS VPN information:

```
Rack1R6#show bgp vpnv4 unicast vrf VPN_A 150.1.8.8
BGP routing table entry for 100:1:150.1.8.8/32, version 418
Paths: (1 available, best #1, table VPN_A, RIB-failure(17))
  Not advertised to any peer
  Local
    150.1.5.5 (metric 66) from 150.1.4.4 (150.1.4.4)
      Origin incomplete, metric 2, localpref 100, valid, internal, best
      Extended Community: RT:100:1 OSPF DOMAIN ID:0x0005:0x000000640200
        OSPF RT:0.0.0.1:2:0 OSPF ROUTER ID:172.16.5.5:0
      Originator: 150.1.5.5, Cluster list: 150.1.4.4
      mpls labels in/out nolabel/35

Rack1R6#show mpls forwarding-table 150.1.5.5
Local  Outgoing      Prefix            Bytes Label   Outgoing    Next
Hop
Label  Label or VC   or Tunnel Id      Switched      interface
16     16            150.1.5.5/32      0             Fa0/0.146
155.1.146.4

Rack1R6#show ip cef vrf VPN_A 150.1.8.8
150.1.8.8/32
  nexthop 155.1.146.4 FastEthernet0/0.146 label 16 35
```

Note

Now confirm that the backdoor link works if the primary link fails. Shutdown SW1's connection to R6 and confirm that the prefixes are available via the backup path. Don't forget to "no shutdown" the interface when you're done with the verifications!

Copyright © 2010 Internetwork Expert

SW1:

```
interface Vlan 67
 shutdown

Rack1SW1#show ip route ospf
     155.1.0.0/24 is subnetted, 7 subnets
O       155.1.8.0 [110/10000] via 155.1.78.8, 00:00:09,
FastEthernet0/15
O       155.1.58.0 [110/10000] via 155.1.78.8, 00:00:09,
FastEthernet0/15
O       155.1.67.0 [110/10002] via 155.1.78.8, 00:00:09,
FastEthernet0/15
     172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
O       172.16.8.8/32 [110/10000] via 155.1.78.8, 00:00:09,
FastEthernet0/15
O E2 192.168.6.0/24 [110/1] via 155.1.78.8, 00:00:09, FastEthernet0/15
     150.1.0.0/16 is variably subnetted, 4 subnets, 2 masks
O E2    150.1.66.66/32 [110/1] via 155.1.78.8, 00:00:09,
FastEthernet0/15
O E2    150.1.55.55/32 [110/1] via 155.1.78.8, 00:00:09,
FastEthernet0/15
O       150.1.8.8/32 [110/10000] via 155.1.78.8, 00:00:09,
FastEthernet0/15
```

Copyright © 2010 Internetwork Expert

14.9 PE-CE Routing with EIGRP

- Replace OSPF with EIGRP on the PE-CE as the routing protocol used for VPN_A.
- Ensure the backdoor link is used for backup and that the primary path between the two sites is across the MPLS VPN cloud.
- Make R4's VLAN 43 interface a part of VPN_A and advertise it into EIGRP.
- All EIGRP routers should be in the same AS.

Configuration

Note

EIGRP is an advanced distance vector protocol that uses composite metric for best path selection. One issue with transporting EIGRP routes over MP-BGP is preserving the original metric values, the route type, the source AS# and the remote Router ID. These all are encoded using special BGP extended-community attributes that allow the remote site to properly decode the incoming routing update information. Why not just carry the metric in the MED attribute? The problem is that the remote site may happen to have different K values and have different a resulting metric. Not something you would really see in real life, but still the implementation could account for this. Of course, the local EIGRP process will treat all prefixes originated in different remote AS# as external, per the normal EIGRP rules. Thus the AS# and the Router-ID information could be crucial to resolve potential routing loops in scenarios with backdoor links between the VPN sites.

Another special attribute used with EIGRP prefixes redistributed into MP-BGP is known as the "cost attribute". While this attribute was designed to have pretty wide use, the main idea is to change the BGP best-path selection process. The problem with MPLS VPN route redistribution is that the same route may enter the PE's BGP table using redistribution – i.e. learned from a CE router and via a BGP update – i.e. learned from a remote site. Per the BGP best-path selection process, locally redistributed prefixes have a BGP weight of 32768 which make them always win the best-path selection process. Therefore, BGP will always choose the locally received update even if the remote site has better a EIGRP metric to reach the destination.

Copyright © 2010 Internetwork Expert

To resolve this issue, EIGRP prefixes redistributed into MP-BGP will have the Cost attribute value set to their composite metric. The BGP process will honor the Cost attribute value before ANY other best-path selection option if the attribute is present. The prefix with the lowest cost will immediately win the best-path selection and will be redistributed into local the EIGRP process. This process happens automatically and does not require any additional configuration. Notice that the Cost attribute is NOT needed with OSPF because the prefixes received via MP-BGP are treated as inter-area and are always less preferred compared to the same prefixes learned as intra-area from a CE device. As for the case when RIP is used for PE-CE routing, Cisco IOS does not implement the Cost attribute at all, and thus the locally redistributed prefixes will always get preferred over MP-BGP learned prefixes, preventing effective RIP deployment in scenarios with backdoor links. This is not considered a big issue as no one really uses RIP for large-scale deployments and backup routing nowadays.

When configuring EIGRP for MPLS VPNs, the same process is shared among multiple VRFs. An address family must be configured per VRF under the routing process and for every address-family you must configure an AS number using the command **autonomous-system N**. This command is mandatory to enable EIGRP for that particular VRF. After that, all you need is to enter the normal network statements as in any EIGRP configuration.

Copyright © 2010 Internetwork Expert

R4:

```
ip vrf VPN_A
 rd 100:1
 route-target both 100:1
!
router eigrp 100
 no auto-summary
 address-family ipv4 vrf VPN_A
  autonomous-system 100
  network 204.12.1.0 0.0.0.255
  redistribute bgp 100 metric 1 1 1 1 1
!
router bgp 100
 address-family ipv4 vrf VPN_A
  redistribute eigrp 100
!
interface FastEthernet 0/0
 ip vrf forwarding VPN_A
 ip address 204.12.1.4 255.255.255.0
```

R5:

```
no router ospf 100
!
router eigrp 100
 no auto-summary
 address-family ipv4 vrf VPN_A
  autonomous-system 100
  network 155.1.58.5 0.0.0.0
  redistribute bgp 100 metric 1 1 1 1 1
!
router bgp 100
 address-family ipv4 vrf VPN_A
  redistribute eigrp 100
```

Copyright © 2010 Internetwork Expert

R6:
```
no router ospf 100
!
router eigrp 100
 no auto-summary
 address-family ipv4 vrf VPN_A
  autonomous-system 100
  network 155.1.67.6 0.0.0.0
  redistribute bgp 100 metric 1 1 1 1 1
!
router bgp 100
 address-family ipv4 vrf VPN_A
  redistribute eigrp 100
```

SW1:
```
no router ospf 1
!
router eigrp 100
 no auto-summary
 network 0.0.0.0 255.255.255.255
!
! Make sure this link is backup
!
interface FastEthernet 0/15
 delay 1000
```

SW2:
```
no router ospf 1
!
router eigrp 100
 no auto-summary
 network 0.0.0.0 255.255.255.255
!
! Make sure this link is backup
!
interface FastEthernet 0/15
 delay 1000
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the EIGRP routes at SW1 and SW2. Notice that the primary path is chosen via the PE-routers.

```
Rack1SW1#show ip route eigrp
D    204.12.1.0/24 [90/28416] via 155.1.67.6, 00:37:23, Vlan67
     155.1.0.0/24 is subnetted, 7 subnets
D       155.1.8.0 [90/28672] via 155.1.67.6, 00:39:53, Vlan67
D       155.1.58.0 [90/28416] via 155.1.67.6, 00:39:53, Vlan67
     172.16.0.0/24 is subnetted, 2 subnets
D       172.16.8.0 [90/156416] via 155.1.67.6, 00:39:53, Vlan67
D EX 192.168.6.0/24
           [170/2560256512] via 155.1.78.8, 00:39:53, FastEthernet0/15
     150.1.0.0/16 is variably subnetted, 4 subnets, 2 masks
D EX    150.1.66.66/32 [170/2560000512] via 155.1.67.6, 00:39:54,
Vlan67
D EX    150.1.55.55/32 [170/2560000512] via 155.1.67.6, 00:39:54,
Vlan67
D       150.1.8.0/24 [90/156416] via 155.1.67.6, 00:39:54, Vlan67
Rack1SW1#
```

> **Note**
>
> Check the BGP prefixes for redistributed EIGRP routes on R5 and R6 and notice the extended communities used to carry the additional EIGRP information. Also, notice the Cost attribute that carries the original route's metric.

```
Rack1R5#show bgp vpnv4 unicast vrf VPN_A 204.12.1.0
BGP routing table entry for 100:1:204.12.1.0/24, version 505
Paths: (1 available, best #1, table VPN_A)
Flag: 0x820
  Not advertised to any peer
  Local
    150.1.4.4 (metric 65) from 150.1.4.4 (150.1.4.4)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Extended Community: RT:100:1 Cost:pre-bestpath:128:28160
0x8800:32768:0
        0x8801:100:2560 0x8802:65280:25600 0x8803:65281:1500
      mpls labels in/out nolabel/25
```

Copyright © 2010 Internetwork Expert

Note

Take another prefix, which is reachable both via the backdoor link and the MPLS VPN cloud. Notice that BGP has only the path across the MPLS VPN – this is because the CE router has accepted this path as well and suppressed advertising the backup path to the PE router. This is all thanks to BGP cost attribute.

```
Rack1R5#show bgp vpnv4 unicast vrf VPN_A 150.1.8.0
BGP routing table entry for 100:1:150.1.8.0/24, version 446
Paths: (1 available, best #1, table VPN_A)
  Advertised to update-groups:
        1
  Local
    155.1.58.8 from 0.0.0.0 (150.1.5.5)
      Origin incomplete, metric 156160, localpref 100, weight 32768,
valid, sourced, best
      Extended Community: RT:100:1 Cost:pre-bestpath:128:156160
        0x8800:32768:0 0x8801:100:130560 0x8802:65281:25600
0x8803:65281:1500
      mpls labels in/out 25/nolabel
```

> **Note**
>
> Now verify end-to-end connectivity and confirm that the backup path (backdoor link) works as well.

```
Rack1SW1#traceroute 150.1.8.8

Type escape sequence to abort.
Tracing the route to 150.1.8.8

  1 155.1.67.6 0 msec 0 msec 0 msec
  2 155.1.146.4 101 msec 50 msec 101 msec
  3 155.1.58.5 17 msec 25 msec 17 msec
  4 155.1.58.8 25 msec *  17 msec

Rack1SW1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface vlan 67
Rack1SW1(config-if)#shutdown
Rack1SW1(config-if)#^Z

Rack1SW1#sh ip route 150.1.8.0
Routing entry for 150.1.8.0/24
  Known via "eigrp 100", distance 90, metric 409600, type internal
  Redistributing via eigrp 100
  Last update from 155.1.78.8 on FastEthernet0/15, 00:00:05 ago
  Routing Descriptor Blocks:
  * 155.1.78.8, from 155.1.78.8, 00:00:05 ago, via FastEthernet0/15
      Route metric is 409600, traffic share count is 1
      Total delay is 15000 microseconds, minimum bandwidth is 100000
Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

14.10 EIGRP Site-of-Origin

- Configure R5 and R6 to prevent temporary routing loops that may occur due to mutual redistribution between EIGRP and MP-BGP.
- Ensure the path across the MPLS VPN core is used as the primary path between SW1 and SW2.

Configuration

Note

With multi-homed scenarios similar to the ones we are using, BGP and EIGRP perform mutual redistribution at PE routers. This may potentially result in transient routing loops, when a prefix is withdrawn in MP-BGP at one PE router and it is not timely propagated into EIGRP. EIGRP could then feed the invalid information back to BGP at another PE router, keeping the false information circulating between the PE routers until it's eliminated by counting to infinity. The core of this problem is the mutual redistribution that allows the information learned from BGP at one site to re-enter BGP at another site. This problem is commonly resolved by tagging the prefixes from one protocol while redistributing them into another, and then blocking the prefixes from re-entering the domain of origin based on the tag.

Implementing this solution using EIGRP route tags and BGP communities could be cumbersome if done using regular methods. Therefore, Cisco IOS implements a special feature known as EIGRP Site-of-Origin which uses an extended community appended to BGP and EIGRP routing updates (EIGRP's TLV format allows adding this information easily). The feature is configured at the interface level using the command **`ip vrf sitemap <ROUTE_MAP>`** where ROUTE_MAP is a regular route-map that applies the command **`set extcommunity soo ASN:XX`** where ASN:XX is the "VPN identifier" common for all PEs of the same multi-homed site.

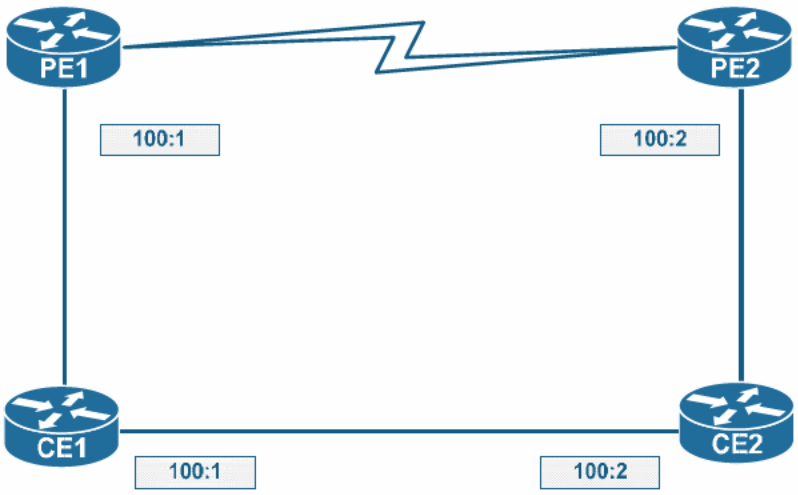
All BGP prefixes redistributed into EIGRP and sent over the interface with the SoO set will have this extended community appended, but only if the community is not already present. If the update to be sent already has the same extended community value set, it is discarded as being redistributed back to the same site. The next thing that this feature does is apply the same extended community to all EIGRP routes received on the interface and redistributed into BGP. All these actions are performed by IOS automatically – you only need to apply the route-map to the relevant interfaces.

Copyright © 2010 Internetwork Expert

There are two common ways of applying the VRF sitemap feature.

1) The SoO is applied on the PE interfaces facing the CE routers. Every PE router uses the same SoO value. This prevents PE routers from learning MP-BGP originated routes from CE routers. However, the side-effect is that the MPLS core cannot be used as a backup path between the segments of the multi-homed sites in case the backdoor link fails. This is because MP-BGP updates for in-site EIGRP prefixes carry the same SoO that is applied to the PE interfaces facing the CE routers and thus cannot be redistributed back to the same site.

2) If you need to preserve the path across the MPLS core network, you should use different SoO values at every PE router of a multi-homed site. However, this means the MP-BGP information injected into EIGRP at one PE router will reach the other PEs without being blocked. To prevent this effect, an additional SoO interface should be configured on the CE routers with the backdoor link. Look at the figure below and notice the SoO settings on the respective PE/CE interfaces:



In this case, if PE1 redistributes a prefix into MP-BGP it will be tagged with 100:1. The prefix will reach PE2 and pass down to CE2. The prefix will finally be stopped at CE1 by SoO filtering and will never loop back to PE1 again. The same applies to prefixes redistributed into MP-BGP by PE2.

Copyright © 2010 Internetwork Expert

R5:
```
route-map EIGRP_SOO
 set extcommunity soo 100:15
!
interface FastEthernet 0/0
 ip vrf sitemap EIGRP_SOO
```

R6:
```
route-map EIGRP_SOO
 set extcommunity soo 100:16
!
interface FastEthernet 0/0.67
 ip vrf sitemap EIGRP_SOO
```

SW1:
```
route-map EIGRP_SOO
 set extcommunity soo 100:16
!
interface FastEthernet 0/15
 ip vrf sitemap EIGRP_SOO
```

SW2:
```
route-map EIGRP_SOO
 set extcommunity soo 100:15
!
interface FastEthernet 0/15
 ip vrf sitemap EIGRP_SOO
```

Copyright © 2010 Internetwork Expert

Verification

Note

Check the BGP prefixes for EIGRP routes and notice the SoO community values:

```
Rack1R6#show bgp vpnv4 unicast vrf VPN_A 150.1.8.0
BGP routing table entry for 100:1:150.1.8.0/24, version 680
Paths: (1 available, best #1, table VPN_A)
  Not advertised to any peer
  Local
    150.1.5.5 (metric 66) from 150.1.4.4 (150.1.4.4)
      Origin incomplete, metric 156160, localpref 100, valid, internal,
best
      Extended Community: SoO:100:15 RT:100:1 Cost:pre-
bestpath:128:156160
        0x8800:32768:0 0x8801:100:130560 0x8802:65281:25600
0x8803:65281:1500
      Originator: 150.1.5.5, Cluster list: 150.1.4.4
      mpls labels in/out nolabel/25

Rack1R6#show bgp vpnv4 unicast vrf VPN_A 150.1.7.0
BGP routing table entry for 100:1:150.1.7.0/24, version 700
Paths: (1 available, best #1, table VPN_A)
  Advertised to update-groups:
        1
  Local
    155.1.67.7 from 0.0.0.0 (150.1.6.6)
      Origin incomplete, metric 156160, localpref 100, weight 32768,
valid, sourced, best
      Extended Community: SoO:100:16 RT:100:1 Cost:pre-
bestpath:128:156160
        0x8800:32768:0 0x8801:100:130560 0x8802:65281:25600
0x8803:65281:1500
      mpls labels in/out 18/nolabel
```

Copyright © 2010 Internetwork Expert

Note

Check the routing tables in the CE routers and confirm that primary paths are via the MPLS core:

```
Rack1SW1#sh ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

D    204.12.1.0/24 [90/28416] via 155.1.67.6, 00:33:53, Vlan67
     155.1.0.0/24 is subnetted, 7 subnets
D       155.1.8.0 [90/28672] via 155.1.67.6, 00:33:53, Vlan67
C       155.1.7.0 is directly connected, Vlan7
D       155.1.58.0 [90/28416] via 155.1.67.6, 00:33:53, Vlan67
C       155.1.37.0 is directly connected, FastEthernet0/3
C       155.1.78.0 is directly connected, FastEthernet0/15
C       155.1.79.0 is directly connected, Vlan79
C       155.1.67.0 is directly connected, Vlan67
     172.16.0.0/24 is subnetted, 2 subnets
D       172.16.8.0 [90/156416] via 155.1.67.6, 00:33:54, Vlan67
C       172.16.7.0 is directly connected, Loopback101
D EX 192.168.6.0/24
           [170/2560256512] via 155.1.78.8, 00:33:54, FastEthernet0/15
     150.1.0.0/16 is variably subnetted, 4 subnets, 2 masks
C       150.1.7.0/24 is directly connected, Loopback0
D EX    150.1.66.66/32 [170/2560000512] via 155.1.67.6, 00:33:55,
Vlan67
D EX    150.1.55.55/32 [170/2560000512] via 155.1.67.6, 00:33:55,
Vlan67
D       150.1.8.0/24 [90/156416] via 155.1.67.6, 00:33:55, Vlan67
```

Copyright © 2010 Internetwork Expert

```
Rack1SW2#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set
D    204.12.1.0/24 [90/28416] via 155.1.58.5, 00:35:33, Vlan58
     155.1.0.0/24 is subnetted, 7 subnets
C       155.1.8.0 is directly connected, Vlan8
D       155.1.7.0 [90/28672] via 155.1.58.5, 00:35:33, Vlan58
C       155.1.58.0 is directly connected, Vlan58
D       155.1.37.0 [90/30976] via 155.1.58.5, 00:35:33, Vlan58
C       155.1.78.0 is directly connected, FastEthernet0/15
D       155.1.79.0 [90/28672] via 155.1.58.5, 00:35:33, Vlan58
D       155.1.67.0 [90/28416] via 155.1.58.5, 00:35:33, Vlan58
     172.16.0.0/24 is subnetted, 2 subnets
C       172.16.8.0 is directly connected, Loopback100
D       172.16.7.0 [90/156416] via 155.1.58.5, 00:35:34, Vlan58
D EX 192.168.6.0/24 [170/2560000512] via 155.1.58.5, 01:30:38, Vlan58
     150.1.0.0/16 is variably subnetted, 4 subnets, 2 masks
D       150.1.7.0/24 [90/156416] via 155.1.58.5, 00:35:34, Vlan58
D EX    150.1.66.66/32 [170/2560000512] via 155.1.58.5, 00:35:34,
Vlan58
D EX    150.1.55.55/32 [170/2560000512] via 155.1.58.5, 00:35:34,
Vlan58
C       150.1.8.0/24 is directly connected, Loopback0
```

Copyright © 2010 Internetwork Expert

Note

Shutdown the PE-CE link at SW1 and confirm that the backup link is still usable (don't forget to bring the link back up when you're done verifying!)

```
Rack1SW1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface vlan 67
Rack1SW1(config-if)#shutdown

Rack1SW1#show ip route eigrp
D    204.12.1.0/24 [90/284416] via 155.1.78.8, 00:00:03,
FastEthernet0/15
     155.1.0.0/24 is subnetted, 6 subnets
D       155.1.8.0 [90/281856] via 155.1.78.8, 00:00:03,
FastEthernet0/15
D       155.1.58.0 [90/281856] via 155.1.78.8, 00:00:03,
FastEthernet0/15
     172.16.0.0/24 is subnetted, 2 subnets
D       172.16.8.0 [90/409600] via 155.1.78.8, 00:00:03,
FastEthernet0/15
D EX 192.168.6.0/24
           [170/2560256512] via 155.1.78.8, 00:36:26, FastEthernet0/15
     150.1.0.0/16 is variably subnetted, 4 subnets, 2 masks
D EX    150.1.66.66/32
           [170/2560256512] via 155.1.78.8, 00:00:03, FastEthernet0/15
D EX    150.1.55.55/32
           [170/2560256512] via 155.1.78.8, 00:00:03, FastEthernet0/15
D       150.1.8.0/24 [90/409600] via 155.1.78.8, 00:00:04,
FastEthernet0/15
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

14.11 PE-CE Routing with BGP

- Erase all EIGRP configurations for VPN_A on R5, R6, SW1 and SW2.
- Configure BGP AS 78 on SW1 and SW2 and peer the CE routers with R5 and R6.
- Advertise the Loopback 0 interfaces of SW1 and SW2 into BGP and ensure end-to-end reachability.

Configuration

Note

BGP seems to be a natural fit for PE-CE routing due to its perfect integration with core MP-BGP routing. Advantages – perfect scalability and precise route control. Disadvantage – slow convergence for intra-site routing. However, as a balanced option you may use an IGP for intra-site routing and BGP for PE-CE peering to get the best of both worlds.

One common problem that arises from the use of BGP is that multiple sites may re-use the same AS number. Due to the BGP loop-prevention mechanism, this will filter BGP updates sent between the sites. There are two solutions to this problem: the first one is configuring the **`allowas-in`** option inbound for the CE peering session. The other option is configuring the **`as-override`** option on the PE routers for the eBGP peering sessions with the CE routers. What this option does, is compares the remote-AS number with the AS number stored in the end of the AS_PATH attribute. If they match, the AS number in AS_PATH is replaced with the local PE router's AS number. The AS numbers used in the BGP peering sessions remain the same, only the AS_PATH attribute is changed when updates are relayed. Notice that the length of the AS_PATH attribute remains the same, which allows for proper best-path selection.

Configuring BGP for PE-CE routing requires only activating the respective VRF's address family under the global BGP process and configuring the BGP peering sessions under this VRF. There is no need to configure any redistribution in this case, as routes are propagated into the VPNv4 table automatically. Just as usual, prefix import and export is controlled by the route-target extended communities. If you want to, you may additionally redistribute other IGP protocols into BGP, or redistribute connected interfaces.

Copyright © 2010 Internetwork Expert

R5:
```
no router eigrp 100
!
router bgp 100
 address-family ipv4 vrf VPN_A
 neighbor 155.1.58.8 remote-as 78
 neighbor 155.1.58.8 as-override
```

R6:
```
no router eigrp 100
!
router bgp 100
 address-family ipv4 vrf VPN_A
 neighbor 155.1.67.7 remote-as 78
 neighbor 155.1.67.7 as-override
```

SW1:
```
no router eigrp 100
!
 router bgp 78
 neighbor 155.1.67.6 remote-as 100
 network 150.1.7.0 mask 255.255.255.0
```

SW2:
```
no router eigrp 100
!
router bgp 78
 neighbor 155.1.58.5 remote-as 100
 network 150.1.8.0 mask 255.255.255.0
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the BGP routes learned on SW1. Look into the BGP table and confirm that the "source" AS 58 no longer appears in the AS_PATH and is replaced by the "core" AS 100. After this, verify end-to-end connectivity across the MPLS core.

```
Rack1SW1#show ip route bgp
B    204.12.1.0/24 [20/0] via 155.1.67.6, 00:00:40
     155.1.0.0/24 is subnetted, 6 subnets
B       155.1.58.0 [20/0] via 155.1.67.6, 00:00:40
     150.1.0.0/16 is variably subnetted, 4 subnets, 2 masks
B       150.1.66.66/32 [20/0] via 155.1.67.6, 00:00:40
B       150.1.55.55/32 [20/0] via 155.1.67.6, 00:00:40
B       150.1.8.0/24 [20/0] via 155.1.67.6, 00:00:40

Rack1SW1#show ip bgp
BGP table version is 9, local router ID is 172.16.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 150.1.7.0/24     0.0.0.0                  0         32768 i
*> 150.1.8.0/24     155.1.67.6                             0 100 100 i
*> 150.1.55.55/32   155.1.67.6                             0 100 i
*> 150.1.66.66/32   155.1.67.6               0             0 100 i
*> 155.1.58.0/24    155.1.67.6                             0 100 ?
r> 155.1.67.0/24    155.1.67.6               0             0 100 ?
*> 204.12.1.0       155.1.67.6                             0 100 ?

Rack1SW1#traceroute
Protocol [ip]:
Target IP address: 150.1.8.8
Source address: 150.1.7.7
Numeric display [n]:
Timeout in seconds [3]:
Probe count [3]:
Minimum Time to Live [1]:
Maximum Time to Live [30]:
Port Number [33434]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Type escape sequence to abort.
Tracing the route to 150.1.8.8

  1 155.1.67.6 [AS 100] 0 msec 9 msec 0 msec
  2 155.1.146.4 50 msec 101 msec 50 msec
  3 155.1.58.5 [AS 100] 42 msec 51 msec 50 msec
  4 155.1.58.8 [AS 100] 25 msec *  17 msec
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

14.12 BGP SoO Attribute

- Configure a "backdoor" BGP peering session between SW1 and SW2 using the inter-switch link configured previously.
- Ensure this configuration does not result in routing loops due to BGP's loop-prevention mechanism being disabled by the AS-Override feature.

Configuration

Note

The use of the AS-Override feature allows circumventing the BGP loop prevention mechanism. However, the drawback is that this may result in routing-loops for multi-homed sites with backdoor links. If a site injects a prefix into MP-BGP the prefix may re-enter the same site via another PE due to the fact that AS-Override will replace the source AS.

In order to overcome this issue for multi-homed sites, BGP implements a SoO attribute similar to the one used for EIGRP. However, this time you configure the SoO value per-neighbor peering session in the PE routers. There are two basic methods of doing this:

1) Setting the SoO attribute for incoming/outgoing prefixes on the peering session using the command **`neighbor <IP> soo <VALUE>`**. This command is available in IOS 12.4(11)T and higher and is a more elegant way of configuring the feature.

2) Setting the SoO attribute using a route-map command **`set extcommunity soo <VALUE>`** and applying this route map inbound to the neighbor session via the command **`neighbor <IP> route-map <NAME> in`**.

BGP SoO works very similar to EIGRP SoO: if an incoming or outgoing update has the SoO value matching the locally configured one, the update is dropped. Otherwise, the update is tagged with the SoO extended community. Based on this similarity, you may quickly find out that using the same SoO at all PEs will prevent the MPLS VPN core from being used as a backup path in the event the backdoor link fails. The solution is, again, using different SoO values at PE routers and applying SoO at the CE routers connected to the backdoor link. However, in this scenario we simply configure the SoO on the PE routers.

Copyright © 2010 Internetwork Expert

R5:
```
router bgp 100
 address-family ipv4 vrf VPN_A
 neighbor 155.1.58.8 soo 100:1
```

R6:
```
router bgp 100
 address-family ipv4 vrf VPN_A
 neighbor 155.1.67.7 soo 100:1
```

SW1:
```
router bgp 78
 neighbor 155.1.78.8 remote-as 78
```

SW2:
```
router bgp 78
 neighbor 155.1.78.7 remote-as 78
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Look into the BGP tables of the PE routers and confirm that BGP prefixes learned from the PEs are now tagged with the SoO attribute. Start with R6 and the prefix 150.1.8.0/24. Confirm that this prefix is not advertised to the CE router (SW1):

```
Rack1R6#show ip bgp vpnv4 vrf VPN_A 150.1.8.0
BGP routing table entry for 100:1:150.1.8.0/24, version 25
Paths: (1 available, best #1, table VPN_A)
Flag: 0x820
  Advertised to update-groups:
        1
  78
    155.1.67.7 from 155.1.67.7 (172.16.7.7)
      Origin IGP, localpref 100, valid, external, best
      Extended Community: SoO:100:1 RT:100:1
      mpls labels in/out 35/nolabel

Rack1R6#sh ip bgp vpnv4 vrf VPN_A neighbors 155.1.67.7 advertised-
routes
BGP table version is 23, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 100:1 (default for vrf VPN_A)
*>i150.1.55.55/32   150.1.5.5                0    100      0 i
*> 150.1.66.66/32   0.0.0.0                  0         32768 i
*>i155.1.58.0/24    150.1.5.5                0    100      0 ?
*> 155.1.67.0/24    0.0.0.0                  0         32768 ?
*>i204.12.1.0       150.1.4.4                0    100      0 ?

Total number of prefixes 5
```

> **Note**
>
> R5 has two prefixes in the BGP table for 150.1.7.0: one learned from R6 via R4 (the RR) and another learned from SW2. Both are tagged with the SoO of 100:1. R5 does not advertise the prefix 150.1.7.0/24 to SW2 per the rules of SoO filtering.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show ip bgp vpnv4 vrf VPN_A 150.1.7.0
BGP routing table entry for 100:1:150.1.7.0/24, version 2
Paths: (2 available, best #2, table VPN_A)
Flag: 0x820
  Advertised to update-groups:
        1
  78
    150.1.6.6 (metric 66) from 150.1.4.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, internal
      Extended Community: SoO:100:1 RT:100:1
      Originator: 150.1.6.6, Cluster list: 150.1.4.4
      mpls labels in/out 42/48
  78
    155.1.58.8 from 155.1.58.8 (172.16.8.8)
      Origin IGP, localpref 100, valid, external, best
      Extended Community: SoO:100:1 RT:100:1
      mpls labels in/out 42/nolabel

Rack1R5#show ip bgp vpnv4 vrf VPN_A neighbors 155.1.58.8 advertised-
routes
BGP table version is 25, local router ID is 150.1.5.5
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 100:1 (default for vrf VPN_A)
*> 150.1.55.55/32   0.0.0.0                  0         32768 i
*>i150.1.66.66/32   150.1.6.6                0    100      0 i
*> 155.1.58.0/24    0.0.0.0                  0         32768 ?
*>i155.1.67.0/24    150.1.6.6                0    100      0 ?
*> 172.16.5.0/24    0.0.0.0                  0         32768 ?
*>i192.168.6.0      150.1.6.6                0    100      0 ?
*>i204.12.1.0       150.1.4.4                0    100      0 ?

Total number of prefixes 7
```

14.13 Internet Access

- Enable RIP on R6's connection to BB1 and configure R6 so that VPN_A customers may access these routes.
- You are allowed to use one static route to accomplish this task.
- Only the 150.X.0.0/16 subnets should be allowed to access the Internet.

Configuration

Note

Internet access is a common service, which often needs to be combined with MPLS VPN services provided by an SP. In many cases, the solution is to simply provide an additional PE-CE link that belongs to the global routing table in the PE, or the special VRF containing the Internet routes. In some situations, however, it is desirable to use a single point of attachment for both types of service. In this case, two problems need to be addressed:

1) Injecting the Internet routes into the respective VRF. In many situations, it is enough to inject just the default route. If the Internet access is provided via the global routing table, a special type of static route should be used. If the Internet access is provided via a special VPN, classic route export/import could be used.

2) The Internet routing table should learn about the prefixes found in the VPN that needs VPN access. In many situations, VPNs use private IP addressing, and thus a kind of NAT mechanism is required to hide the source IP addresses and translate them into routable equivalents.

In this scenario, we are going to deal with the most common case, when Internet access is provided via the global routing table. We assume the VPNs are using private IP addressing, which requires NAT to be configured when accessing the Internet. In our case, "Internet" is emulated by the prefixes learned from BB1. Here is the list of the steps required to configure NAT-based Internet access via the global routing table:

1) Create a special default VRF route that resolves via the global routing table. The syntax is **`ip route vrf <NAME> 0.0.0.0 0.0.0.0 <NEXT_HOP> global`**. The last keyword means that even though the route is bound to a VRF, the next hop should be resolved using the global routing table, and FIB should be programmed accordingly. You may need to redistribute this route into MP-BGP to propagate it to all VPN sites.

Copyright © 2010 Internetwork Expert

2) Enable NAT on the Internet and VPN links. You need to set up all VPN-facing links as NAT inside and the Internet facing link as NAT outside. After this, create a global NAT address pool if needed. This address pool should be reachable via the global routing table.

3) The last step is the core of NAT-based VPN Internet access. You need to configure a source NAT translation rule that matches the VPN source IP addresses and specifies either the global pool or the global interface for address translation. This rule should end with the keyword **`vrf <VRF_NAME>`** which selects the source IP addresses only from the particular VRF. The ability to apply the NAT rules to IP addresses found in a particular VRF is a new feature of the NAT translation engine.

After these three steps have been configured, the VPN clients will use the injected default route to access all non-matched prefixes. The packets will be routed to the Internet-access router which has the NAT rule set up. After this, the source IP addresses will be changed to the addresses routable via the global table, and the respective NAT entries will be created. Even though the packets originally belong to the VRF, they will be switched using the global routing table and the NAT entries will serve as "entry points" back to the VRF for the returning packets.

Notice that our solution uses BGP, and thus we cannot simply redistribute the static default route. We use the BGP address-family command **`default-information originate`** to accomplish this task. This command requires the default route to be advertised into the local BGP table by using either redistribution or the **`network`** command. Only after this will the default route be propagated to all peers.

Copyright © 2010 Internetwork Expert

R6:

```
router rip
 version 2
 no auto-summary
 network 54.0.0.0
!
ip route vrf VPN_A 0.0.0.0 0.0.0.0 54.1.1.254 global
!
router bgp 100
!
! Let all sites know about the default route.
!
 address-family ipv4 vrf VPN_A
 default-information originate
 redistribute static
!
interface Serial 0/0/0
 ip nat outside
!
interface FastEthernet 0/0.146
 ip nat inside
!
interface FastEthernet 0/0.67
 ip nat inside
!
ip access-list standard VPN_PREFIXES
 permit 150.1.0.0 0.0.255.255
!
ip nat inside source list VPN_PREFIXES interface Serial 0/0/0 vrf VPN_A
overload
```

Copyright © 2010 Internetwork Expert

Verification

Note

Check the routing table of any of the CEs for the default route:

```
Rack1SW1#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 155.1.67.6 to network 0.0.0.0

B    204.12.1.0/24 [20/0] via 155.1.67.6, 00:00:15
     155.1.0.0/24 is subnetted, 6 subnets
C       155.1.7.0 is directly connected, Vlan7
B       155.1.58.0 [20/0] via 155.1.67.6, 00:00:15
C       155.1.37.0 is directly connected, FastEthernet0/3
C       155.1.78.0 is directly connected, FastEthernet0/15
C       155.1.79.0 is directly connected, Vlan79
C       155.1.67.0 is directly connected, Vlan67
     172.16.0.0/24 is subnetted, 2 subnets
B       172.16.5.0 [200/0] via 155.1.58.5, 00:00:27
C       172.16.7.0 is directly connected, Loopback101
B    192.168.6.0/24 [200/0] via 155.1.58.5, 00:00:05
B    192.168.7.0/24 [20/0] via 155.1.67.6, 00:00:28
     150.1.0.0/16 is variably subnetted, 4 subnets, 2 masks
C       150.1.7.0/24 is directly connected, Loopback0
B       150.1.66.66/32 [20/0] via 155.1.67.6, 00:00:29
B       150.1.55.55/32 [20/0] via 155.1.67.6, 00:00:16
B       150.1.8.0/24 [200/0] via 155.1.78.8, 01:41:57
B*   0.0.0.0/0 [20/0] via 155.1.67.6, 00:00:29
```

Note

Test connectivity to any of the “Internet” routes, learned by R6 via RIP from the CEs and verify the associated NAT translations for each on R6:

Copyright © 2010 Internetwork Expert

```
Rack1R6#show ip route rip
R    212.18.1.0/24 [120/1] via 54.1.1.254, 00:00:11, Serial0/0/0
R    212.18.0.0/24 [120/1] via 54.1.1.254, 00:00:11, Serial0/0/0
R    212.18.3.0/24 [120/1] via 54.1.1.254, 00:00:11, Serial0/0/0
R    212.18.2.0/24 [120/1] via 54.1.1.254, 00:00:11, Serial0/0/0

Rack1SW1#ping 212.18.1.1 source loopback0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 212.18.1.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.7.7
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 25/30/34 ms
Rack1SW1#

Rack1R6#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside
global
icmp 54.1.1.6:3        150.1.7.7:3        212.18.1.1:3
212.18.1.1:3

Rack1SW2#ping 212.18.1.1 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 212.18.1.1, timeout is 2 seconds:
Packet sent with a source address of 150.1.8.8
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 25/32/34 ms
Rack1SW2#

Rack1R6#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside
global
icmp 54.1.1.6:0        150.1.8.8:0        212.18.1.1:0
212.18.1.1:0
```

Copyright © 2010 Internetwork Expert

Copyright © 2010 Internetwork Expert

14.14 AToM

- Create a P2P L2VPN connection between the VLAN 5 interface of R5 and the unused interface of R6.
- Use the method that provides the minimum amount of transport overhead.

Configuration

Note

Point-to-Point L2 VPNs provide transparent L2 connectivity to the attached circuits of two separate sites connected via a packet network. The general idea of L2 VPNs is to provide this transparent connectivity without the need for provisioning actual L2 connections across the SP core. There are two common solutions for P2P L2VPNs: Any Transport over MPLS or AToM and Layer 2 Tunneling Protocol Version 3 or L2TPv3. Both technologies work as a simple P2P tunnel known as a "pseudowire", capturing L2 frames at one side and transporting them across the packet network to the other side of the connection. Some of the information in the original frame might be dropped at ingress and recovered at egress, depending on the technology.

In this task we will be concerned with AToM, which provides less encapsulation overhead than L2TPv3 as it uses MPLS encapsulation for tunneling. Also, we will be dealing with Ethernet as the only underlying technology for L2VPN connectivity. First of all, AToM transport is configured manually, using the interface-level command **`xconnect`**. This command takes additional parameters, including the pseudowire class, destination IP address and the VC (virtual circuit) ID. The pseudowire class is defined globally using the command **`pseudowire-class <NAME>`** and specifies additional L2VPN parameters. As soon as AToM learns the destination IP address and the VC ID, it attempts to establish a targeted LDP session with the remote end. The source IP address used for this session is based on the configured MPLS LDP router identifier, i.e. per the normal LDP rules. If the remote end is configured properly, the session will become active and the two parties will attempt to exchange the labels identifying their attached circuits (ACs). For these negotiations to succeed, the following parameters need to match:

1) The VC identifiers at each end must match, so that the connection is uniquely addressed. It's up to you to provision an optimal numbering scheme
2) The VC type: either port-mode or VLAN-mode for Ethernet L2VPNs. A Port-Mode connection is configured by applying the **`xconnect`** command to the physical interface. It will replicate all Ethernet frames received, including all tagged frames. In VLAN-based mode you configure the **`xconnect`** statement under a sub-interface with the proper encapsulation and VLAN number specified. This mode will only intercept and forward the respective VLAN frames across the

Copyright © 2010 Internetwork Expert

pseudowire.
3) The MTU settings for the interfaces being connected. Notice that a MTU mismatch often happens in situations when you connect Gigabit and Fast Ethernet ports together.
4) Authentication settings must also match. Since AToM uses LDP for label exchange, you may find that your LDP password policy may require you to authenticate the new session as well.

If all parameters match, the label exchange will succeed, and the pseudowire will become active. The packets intercepted at one end of the connection, are prepended with two MPLS labels: the topmost label is the transport label for the other end's /32 IP address and the second label identifies the remote AC. AToM additionally prepends a special "Control Word" that carries extra information, which is not relevant in the context of this discussion. The resulting MPLS packet is label switched to the remote end, where the AC label is used to select the proper outgoing circuit and to switch the packet.

Notice that if you are using AToM in VLAN-based mode, you might or might not have to use the same VLAN id at both ends. This depends on the platform's hardware/software ability to do VLAN rewrite; Most ISR IOS-based platforms can do that, though.

Copyright © 2010 Internetwork Expert

R5:

```
default interface FastEthernet 0/1
interface FastEthernet 0/1
 xconnect 150.1.6.6 100 encapsulation mpls
!
! We need the password per our LDP password policy
!
mpls ldp neighbor 150.1.6.6 password CISCO
```

R6:

```
interface FastEthernet 0/1
 no shutdown
 xconnect 150.1.5.5 100 encapsulation mpls
!
mpls ldp neighbor 150.1.5.5 password CISCO
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Issue the following command to check the status of the LDP session and L2 transport connection. Notice the labels used for AC identification and the MTU setting. Additionally, the command displays the label stack used to transport the actual L2 frames.

```
Rack1R6#show mpls l2transport vc detail
Local interface: Fa0/1 up, line protocol up, Ethernet up
  Destination address: 150.1.5.5, VC ID: 100, VC status: up
    Output interface: Fa0/0.146, imposed label stack {16 40}
    Preferred path: not configured
    Default path: active
    Next hop: 155.1.146.4
  Create time: 00:00:22, last status change time: 00:00:15
  Signaling protocol: LDP, peer 150.1.5.5:0 up
    MPLS VC labels: local 27, remote 40
    Group ID: local 0, remote 0
    MTU: local 1500, remote 1500
    Remote interface description:
  Sequencing: receive disabled, send disabled
  VC statistics:
    packet totals: receive 896, send 28
    byte totals:   receive 53918, send 2520
    packet drops:  receive 0, seq error 0, send 0
```

> **Note**
>
> Now configure SW3 and SW4 as the CE devices for the L2 VPN. Assign them any IP addresses on the same subnet and confirm that you can ping across:

```
SW4:
interface FastEthernet 0/6
 no switchport
 ip address 56.56.56.10 255.255.255.0

SW3:
interface FastEthernet 0/5
 no switchport
 ip address 56.56.56.9 255.255.255.0

Rack1SW3#ping 56.56.56.10

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 56.56.56.10, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 33/35/42 ms
```

Copyright © 2010 Internetwork Expert

14.15 L2TPV3

- Modify the solution for the previous task to use L2TPv3 instead of MPLS as the underlying transport.
- Ensure that the packets are never fragmented in the core and automatic MTU detection is in progress.

Configuration

Note

L2TPv3 is an alternative to MPLS for dynamic tunneling. It is based on the L2TPv2 protocol, which was used for PPP tunneling in dial VPDNs. L2TPv3 does not require an MPLS layer be deployed in the network, as it uses normal IP packets (either IP protocol 115 or UDP packets) to tunnel the payload. This allows L2TPv3 to be used in the networks that lack MPLS support, or in scenarios that require inter-provider operations, where one of the operators does not implement MPLS.

The deployment simplicity comes at a price, of course. First, L2TPv3 has significantly larger overhead than MPLS-based tunneling. This might raise concerns related to the maximum MTU settings of the underlying transport network. Secondly, L2TPv3 does not allow for effective traffic engineering like MPLS does since L2TPv3 uses a globally significant "label" – encapsulating IP packet whereas MPLS implements locally significant tagging. .

Configuring L2TPv3 pseudowire is similar to setting up an AToM pseudowire, it just has more options. First of all, it is mandatory to specify a pseudowire class with the xconnect command that uses L2TPv3 encapsulation. This is due to the fact that you need to specify the source IP address used for L2TPv3 session via the pw-command **`ip local interface <IFNAME>`**. The other important options for L2TPv3 include forcefully setting DF-bit using the pw-class command **`ip dfbit set`**, which avoids in-core fragmentation and performance degradation. By default, the DF-bit is not set for tunneled packets. If you want to copy the TOS byte from encapsulated packets (if they are IP) use the command **`ip tos reflect`** or the command **`ip tos <VALUE>`** to manually set this TOS byte. One other feature is packet sequencing, a feature which allows the dropping of out of sequence packets but adds additional overhead. It is enabled using the command **`sequencing`**, and usually is needed only with the applications that are sensitive to packet reordering, such as SANs.

Copyright © 2010 Internetwork Expert

Normally, when using L2TPv3 you may want to increase the core network MTU by a value that is big enough to accommodate the client network MTU plus the L2TPv3 overhead. In situations where you cannot do that, you may have to implement some solutions at the client side to avoid PMTU problems, such as using the IOS command **ip tcp adjust-mss**. One L2TPv3 command that may help you with customer PMTU discovery is **ip local pmtu** which allows the encapsulating PE to forward back to the customer ICMP unreachable messages thus informing the IP-capable end-devices of fragmentation issues. This command should always be used in conjunction with **ip dfbit set** to ensure that all packets are subject to drop if they cannot be fragmented.

Similar to AToM, L2TPv3 supports port-mode and VLAN-mode VC types, but unlike the MPLS-based technology, it does not require the MTU value to match in order to bring up the pseudowire.

R5:

```
pseudowire-class L2TPV3
 encapsulation l2tpv3
 ip local interface Loopback0
 ip pmtu
 ip dfbit set
 ip tos reflect
!
default interface FastEthernet 0/1
interface FastEthernet 0/1
 xconnect 150.1.6.6 100 encapsulation l2tpv3 pw-class L2TPV3
```

R6:

```
pseudowire-class L2TPV3
 encapsulation l2tpv3
 ip local interface Loopback0
 ip pmtu
 ip dfbit set
 ip tos reflect
!
default interface FastEthernet 0/1
interface FastEthernet 0/1
 xconnect 150.1.5.5 100 encapsulation l2tpv3 pw-class L2TPV3
```

Copyright © 2010 Internetwork Expert

Verification

Note

Use the show command below on the PE routers. Pay special attention to the Session UP and Circuit UP messages which signal a healthy P2P connection. Typical problems with connection establishment could include unreachable endpoint IP addresses, a shutdown interface, mismatching parameters or mismatching VC-types.

Notice the FW header information which is the dump of the encapsulating L2TPv3 header. You may use this information to adjust the core network MTU sizes properly.

Copyright © 2010 Internetwork Expert

```
Rack1R5#show l2tp session all

L2TP Session Information Total tunnels 1 sessions 1

Session id 19547 is up, tunnel id 38503
  Remote session id is 55660, remote tunnel id 34507
  Locally initiated session
Call serial number is 1694600001
Remote tunnel name is Rack1R6
  Internet address is 150.1.6.6
Local tunnel name is Rack1R5
  Internet address is 150.1.5.5
IP protocol 115
  Session is L2TP signaled
  Session state is established, time since change 00:00:35
    4 Packets sent, 6 received
    562 Bytes sent, 923 received
  Last clearing of counters never
  Counters, ignoring last clear:
    4 Packets sent, 6 received
    562 Bytes sent, 923 received
    Receive packets dropped:
      out-of-order:             0
      total:                    0
    Send packets dropped:
      exceeded session MTU:     0
      total:                    0
  DF bit on, ToS reflect enabled, ToS value 0, TTL value 255
  Sending UDP checksums are disabled
  Receiving UDP checksums are verified
  Session PMTU enabled, path MTU is not known
  No session cookie information available
  FS cached header information:
    encap size = 24 bytes
    45000014 00004000 FF734469 96010505
    96010606 0000D96C
  Sequencing is off
  Conditional debugging is disabled
  SSM switch id is 4098, SSM segment id is 8196
  Unique ID is 5
Session Layer 2 circuit, type is Ethernet, name is FastEthernet0/1
  Session vcid is 100
  Circuit state is UP
    Local circuit state is UP
    Remote circuit state is UP
```

Copyright © 2010 Internetwork Expert

Note

Now configure SW3 and SW4 as the CE devices for the L2 VPN. Assign them any IP addresses on the same subnet and confirm that you can ping across:

```
SW4:
interface FastEthernet 0/6
 no switchport
 ip address 56.56.56.10 255.255.255.0

SW3:
interface FastEthernet 0/5
 no switchport
 ip address 56.56.56.9 255.255.255.0

Rack1SW3#ping 56.56.56.10

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 56.56.56.10, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 33/35/42 ms
```

Note

Verify that Path MTU discovery is working for the pseudowire. Make SW3 send large packets with DF-bit set. Notice that the PE router is sending ICMP unreachable messages to the source IP address found in the encapsulated packets, allowing SW3 to properly implement PMTU discovery.

```
Rack1SW3#debug ip icmp
ICMP packet debugging is on

Rack1SW3#ping 56.56.56.10 size 1540 df-bit

Type escape sequence to abort.
Sending 5, 1540-byte ICMP Echos to 56.56.56.10, timeout is 2 seconds:
Packet sent with the DF bit set

ICMP: dst (56.56.56.10): frag. needed and DF set.
ICMP: time exceeded rcvd from 56.56.56.10
ICMP: dst (56.56.56.10): frag. needed and DF set.
ICMP: dst (56.56.56.10): frag. needed and DF set.
ICMP: dst (56.56.56.10): frag. needed and DF set.
ICMP: dst (56.56.56.10): frag. needed and DF set.
Success rate is 0 percent (0/5)
```

Copyright © 2010 Internetwork Expert

Note

Notice that the above procedure also permitted R5 to correctly discover Path MTU. This could be seen in the output of the same command we used before:

```
Rack1R5#show l2tp session all

L2TP Session Information Total tunnels 1 sessions 1

Session id 19547 is up, tunnel id 38503
  Remote session id is 55660, remote tunnel id 34507
  Locally initiated session
Call serial number is 1694600001
Remote tunnel name is Rack1R6
  Internet address is 150.1.6.6
Local tunnel name is Rack1R5
  Internet address is 150.1.5.5
IP protocol 115
  Session is L2TP signaled
  Session state is established, time since change 00:09:16
    114 Packets sent, 130 received
    31488 Bytes sent, 34186 received
  Last clearing of counters never
  Counters, ignoring last clear:
    114 Packets sent, 130 received
    31488 Bytes sent, 34186 received
    Receive packets dropped:
      out-of-order:              0
      total:                     0
    Send packets dropped:
      exceeded session MTU:      0
      total:                     0
  DF bit on, ToS reflect enabled, ToS value 0, TTL value 255
  Sending UDP checksums are disabled
  Receiving UDP checksums are verified
  Session PMTU enabled, path MTU is 1496 bytes
  No session cookie information available
<snip>
```

Copyright © 2010 Internetwork Expert

14.16 MPLS VPN Performance Tuning

- Configure the PE and P routers to minimize the amount of time needed to propagate topology changes between CE-sites.

Configuration

Note

MPLS VPNs utilize MP-BGP as the transport for IGP routing updates between VPN sites. BGP was designed as a highly-scalable protocol, this however limits its ability to quickly propagate changes in the network topology. One of the reasons for this is the goal to maintain network stability, which, of course, slows convergence. Thus, it is not uncommon to see network changes at one site take a minute or even more to propagate to other sites, even with "fast" protocols such as OSPF and EIGRP. The following factors affect the amount of time it takes to propagate a topology change from one PE to another across the MP-BGP core:

1) The time it takes for the IGP update to be redistributed into BGP. In the past this was limited by **`bgp scan-interval`** general scanning interval, but recent IOS versions made this feature event driven. This makes IGP to BGP route redistribution almost instant.

2) The time it takes the local BGP speaker and other BGP speakers to propagate updates to their peers. This time is controlled by the timer known as the peer advertisement-interval. This interval specifies the periodic "batching" of events: BGP waits for the timer to expire and accumulates the updates instead of sending every one immediately. The purpose of this is to prevent a constant load on the peer's CPU for best-path re-calculations. The default interval is 5 seconds, and could be set as low as 0 seconds using the command **`neighbor <IP> advertisement-interval`**.

3) The time it takes the PE router's BGP process to import the MP-BGP VPNv4 prefixes into the local VRF table. This timer is controlled by the command **`bgp scan-time import <5-60>`** which should be entered under the VPNv4 address-family configuration. The default value is 15 seconds, and setting it to 5 seconds may significantly decrease the update propagation time.

The above discussed commands could be tuned to their minimal values, resulting in a change in propagation time between 5-10 seconds. This is probably the best you can get with MPLS VPNs – everything else depends on IGP convergence. Also notice that by tuning these parameters you do not improve the time it takes to detect link or router failures in the MPLS core.

Copyright © 2010 Internetwork Expert

Configuration

```
R4:
router bgp 100
 address-family vpnv4 unicast
 neighbor 150.1.5.5 advertisement-interval 0
 neighbor 150.1.6.6 advertisement-interval 0
 bgp scan import 5

R5 & R6:
router bgp 100
 address-family vpnv4 unicast
 neighbor 150.1.4.4 advertisement-interval 0
 bgp scan import 5
```

Copyright © 2010 Internetwork Expert

Verification

> **Note**
>
> Check the optimized BGP timers on one of the routers. The other routers configuration is verified similarly.

```
Rack1R5#show bgp vpnv4 unicast all neighbors 150.1.4.4  | inc adv
    Route refresh: advertised and received(new)
    New ASN Capability: advertised and received
    Address family VPNv4 Unicast: advertised and received
  Default minimum time between advertisement runs is 0 seconds
```

> **Note**
>
> To validate the import timer, run the following debugging command and notice the timestamps increments:

```
Rack1R5#debug ip bgp events
BGP events debugging is on

Rack1R5#
23:27:41: BGP(4): Import timer expired. Walking from 55 to 55
Rack1R5#
23:27:46: BGP(4): Import timer expired. Walking from 55 to 55
Rack1R5#
23:27:51: BGP(4): Import timer expired. Walking from 55 to 55
```

Copyright © 2010 Internetwork Expert