CCIE Routing and Switching Official Exam Certification Guide

Second Edition

Wendell Odom, CCIE No. 1624
Contributing Authors: Jim Geier and Naren Mehta

Cisco Press
800 East 96th Street
Indianapolis, Indiana 46240 USA

CCIE Routing and Switching Official Exam Certification Guide, Second Edition

Wendell Odom, CCIE No. 1624
Contributing authors: Jim Geier and Naren Mehta

Copyright © 2006 Cisco Systems, Inc.

Cisco Press logo is a trademark of Cisco Systems, Inc.

Published by:
Cisco Press
800 East 96th Street
Indianapolis, IN 46240 USA

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage and retrieval system, without written permission from the publisher, except for the inclusion of brief quotations in a review.

Printed in the United States of America 1 2 3 4 5 6 7 8 9 0

First Printing February 2006

Library of Congress Cataloging-in-Publication Number: 2004113160

ISBN: 1-58720-141-0

Trademark Acknowledgments

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Cisco Press or Cisco Systems, Inc. cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Warning and Disclaimer

This book is designed to provide information about the Cisco CCIE Routing and Switching Written Exam, No. 350-001. Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied.

The information is provided on an “as is” basis. The authors, Cisco Press, and Cisco Systems, Inc. shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the discs or programs that may accompany it.

The opinions expressed in this book belong to the author and are not necessarily those of Cisco Systems, Inc.

Corporate and Government Sales

Cisco Press offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales.

For more information please contact: **U.S. Corporate and Government Sales** 1-800-382-3419 corpsales@pearsontechgroup.com

For sales outside the U.S. please contact: **International Sales** international@pearsoned.com

Feedback Information

At Cisco Press, our goal is to create in-depth technical books of the highest quality and value. Each book is crafted with care and precision, undergoing rigorous development that involves the unique expertise of members from the professional technical community.

Readers’ feedback is a natural continuation of this process. If you have any comments regarding how we could improve the quality of this book, or otherwise alter it to better suit your needs, you can contact us through e-mail at feedback@ciscopress.com. Please make sure to include the book title and ISBN in your message.

We greatly appreciate your assistance.

Publisher	John Wait
Editor-in-Chief	John Kane
Cisco Representative	Anthony Wolfenden
Cisco Press Program Manager	Jeff Brady
Executive Editor	Brett Bartow
Production Manager	Patrick Kanouse
Senior Development Editor	Christopher Cleveland
Copy Editor	Bill McManus
Technical Editors	Maurilio Gorito, Rus Healy, Paul Negron, William R. Parkhurst
Editorial Assistant	Raina Han
Cover and Book Designer	Louisa Adair
Composition	Interactive Composition Corporation
Indexer	Lisa Stumpf

CISCO SYSTEMS

Corporate Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 526-4100

European Headquarters
Cisco Systems International BV
Haarlerbergpark
Haarlerbergweg 13-19
1101 CH Amsterdam
The Netherlands
www-europe.cisco.com
Tel: 31 0 20 357 1000
Fax: 31 0 20 357 1100

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
www.cisco.com
Tel: 408 526-7660
Fax: 408 527-0883

Asia Pacific Headquarters
Cisco Systems, Inc.
Capital Tower
168 Robinson Road
#22-01 to #29-01
Singapore 068912
www.cisco.com
Tel: +65 6317 7777
Fax: +65 6317 7799

Cisco Systems has more than 200 offices in the following countries and regions. Addresses, phone numbers, and fax numbers are listed on the **Cisco.com Web site at www.cisco.com/go/offices.**

Argentina • Australia • Austria • Belgium • Brazil • Bulgaria • Canada • Chile • China PRC • Colombia • Costa Rica • Croatia • Czech Republic Denmark • Dubai, UAE • Finland • France • Germany • Greece • Hong Kong SAR • Hungary • India • Indonesia • Ireland • Israel • Italy Japan • Korea • Luxembourg • Malaysia • Mexico • The Netherlands • New Zealand • Norway • Peru • Philippines • Poland • Portugal Puerto Rico • Romania • Russia • Saudi Arabia • Scotland • Singapore • Slovakia • Slovenia • South Africa • Spain • Sweden Switzerland • Taiwan • Thailand • Turkey • Ukraine • United Kingdom • United States • Venezuela • Vietnam • Zimbabwe

Copyright © 2003 Cisco Systems, Inc. All rights reserved. CCIP, CCSP, the Cisco Arrow logo, the Cisco *Powered* Network mark, the Cisco Systems Verified logo, Cisco Unity, Follow Me Browsing, FormShare, iQ Net Readiness Scorecard, Networking Academy, and ScriptShare are trademarks of Cisco Systems, Inc.; Changing the Way We Work, Live, Play, and Learn, The Fastest Way to Increase Your Internet Quotient, and iQuick Study are service marks of Cisco Systems, Inc.; and Aironet, ASIST, BPX, Catalyst, CCDA, CCDP, CCIE, CCNA, CCNP, Cisco, the Cisco Certified Internetwork Expert logo, Cisco IOS, the Cisco IOS logo, Cisco Press, Cisco Systems, Cisco Systems Capital, the Cisco Systems logo, Empowering the Internet Generation, Enterprise/Solver, EtherChannel, EtherSwitch, Fast Step, GigaStack, Internet Quotient, IOS, IP/TV, iQ Expertise, the iQ logo, LightStream, MGX, MICA, the Networkers logo, Network Registrar, *Packet*, PIX, Post-Routing, Pre-Routing, RateMUX, Registrar, SlideCast, SMARTnet, StrataView Plus, Stratm, SwitchProbe, TeleRouter, TransPath, and VCO are registered trademarks of Cisco Systems, Inc. and/or its affiliates in the U.S. and certain other countries.

All other trademarks mentioned in this document or Web site are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (0303R)

Printed in the USA

About the Author

Wendell Odom, CCIE No. 1624, is a senior instructor with Skyline Advanced Technology Services (http://www.skyline-ats.com), where he teaches the QOS, CCIE, and SAN courses. Wendell has worked in the networking arena for 20 years, with jobs in pre- and post-sales technical consulting, teaching, and course development. He has authored several Cisco Press books, including the best-selling *CCNA INTRO* and *ICND Exam Certification Guides*, the *Cisco QoS Exam Certification Guide*, and the introductory text *Computer Networking First-Step*.

About the Contributing Authors

Jim Geier, author of Chapters 22 and 23, is the founder of Wireless-Nets, Ltd., (www.wireless-nets.com) and the company's principal consultant. His 25 years of experience includes the analysis, design, development, installation, and support of numerous wired and wireless network systems for cities, enterprises, airports, retail stores, manufacturing facilities, warehouses, and hospitals throughout the world. Under Wireless-Nets, Ltd., Jim founded the Independent Wireless Networking Academy (www.wirelessnetworkingacademy.com) to prepare people for working in the wireless networking industry. He has instructed hundreds of people on designing and deploying wireless LAN solutions.

Jim is the author of several books, including *Wireless LANs: Implementing Interoperable Networks*, Second Edition (SAMS), *Certified Wireless Analysis Professional—Official Study Guide* (McGraw-Hill), *Wireless Networks First-Step* (Cisco Press), *Wireless Networking Handbook* (Macmillan), and *Network Reengineering* (McGraw-Hill). Jim is the author of hundreds of articles for industry magazines and online publications, such as *Network Computing*, *Network World*, Wi-FiPlanet.com, and Mobilepipeline.com. He has been writing about computer networking topics, with emphasis on wireless systems, for the past 12 years. He is currently the editor-in-chief and regular contributor of mobilizedsoftware.com, an online publication providing education to enterprises involved with implementing mobile wireless systems.

Jim has been an active member of the IEEE 802.11 Working Group, developing international standards for wireless LANs. He has also been an active member within the Wi-Fi Alliance, responsible for certifying interoperability of 802.11 ("Wi-Fi") wireless LANs. He served as Chairman of the IEEE Computer Society, Dayton Section, and Chairman of the IEEE International Conference on Wireless LAN Implementation. Jim is an advisory board member of several leading wireless LAN companies and an advisor for the Certified Wireless Network Professional (CWNP) independent certification program for people deploying wireless LANs.

Jim's education includes a bachelor's and master's degree in electrical engineering and a master's degree in business administration.

Contact Jim at jimgeier@wireless-nets.com.

Naren Mehta, CCIE No. 9797 (Routing and Switching, Security), author of Chapters 19 and 20, is a senior partner and director of training for an internationally known training and consulting company that specializes in providing customized, one-to-one training, for CCIE lab students and consulting for Cisco networks. Naren has been in the training and consulting field for the past 15 years and teaching Cisco certification courses ranging from CCNA to CCIE (written and lab) for the past 7 years. His experience includes the analysis, design, installation, training, and support for various Cisco networks for financial, manufacturing, utility, and healthcare industries. His specialty is explaining complex concepts in such a way that it becomes easier for anybody to understand them. Naren has been a source of inspiration, motivation, and encouragement for many of his students who wanted to pursue their CCIE lab certification and helped them pass their CCIE Routing and Switching and Security lab certification exams. He has an MBA in marketing and finance, an MS in industrial engineering, and a BS in mechanical engineering.

About the Technical Reviewers

Maurilio Gorito, CCIE No. 3807, works for Cisco Systems, Inc., as part of the CCIE team. As content manager, Maurilio is responsible for managing the content development process for the CCIE Routing and Switching lab and written exams and proctoring the CCIE Routing and Switching, Service Provider, and CCIE Security lab exams.

Rus Healy, CCIE No. 15025, is program manager for Technical Training and Certifications for Microwave Data Systems in Rochester, New York, a leading manufacturer of data radios for industrial and public-safety applications. His other interests include bicycling, woodworking, and camping with his family. Rus completed his CCIE Routing and Switching certification while working on this book. He also holds a CCDP and three Microwave Data Systems technical certifications. He lives in the Finger Lakes region of western New York with his wife, Nancy, and their two children, Gwen and Trevor.

Paul Negron, CCIE No. 14856, has been involved with networking technologies for 13 years. He has been a senior instructor for Skyline Advanced Technical Services for the past 5 years. He has been involved with the designing of core network services for a number of service providers. He currently instructs all the CCIP level courses as well as the Advanced BGP, MPLS, and QOS courses. Paul has six years experience with Satellite Communications as well as six years with Cisco platforms. Paul holds several Cisco certifications, including CCIE Service Provider.

William R. Parkhurst, Ph.D., CCIE No. 2969, is a design consultant with Cisco Systems specializing in IP core and mobile wireless networks. Before his current position, Bill was on the CCIE team and managed the development of the CCIE Service Provider and Voice tracks. Bill holds a Ph.D. in electrical and computer engineering from Wichita State University and a bachelor's degree in political science from the University of Maryland. Bill is the author of three Cisco Press books: *Routing First-Step*, *Cisco OSPF Command and Configuration Handbook*, and *Cisco BGP-4 Command and Configuration Handbook*.

Dedication

For Lavinnie Viola McCoy Odom, aka Granny, Aunt Bill, and cousin “it.” Thanks for the hugs, prayers, late-night fried egg sandwiches, and sheets warmed by the heater in the dead of winter. 1914–2004.

Acknowledgments

Setting out to write a CCIE-level book can be a bit intimidating. However, having the right set of technical editors has made the process much less difficult and has made the text much better. Maurilio and Bill provided considerable technical input, as well as providing unique insights based on their roles inside Cisco's CCIE program. Rus did a great job of helping us keep the right depth to meet a typical candidate, as he had just passed his CCIE Routing and Switching written exam as we started this project, and just completed his Routing and Switching lab by conclusion of the book. (Congrats, Rus!) And thanks to Paulie for jumping in to help with the tech edits later in the writing process. Together they made many valuable comments that improved the book.

The production team, headed by Patrick Kanouse, did their usual excellent job. Like the "behind the scenes" people in many businesses, their specific efforts may not be obvious to the public, but it's no less appreciated by me. Thanks for cleaning up my Southern English, drawing nice figures from my beautiful PowerPoints, and putting up with my repetitive, "That's what I asked for, but I changed my mind—can I make one more change?" e-mails. You folks make me look good on paper—if only you could be in charge of my wardrobe too, I'd look good all the time!

Brett Bartow, executive editor, did his usual New-York-Yankees-like job in helping steer this project to completion. In between talking about sports, Brett worked through the many changes in direction with this book, and helped guide us to the right product. He found Jim and Naren, who were vital to the process as well. And yes, so the whole world knows, he did win his fantasy baseball league in 2005—again proving he's a really smart guy.

Chris Cleveland developed this book, which means he got to see the rawest form of the materials, and multiple times. Chris continues to be simply the best in the business—You da man, Chris C!

Jim Geier and Naren Mehta came through by writing two of the nine parts of the text. Jim, an accomplished author with wireless technologies, did his usual wonderful job on the wireless chapters of the book. Thanks, Jim, for adding your depth of knowledge to my weakest area for this book! Naren brought a great depth of experience to his multicast chapters, as he spends most days teaching CCIE lab prep courses. I expect to see some good CCIE-level books from Naren in the future!

On the personal side, my wife Kris gets big praise for just being her usual wonderful self even when things get a little tough with the writing schedule. I could not do it without you doll! And finally, without the daily grace and mercy from Jesus, none of these books would ever be possible—thanks for watching over every little hair on my head.

This Book Is Safari Enabled

The Safari® Enabled icon on the cover of your favorite technology book means the book is available through Safari Bookshelf. When you buy this book, you get free access to the online edition for 45 days.

Safari Bookshelf is an electronic reference library that lets you easily search thousands of technical books, find code samples, download chapters, and access technical information whenever and wherever you need it.

To gain 45-day Safari Enabled access to this book:

- Go to http://www.ciscopress.com/safarienabled
- Complete the brief registration form
- Enter the coupon code T6H4-5DXH-4KC2-I9HE-AJF6

If you have difficulty registering on Safari Bookshelf or accessing the online edition, please e-mail customer-service@safaribooksonline.com.

Contents at a Glance

Introduction xxxi

Part I Bridging and LAN Switching 3

Chapter 1 Ethernet Basics 5

Chapter 2 Virtual LANs and VLAN Trunking 27

Chapter 3 Spanning Tree Protocol 57

Part II TCP/IP 89

Chapter 4 IP Addressing 91

Chapter 5 IP Services 131

Chapter 6 TCP/IP Transport and Application Services 151

Part III IP Routing 171

Chapter 7 IP Forwarding (Routing) 173

Chapter 8 RIP Version 2 201

Chapter 9 EIGRP 229

Chapter 10 OSPF 255

Chapter 11 IGP Route Redistribution, Route Summarization, and Default Routing 313

Chapter 12 Fundamental BGP Operations 355

Chapter 13 BGP Routing Policies 417

Part IV Quality of Service 483

Chapter 14 Classification and Marking 485

Chapter 15 Congestion Management and Avoidance 515

Chapter 16 Shaping and Policing 551

Part V WAN 587

Chapter 17 Synchronous Serial Links and Protocols 589

Chapter 18 Frame Relay 607

Part VI IP Multicast 627

Chapter 19 Introduction to IP Multicasting 629

Chapter 20 IP Multicast Routing 679

Part VII Security 739

Chapter 21 Security 741

Part VIII Enterprise Wireless Mobility 783

Chapter 22 IEEE 802.11 Fundamentals 785

Chapter 23 Wireless LAN Solutions 825

Part IX OSI and Cisco Device Basics 847

Chapter 24 Miscellaneous Networking Theory and Practices 849

Part X Appendixes 865

Appendix A Answers to the “Do I Know This Already?” Quizzes 867

Appendix B CCIE Routing and Switching Exam Updates: Version 1.0 891

Appendix C MPLS 895

Appendix D Decimal to Binary Conversion Table 953

Glossary 959

Index 1012

Contents

Introduction xxxi

Part I Bridging and LAN Switching 2

Chapter 1 Ethernet Basics 5

"Do I Know This Already?" Quiz 5

Foundation Topics 8

Ethernet Layer 1: Wiring, Speed, and Duplex 8

RJ-45 Pinouts and Category 5 Wiring 8

Auto-negotiation, Speed, and Duplex 9

CSMA/CD 10

Collision Domains and Switch Buffering 10

Basic Switch Port Configuration 12

Ethernet Layer 2: Framing and Addressing 14

Types of Ethernet Addresses 16

Ethernet Address Formats 17

Protocol Types and the 802.3 Length Field 18

Switching and Bridging Logic 19

Foundation Summary *25*

Memory Builders 25

Fill in Key Tables from Memory 25

Definitions 25

Further Reading 25

Chapter 2 Virtual LANs and VLAN Trunking 27

"Do I Know This Already?" Quiz 27

Foundation Topics 31

Virtual LANs 31

VLAN Configuration 31

Using VLAN Database Mode to Create VLANs 32

Using Configuration Mode to Put Interfaces into VLANs 34

Using Configuration Mode to Create VLANs 35

Private VLANs 36

VLAN Trunking Protocol 38

VTP Process and Revision Numbers 39

VTP Configuration 40

Normal-Range and Extended-Range VLANs 42

Storing VLAN Configuration 43

VLAN Trunking: ISL and 802.1Q 44

ISL and 802.1Q Concepts 44

ISL and 802.1Q Configuration 45

Allowed, Active, and Pruned VLANs 48

Trunk Configuration Compatibility 48

Configuring Trunking on Routers 49
802.1Q-in-Q Tunneling 51
Foundation Summary 53
Memory Builders 54
Fill in Key Tables from Memory 54
Definitions 54
Further Reading 55

Chapter 3 Spanning Tree Protocol 57

"Do I Know This Already?" Quiz 57
Foundation Topics 61
802.1D Spanning Tree Protocol 61
Choosing Which Ports Forward: Choosing Root Ports and Designated Ports 61
Electing a Root Switch 61
Determining the Root Port 63
Determining the Designated Port 64
Converging to a New STP Topology 65
Topology Change Notification and Updating the CAM 66
Transitioning from Blocking to Forwarding 67
Per-VLAN Spanning Tree and STP over Trunks 68
STP Configuration and Analysis 70
Optimizing Spanning Tree 73
PortFast, UplinkFast, and BackboneFast 73
PortFast 74
UplinkFast 74
BackboneFast 75
PortFast, UplinkFast, and BackboneFast Configuration 75
PortChannels 76
Load Balancing Across PortChannels 76
PortChannel Discovery and Configuration 77
Rapid Spanning Tree Protocol 78
Multiple Spanning Trees: IEEE 802.1s 80
Protecting STP 82
Root Guard and BPDU Guard: Protecting Access Ports 82
UDLD and Loop Guard: Protecting Trunks 83
Foundation Summary 85
Memory Builders 87
Fill in Key Tables from Memory 87
Definitions 87
Further Reading 87

Part II TCP/IP 89

Chapter 4 IP Addressing 91

"Do I Know This Already?" Quiz 91
Foundation Topics 94

IP Addressing and Subnetting 94
IP Addressing and Subnetting Review 94
Subnetting a Classful Network Number 95
Comments on Classless Addressing 97
Subnetting Math 97
Dissecting the Component Parts of an IP Address 97
Finding Subnet Numbers and Valid Range of IP Addresses—Binary 98
Decimal Shortcuts to Find the Subnet Number and Valid Range of IP Addresses 99
Determining All Subnets of a Network—Binary 102
Determining All Subnets of a Network—Decimal 104
VLSM Subnet Allocation 105
Route Summarization Concepts 107
Finding Inclusive Summary Routes—Binary 108
Finding Inclusive Summary Routes—Decimal 109
Finding Exclusive Summary Routes—Binary 110
CIDR, Private Addresses, and NAT 111
Classless Interdomain Routing 111
Private Addressing 113
Network Address Translation 113
Static NAT 115
Dynamic NAT Without PAT 116
Overloading NAT with Port Address Translation 117
Dynamic NAT and PAT Configuration 118
IP Version 6 119
IPv6 Address Formats 120
Aggregatable Global Unicast Addresses 120
Simple IPv6 Configuration 121
IPv6 Addressing Summary 123
Foundation Summary 125
Memory Builders 128
Fill in Key Tables from Memory 128
Definitions 129
Further Reading 129

Chapter 5 IP Services 131
"Do I Know This Already?" Quiz 131
Foundation Topics 134
ICMP 134
ICMP Unreachable 135
Time Exceeded ICMP Message 136
ICMP Redirect 137
ARP, Proxy ARP, Reverse ARP, BOOTP, and DHCP 137
ARP and Proxy ARP 137
RARP, BOOTP, and DHCP 139

HSRP, VRRP, and GLBP 141
Network Time Protocol 143
Foundation Summary 146
Memory Builders 148
Fill in Key Tables from Memory 148
Definitions 148
Further Reading 149

Chapter 6 TCP/IP Transport and Application Services 151
"Do I Know This Already?" Quiz 151
Foundation Topics 154
TCP and UDP 154
TCP Connections and Port Numbers 155
TCP Error Recovery 157
TCP Dynamic Windowing 157
TCP Header Miscellany 159
TCP/IP Applications 160
Passive and Active Mode FTP 161
Application Authentication and Privacy 163
Network Management and SNMP 163
SNMP Protocol Messages 165
SNMP MIBs 166
SNMP Security 167
Foundation Summary 168
Memory Builders 168
Fill in Key Tables from Memory 168
Definitions 169
Further Reading 169

Part III IP Routing 171

Chapter 7 IP Forwarding (Routing) 173
"Do I Know This Already?" Quiz 173
Foundation Topics 177
IP Forwarding 177
Process Switching, Fast Switching, and Cisco Express Forwarding 178
Building Adjacency Information: ARP and Inverse ARP 179
Frame Relay Inverse ARP 180
Static Configuration of Frame Relay Mapping Information 183
Disabling InARP 184
Classless and Classful Routing 185
Multilayer Switching 186
MLS Logic 186
Using Routed Ports and PortChannels with MLS 187
MLS Configuration 188

Policy Routing 191
Foundation Summary 197
Memory Builders 198
Fill in Key Tables from Memory 198
Definitions 199
Further Reading 199

Chapter 8 RIP Version 2 201
"Do I Know This Already?" Quiz 201
Foundation Topics 204
RIP Version 2 Basics 204
RIP Convergence and Loop Prevention 205
Converged Steady-State Operation 206
Triggered (Flash) Updates and Poisoned Routes 208
RIP Convergence When Routing Updates Cease 210
Convergence Extras 212
RIP Configuration 213
Enabling RIP and the Effects of Autosummarization 214
RIP Authentication Configuration 216
RIP Next-Hop Feature and Split Horizon 219
RIP Offset Lists 220
Route Filtering with Distribute Lists and Prefix Lists 222
Foundation Summary 225
Memory Builders 227
Fill in Key Tables from Memory 227
Definitions 227
Further Reading 227

Chapter 9 EIGRP 229
"Do I Know This Already?" Quiz 229
Foundation Topics 233
EIGRP Basics and Steady-State Operation 233
Hellos, Neighbors, and Adjacencies 233
EIGRP Updates 236
The EIGRP Topology Table 238
EIGRP Convergence 240
Input Events and Local Computation 241
Going Active on a Route 243
Stuck-in-Active 245
Limiting Query Scope 246
EIGRP Configuration 246
EIGRP Configuration Example 246
EIGRP Load Balancing 249
EIGRP Configuration Options That Are Similar to RIP 250

Foundation Summary 251
Memory Builders 253
Fill in Key Tables from Memory 253
Definitions 253
Further Reading 253

Chapter 10 OSPF 255
"Do I Know This Already?" Quiz 255
Foundation Topics 260
OSPF Database Exchange 260
OSPF Router IDs 260
Becoming Neighbors, Exchanging Databases, and Becoming Adjacent 261
Becoming Neighbors: The Hello Process 263
Flooding LSA Headers to Neighbors 264
Requesting, Getting, and Acknowledging LSAs 265
Designated Routers on LANs 266
Designated Router Optimization on LANs 266
DR Election on LANs 268
Designated Routers on WANs and OSPF Network Types 269
Caveats Regarding OSPF Network Types over NBMA Networks 270
Example of OSPF Network Types and NBMA 271
SPF Calculation 274
Steady-State Operation 275
OSPF Design and LSAs 275
OSPF Design Terms 276
LSA Types and Network Types 277
LSA Types 1 and 2 278
LSA Type 3 and Inter-Area Costs 281
LSA Types 4 and 5, and External Route Types 1 and 2 284
OSPF Design in Light of LSA Types 286
Stubby Areas 287
OSPF Configuration 290
OSPF Costs and Clearing the OSPF Process 292
Alternatives to the OSPF Network Command 295
OSPF Filtering 295
Filtering Routes Using the **distribute-list** *Command 295*
OSPF ABR LSA Type 3 Filtering 297
Filtering Type 3 LSAs with the **area range** *Command 299*
Virtual Link Configuration 299
Configuring OSPF Authentication 301
OSPF Stub Router Configuration 303
Foundation Summary 305
Memory Builders 310
Fill in Key Tables from Memory 310
Definitions 310
Further Reading 311

Chapter 11 IGP Route Redistribution, Route Summarization, and Default Routing 313
"Do I Know This Already?" Quiz 313
Foundation Topics 317
Route Maps, Prefix Lists, and Administrative Distance 317
Configuring Route Maps with the **route-map** *Command 317*
Route Map **match** *Commands for Route Redistribution 319*
Route Map **set** *Commands for Route Redistribution 320*
IP Prefix Lists 321
Administrative Distance 323
Route Redistribution 324
The Mechanics of the **redistribute** *Command 324*
Redistribution Using Default Settings 325
Setting Metrics, Metric Types, and Tags 328
Redistributing a Subset of Routes Using a Route Map 329
Mutual Redistribution at Multiple Routers 333
Preventing Suboptimal Routes by Setting the Administrative Distance 335
Preventing Suboptimal Routes by Using Route Tags 338
Using Metrics and Metric Types to Influence Redistributed Routes 340
Route Summarization 342
EIGRP Route Summarization 344
OSPF Route Summarization 344
RIP Route Summarization 345
Default Routes 345
Using Static Routes to 0.0.0.0, with **redistribute static** *347*
Using the **default-information originate** Command *348*
Using the **ip default-network** *Command 349*
Using Route Summarization to Create Default Routes 350
Foundation Summary 352
Memory Builders 353
Fill in Key Tables from Memory 353
Definitions 353
Further Reading 353
Chapter 12 Fundamental BGP Operations 355
"Do I Know This Already?" Quiz 355
Foundation Topics 360
Building BGP Neighbor Relationships 361
Internal BGP Neighbors 362
External BGP Neighbors 365
Checks Before Becoming BGP Neighbors 366
BGP Messages and Neighbor States 368
BGP Message Types 368
Purposefully Resetting BGP Peer Connections 369

Building the BGP Table 370
Injecting Routes/Prefixes into the BGP Table 370
The BGP **network** *Command 370*
Redistributing from an IGP, Static, or Connected Route 373
The Impact of Auto-Summary on Redistributed Routes and the **network** *Command 375*
Manual Summaries and the AS_PATH Path Attribute 378
Adding Default Routes to BGP 381
The ORIGIN Path Attribute 382
Advertising BGP Routes to Neighbors 383
The BGP Update Message 383
Determining the Contents of Updates 384
Example: Impact of the Decision Process and NEXT_HOP on BGP Updates 386
Summary of Rules for Routes Advertised in BGP Updates 392
Building the IP Routing Table 392
Adding eBGP Routes to the IP Routing Table 392
Backdoor Routes 393
Adding iBGP Routes to the IP Routing Table 394
Using Sync and Redistributing Routes 396
Disabling Sync and Using BGP on All Routers in an AS 398
Confederations 399
Configuring Confederations 401
Route Reflectors 404
Foundation Summary 410
Memory Builders 414
Fill in Key Tables from Memory 414
Definitions 414
Further Reading 415

Chapter 13 BGP Routing Policies 417
"Do I Know This Already?" Quiz 417
Foundation Topics 423
Route Filtering and Route Summarization 423
Filtering BGP Updates Based on NLRI 424
Route Map Rules for NLRI Filtering 427
Soft Reconfiguration 428
Comparing BGP Prefix Lists, Distribute Lists, and Route Maps 428
Filtering Subnets of a Summary Using the **aggregate-address** *Command 429*
Filtering BGP Updates by Matching the AS_PATH PA 430
The BGP AS_PATH and AS_PATH Segment Types 431
Using Regular Expressions to Match AS_PATH 433
Example: Matching AS_PATHs Using AS_PATH Filters 436
Matching AS_SET and AS_CONFED_SEQ 439
BGP Path Attributes and the BGP Decision Process 442
Generic Terms and Characteristics of BGP PAs 442
The BGP Decision Process 444

Clarifications of the BGP Decision Process 445
Two Final Tiebreaker Steps in the BGP Decision Process 445
Adding Multiple BGP Routes to the IP Routing Table 446
Mnemonics for Memorizing the Decision Process 446
Configuring BGP Policies 448
Background: BGP PAs and Features Used by Routing Policies 448
Step 0: NEXT_HOP Reachable 450
Step 1: Administrative Weight 450
Step 2: Highest Local Preference (LOCAL_PREF) 453
Step 3: Choose Between Locally Injected Routes Based on ORIGIN PA 456
Step 4: Shortest AS_PATH 457
Removing Private ASNs 457
AS_PATH Prepending and Route Aggregation 458
Step 5: Best ORIGIN PA 461
Step 6: Smallest Multi-Exit Discriminator 461
Configuring MED: Single Adjacent AS 463
Configuring MED: Multiple Adjacent Autonomous Systems 464
The Scope of MED 464
Step 7: Prefer Neighbor Type eBGP over iBGP 465
Step 8: Smallest IGP Metric to the NEXT_HOP 465
The maximum-paths Command and BGP Decision Process Tiebreakers 465
Step 9: Lowest BGP Router ID of Advertising Router (with One Exception) 466
Step 10: Lowest Neighbor ID 466
The BGP **maximum-paths** *Command 466*
BGP Communities 468
Matching COMMUNITY with Community Lists 472
Removing COMMUNITY Values 473
Filtering NLRI Using Special COMMUNITY Values 474
Foundation Summary 476
Memory Builders 480
Fill in Key Tables from Memory 480
Definitions 480
Further Reading 480

Part IV Quality of Service 483

Chapter 14 Classification and Marking 485
"Do I Know This Already?" Quiz 485
Foundation Topics 489
Fields That Can Be Marked for QoS Purposes 489
IP Precedence and DSCP Compared 489
DSCP Settings and Terminology 490
The Class Selector PHB and DSCP Values 491
The Assured Forwarding PHB and DSCP Values 491
The Expedited Forwarding PHB and DSCP Values 492

Non-IP Header Marking Fields 493
Ethernet LAN Class of Service 493
WAN Marking Fields 493
Locations for Marking and Matching 494
Cisco Modular QoS CLI 495
The Mechanics of MQC 496
Classification Using Class Maps 497
Using Multiple **match** *Commands 498*
Classification Using NBAR 499
Classification and Marking Tools 500
Class-Based Marking (CB Marking) Configuration 500
CB Marking Example 501
CB Marking of CoS and DSCP 505
Network-Based Application Recognition 507
CB Marking Design Choices 508
Marking Using Policers 509
Policy Routing for Marking 510
Foundation Summary 511
Memory Builders 513
Fill in Key Tables from Memory 513
Definitions 513
Further Reading 513

Chapter 15 Congestion Management and Avoidance 515

"Do I Know This Already?" Quiz 515
Foundation Topics 519
Cisco Router Queuing Concepts 519
Software Queues and Hardware Queues 519
Queuing on Interfaces Versus Subinterfaces and Virtual Circuits 520
Comparing Queuing Tools 520
Queuing Tools: FIFO, PQ, CQ, WFQ, CBWFQ, and LLQ 521
FIFO Queuing 521
Priority Queuing 522
Custom Queuing 523
Weighted Fair Queuing 524
WFQ Scheduler: The Process 525
WFQ Drop Policy, Number of Queues, and Queue Lengths 526
Types of WFQ Queues 527
WFQ Configuration 527
Class-Based WFQ and Low-Latency Queuing 529
CBWFQ Basic Features and Configuration 529
Defining and Limiting CBWFQ Bandwidth 532
Low-Latency Queuing 534
Defining and Limiting LLQ Bandwidth 537

LLQ with More Than One Priority Queue 538
Miscellaneous CBWFQ/LLQ Topics 538
Queuing Summary 538
Weighted Random Early Detection 539
How WRED Weights Packets 541
WRED Configuration 542
LAN Switch Congestion Management and Avoidance 542
Cisco 3550 Switch Egress Queuing 543
Cisco 3550 Congestion Avoidance 545
Comparisons Between Cisco 3550 and 2950 Switches 547
Foundation Summary 549
Memory Builders 549
Fill in Key Tables from Memory 549
Definitions 549
Further Reading 549

Chapter 16 Shaping and Policing 551
"Do I Know This Already?" Quiz 551
Foundation Topics 555
Traffic-Shaping Concepts 555
Shaping Terminology 555
Shaping with an Excess Burst 557
Underlying Mechanics of Shaping 557
Traffic-Shaping Adaptation on Frame Relay Networks 559
Class-Based Shaping Configuration 559
Tuning Shaping for Voice Using LLQ and a Small Tc 561
Configuring Shaping by Bandwidth Percent 564
CB Shaping to a Peak Rate 565
Adaptive Shaping 565
Frame Relay Traffic Shaping Configuration 565
FRTS Configuration Using the **traffic-rate** *Command 567*
Setting FRTS Parameters Explicitly 568
FRTS Configuration Using LLQ 569
FRTS Adaptive Shaping 570
Policing Concepts and Configuration 571
CB Policing Concepts 571
Single-Rate, Two-Color Policing (One Bucket) 571
Single-Rate, Three-Color Policer (Two Buckets) 573
Two-Rate, Three-Color Policer (Two Buckets) 573
Class-Based Policing Configuration 575
Single-Rate, Three-Color Policing of All Traffic 575
Policing a Subset of the Traffic 576
CB Policing Defaults for Bc and Be 577
Configuring Dual-Rate Policing 577

Multi-Action Policing 578
Policing by Percentage 578
Committed Access Rate 579
Foundation Summary 582
Memory Builders 584
Fill in Key Tables from Memory 584
Definitions 584
Further Reading 585

Part V WAN 587

Chapter 17 Synchronous Serial Links and Protocols 589
"Do I Know This Already?" Quiz 589
Foundation Topics 592
Synchronous Serial Links 592
T1 Framing and Encoding 592
T1 Alarms 594
Carrier Detect and Interface Resets 594
Point-to-Point Protocol 595
PPP Link Control Protocol 596
Basic LCP/PPP Configuration 597
Multilink PPP 598
MLP Link Fragmentation and Interleaving 600
PPP Compression 601
PPP Layer 2 Payload Compression 602
Header Compression 602
Foundation Summary 604
Memory Builders 605
Fill in Key Tables from Memory 605
Definitions 605
Further Reading 605

Chapter 18 Frame Relay 607
"Do I Know This Already?" Quiz 607
Foundation Topics 610
Frame Relay Concepts 610
Frame Relay Data Link Connection Identifiers 610
Local Management Interface 611
Frame Relay Headers and Encapsulation 612
Frame Relay Congestion: DE, BECN, and FECN 613
Adaptive Shaping, FECN, and BECN 614
The Discard Eligibility Bit 615
Frame Relay Configuration 615
Frame Relay Configuration Basics 615
Frame Relay Payload Compression 619
Frame Relay Fragmentation 620

Foundation Summary 623
Memory Builders 624
Fill in Key Tables from Memory 624
Definitions 625
Further Reading 625

Part VI IP Multicast 627

Chapter 19 Introduction to IP Multicasting 629
"Do I Know This Already?" Quiz 629
Foundation Topics 632
Why Do You Need Multicasting? 632
Problems with Unicast and Broadcast Methods 632
How Multicasting Provides a Scalable and Manageable Solution 635
Multicast IP Addresses 638
Multicast Address Range and Structure 638
Well-Known Multicast Addresses 638
Multicast Addresses for Permanent Groups 639
Multicast Addresses for Source-Specific Multicast Applications and Protocols 640
Multicast Addresses for GLOP Addressing 640
Multicast Addresses for Private Multicast Domains 640
Multicast Addresses for Transient Groups 641
Summary of Multicast Address Ranges 641
Mapping IP Multicast Addresses to MAC Addresses 642
Managing Distribution of Multicast Traffic with IGMP 643
Joining a Group 644
Internet Group Management Protocol 645
IGMP Version 1 645
IGMPv1 Host Membership Query Functions 646
IGMPv1 Host Membership Report Functions 647
IGMPv1 Leave Mechanism 651
IGMPv1 Querier 651
IGMP Version 2 651
IGMPv2 Leave Group and Group-Specific Query Messages 654
IGMPv2 Querier 656
IGMPv1 and IGMPv2 Interoperability 657
IGMPv2 Host and IGMPv1 Routers 657
IGMPv1 Host and IGMPv2 Routers 658
IGMPv1 and IGMPv2 Routers 658
Timers Used in IGMPv1 and IGMPv2 659
IGMP Version 3 659
Comparison of IGMPv1, IGMPv2, and IGMPv3 661
Multicast Listener Discovery Protocol 662
LAN Multicast Optimizations 662
Cisco Group Management Protocol 663

IGMP Snooping 669
Router-Port Group Management Protocol 673
Foundation Summary 676
Memory Builders 676
Fill in Key Tables from Memory 677
Definitions 677
Further Reading 677
References in This Chapter 677
Chapter 20 IP Multicast Routing 679
"Do I Know This Already?" Quiz 679
Foundation Topics 683
Multicast Routing Basics 683
Overview of Multicast Routing Protocols 684
Multicast Forwarding Using Dense Mode 684
Reverse-Path-Forwarding Check 685
Multicast Forwarding Using Sparse Mode 687
Multicast Scoping 689
TTL Scoping 689
Administrative Scoping 690
Dense-Mode Routing Protocols 690
Operation of Protocol Independent Multicast Dense Mode 691
Forming PIM Adjacencies Using PIM Hello Messages 691
Source-Based Distribution Trees 692
Prune Message 693
PIM-DM: Reacting to a Failed Link 695
Rules for Pruning 697
Steady-State Operation and the State Refresh Message 699
Graft Message 700
LAN-Specific Issues with PIM-DM and PIM-SM 702
Prune Override 702
Assert Message 703
Designated Router 704
Summary of PIM-DM Messages 705
Distance Vector Multicast Routing Protocol 706
Multicast Open Shortest Path First 706
Sparse-Mode Routing Protocols 707
Operation of Protocol Independent Multicast Sparse Mode 707
Similarities Between PIM-DM and PIM-SM 707
Sources Sending Packets to the Rendezvous Point 708
Joining the Shared Tree 710
Completion of the Source Registration Process 712
Shared Distribution Tree 714
Steady-State Operation by Continuing to Send Joins 715

Examining the RP's Multicast Routing Table 716
Shortest-Path Tree Switchover 717
Pruning from the Shared Tree 719
Dynamically Finding RPs and Using Redundant RPs 720
Dynamically Finding the RP Using Auto-RP 721
Dynamically Finding the RP Using BSR 724
Anycast RP with MSDP 726
Summary: Finding the RP 728
Bidirectional PIM 729
Comparison of PIM-DM and PIM-SM 730
Foundation Summary 732
Memory Builders 736
Fill in Key Tables from Memory 736
Definitions 736
Further Reading 737

Part VII Security 739

Chapter 21 Security 741

"Do I Know This Already?" Quiz 741
Foundation Topics 745
Router and Switch Device Security 745
Simple Password Protection for the CLI 745
Better Protection of Enable and Username Passwords 746
User Mode and Privileged Mode AAA Authentication 747
Using a Default Set of Authentication Methods 748
Using Multiple Authentication Methods 749
Groups of AAA Servers 750
Overriding the Defaults for Login Security 751
PPP Security 752
Layer 2 Security 752
Switch Security Best Practices for Unused and User Ports 753
Port Security 754
Dynamic ARP Inspection 758
DHCP Snooping 761
IP Source Guard 763
802.1X Authentication Using EAP 764
General Layer 2 Security Recommendations 766
Layer 3 Security 768
IP Access Control List Review 769
ACL Rule Summary 770
Wildcard Masks 772
General Layer 3 Security Considerations 772
Smurf Attacks, Directed Broadcasts, and RPF Checks 772
Inappropriate IP Addresses 774
TCP SYN Flood, the Established Bit, and TCP Intercept 775

Foundation Summary 778
Memory Builders 780
Fill in Key Tables from Memory 780
Definitions 781
Further Reading 781

Part VIII Enterprise Wireless Mobility 783

Chapter 22 IEEE 802.11 Fundamentals 785
"Do I Know This Already?" Quiz 785
Foundation Topics 788
802.11 Physical Layer Standards 788
802.11a 788
802.11b 789
802.11g 790
802.11n 791
Comparison of 802.11 Standards 791
Wireless System Configuration 791
Infrastructure Mode Configuration 792
Ad Hoc Mode Configuration 794
Wireless Hardware Components 794
Radio Cards 795
Access Points 795
Antennas 795
Repeaters 796
Bridges 797
Routers 797
Radio Frequency Peripherals 797
Infrastructure Mode Operation 798
Scanning 798
Passive Scanning 798
Active Scanning 799
Connecting with a Network 799
Data Transfer 799
Roaming 800
Ad Hoc Mode Operation 800
Wireless Configuration Parameters 801
SSID 802
RF Channels 803
Transmit Power 804
Data Rates 804
Power-Save Mode 805
RTS/CTS 806
Fragmentation 808
RTS/CTS and Fragmentation Summary 808

Wireless Medium Access 809
Wireless Security 810
WEP 811
TKIP 811
AES 812
WPA 812
Open System Authentication 812
Shared Key Authentication 812
Virtual Private Networks 813
Comparing Wireless Security 813
RF Signal Concepts 814
Modulation 814
RF Signal Characteristics 815
Gain 816
Signal-to-Noise Ratio 816
Spread Spectrum 817
Orthogonal Frequency Division Multiplexing 818
FCC Rules 819
RF Interference 819
Multipath 820
Foundation Summary 822
Memory Builders 822
Definitions 823
Further Reading 823

Chapter 23 Wireless LAN Solutions 825

"Do I Know This Already?" Quiz 825
Foundation Topics 828
Cisco Structured Wireless-Aware Network 828
Wireless Domain Services 828
Intrusion Detection System 829
Cisco SWAN Hardware 831
Cisco Wireless LAN Hardware 832
CiscoWorks Wireless LAN Solution Engine 834
Automatic Access Point Configuration 834
Assisted Site Surveys 835
Centralized Firmware Updates 835
Dynamic Grouping 835
VLAN Configuration 835
Multiple Service Set Identifier Support 835
Customizable Thresholds 835
Fault Status 836
Intrusion Detection System 836
Security Policy Monitoring 836

Secure User Interface 836
Air/RF Scanning and Monitoring 836
Self-Healing Functions 837
Reporting, Trending, Planning, and Troubleshooting 837
Applying Wireless LANs in Enterprises 837
Enterprise Security 837
Voice Services 839
Public Wireless LANs 840
Small Office and Home Wireless LANs 842
Foundation Summary 845
Memory Builders 845
Fill in Key Tables from Memory 845
Definitions 845
Further Reading 845

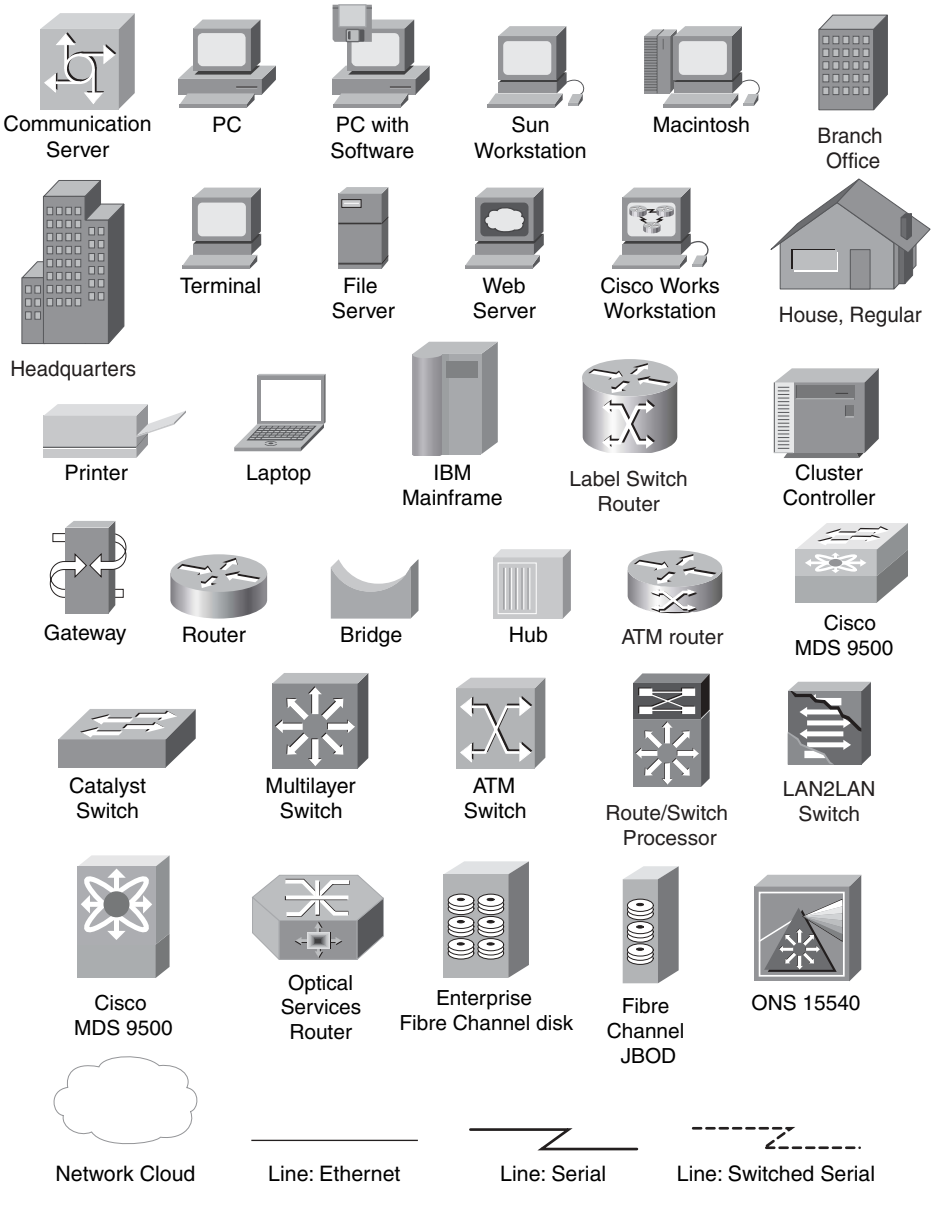
Part IX OSI and Cisco Device Basics 847

Chapter 24 Miscellaneous Networking Theory and Practices 849
"Do I Know This Already?" Quiz 849
Foundation Topics 851
The OSI and TCP/IP Models 851
OSI Layers 851
OSI Layering Concepts and Benefits 854
OSI Terminology 855
OSI Layer Interactions 856
Router Operation Miscellany 858
Cisco IOS Software Boot Sequences and the Configuration Register 858
The Configuration Register 858
The boot system Command 859
CLI Help Features 860
Foundation Summary 862
Memory Builders 863
Fill in Key Tables from Memory 863
Definitions 863

Part X Appendixes 865

Appendix A Answers to the "Do I Know This Already?" Quizzes 867
Appendix B CCIE Routing and Switching Exam Updates: Version 1.0 891
Appendix C MPLS 895
Appendix D Decimal to Binary Conversion Table 953
Glossary 959
Index 1012

Icons Used in This Book



Command Syntax Conventions

The conventions used to present command syntax in this book are the same conventions used in the *Cisco IOS Command Reference*, which describes these conventions as follows:

- **Boldface** indicates commands and keywords that are entered literally as shown. In actual configuration examples and output (not general command syntax), boldface indicates commands that are manually input by the user (such as a **show** command).
- *Italics* indicate arguments for which you supply actual values.
- Vertical bars | separate alternative, mutually exclusive elements.
- Square brackets, [ ], indicate optional elements.
- Braces, { }, indicate a required choice.
- Braces within brackets, [{ }], indicate a required choice within an optional element.

Introduction

The Cisco Certified Internetwork Expert (CCIE) certification may be the most challenging and prestigious of all networking certifications. It has received numerous awards, and certainly has built a reputation as one of the most difficult certifications to earn in all of the computing world. Having a CCIE certification opens doors professionally, typically results in higher pay, and looks great on a résumé.

Cisco currently offers several CCIE certifications, with several others that are no longer offered. The following list details the currently available CCIE certifications as of the time of publication of this book; check http://www.cisco.com/go/ccie for the latest information. The certifications are listed in the order in which they were made available to the public.

- CCIE Routing and Switching
- CCIE Security
- CCIE Service Provider (formerly known as Communications and Services)
- CCIE Voice
- CCIE Storage Networking

Each of the CCIE certifications requires the candidate to pass both a written exam and a one-day hands-on lab exam. The written exam is intended to test your knowledge of theory, protocols, and configurations that follow good design practices. The lab exam proves that you can configure and troubleshoot actual lab gear.

Why Should I Take the CCIE Routing and Switching Written Exam?

The first and most obvious reason to take the CCIE Routing and Switching written exam is that it is the first step toward obtaining the CCIE Routing and Switching certification. Also, you cannot schedule a CCIE lab exam until you pass the appropriate written exam. In short, if you want all the professional benefits of a CCIE Routing and Switching certification, you start by passing the written exam.

The benefits of getting a CCIE certification are varied, but here are just a few of the reasons:

- Better pay
- Better career advancement/new job
- Applies to certain minimum requirements for Cisco Channel Partners, making you more valuable to Channel Partners
- Better movement through the problem-resolution process when calling the Cisco TAC
- Prestige
- Credibility for consultants and customer engineers, including the use of the Cisco CCIE logo

The other big reason to take the CCIE Routing and Switching written exam is that it recertifies an individual's associate-, professional-, and expert-level Cisco certifications. In other words, passing any CCIE written exam recertifies that person's CCNA, CCNP, CCIP, CCSP, CCDP, and so on. (Recertification requirements do change, so please verify the requirements at Cisco.com.)

The CCIE Routing and Switching Written Exam 350-001

The CCIE Routing and Switching written exam, at least as of the time of publication, consists of a 2-hour exam administered at a proctored exam facility affiliated either with Pearson VUE (http://www.VUE.com/cisco) or Thomson Prometric (http://www.2test.com). The exam typically includes approximately 100 multiple-choice questions, with no simulation questions currently on the written exam. Because the written exam is typically followed at some point by an attempt at passing the lab exam, Cisco has little motivation to add simulator questions to any of the CCIE written exams.

As with most exams of any kind, everyone wants to know what is on the exam. Cisco provides general guidance as to topics on the exam in the CCIE Routing and Switching written exam blueprint, the most recent copy of which can be accessed at http://www.cisco.com/go/ccie.

Cisco changes both the written and lab blueprints over time, and with CCIE, Cisco seldom, if ever, changes the exam number. (Cisco changes the exam numbers of the associate- and professional-level certifications when it makes major changes to what is covered on those exams.) Knowing that the content will change over time, this book includes Appendix B, "CCIE Exam Updates." This appendix will include coverage of any newly added topics to the CCIE Routing and Switching written exam. When Cisco changes the blueprint, the authors will add content to cover the new topics at http://www.ciscopress.com/title/1587201410, with that content also being available to all readers who have bought the earlier edition of the book. For future printings, Cisco Press will put that new content into Appendix B.

The CCIE Routing and Switching written exam blueprint, as of the time of publication, is as follows:

I. General Networking Theory

A. OSI Models
B. General Routing Concepts
C. Standards
D. Protocol Mechanics
E. Commands

II. Bridging and LAN Switching

A. Transparent
B. LAN Switching
C. MLS

D. Data Link Layer

E. Ethernet

F. Catalyst IOS Configuration Commands

III. IP

A. Addressing

B. Services

C. Applications

D. Transport

E. IPv6

F. Network Management

IV. IP Routing

A. OSPF

B. BGP

C. EIGRP

D. Route filtering and Policy Routing

E. DDR

F. RIPv2

G. The use of ‘show’ and ‘debug’ commands

V. QoS

A. Traffic classification

B. Congestion management

C. Congestion avoidance

VI. WAN

A. Frame Relay

B. Physical Layer

C. Leased Line Protocols

VII. IP Multicast

A. IGMP/CGMP

B. Addressing

C. Distribution Trees

D. PIM-SM Mechanics

E. Rendezvous Points

F. RPF

VIII. Security

A. Access Lists
B. LAN security
C. Device Security/Access
D. Spoofing

IX. Enterprise Wireless Mobility

A. Standards
B. Hardware
C. SWAN
D. RF Troubleshooting
E. VoWLAN
F. Products

The blueprint tells you what major topics to study, and which not to study by implication. However, the blueprint does not provide many details about the scope and depth covered for each topic. For example, the blueprint lists BGP, without any details. While the lack of details on the depth and breadth of coverage may be a little frustrating, the positive perspective is that the lab can cover far more details—so it is never a bad idea to study too many details for the written exam, because the extra topics are probably topics that could be on the lab exam anyway.

Knowing what topics Cisco does not list in the blueprint is also useful, particularly topics that Cisco has removed from earlier blueprints. For example, Cisco announced the removal of ISDN/DDR, IS-IS, ATM, and SONET from the written exam blueprint during the summer if 2005, making it a reasonable strategy to simply not study those topics today. Also, there is a possibility that MPLS might be added back to the exam—check http://www.cisco.com/go/ccie for the latest information regarding MPLS or any other new or deleted blueprint topics.

About the *CCIE Routing and Switching Official Exam Certification Guide*, Second Edition

This section provides a brief insight into the contents of the book, the major goals, and some of the book features that you will encounter when using this book.

Book Organization

This book contains nine major parts, one corresponding to each of the nine parts of the CCIE Routing and Switching written exam blueprint. Each part has one or more chapters covering the major topic areas inside each part of the blueprint.

The order of the parts inside the book mostly matches the blueprint, with one exception: Part I in the blueprint (General Networking Theory) is covered as Part IX, "OSI and Cisco Device Basics," of this book. We decided to make the blueprint's Part I be the final part of the book for two main reasons: first, many of the topics in that part of the blueprint are more easily covered as part of other topics, so the details were included in other parts of the book. Second, several of the topics from Part I of the blueprint are relatively basic, being covered on the CCNA exam, so we actually debated whether to bother including the topics in this book at all. However, to be complete, the topics are included, but placed at the end of the book.

Each part of the book has one or more chapters. Some have a single chapter, such as Part VII, "Security." However, Part III, "IP Routing," has seven chapters, and a lot of page count.

Beyond the chapters in the nine major parts of the book, you will find several useful appendixes gathered in Part X. In particular, Appendix B, "CCIE Exam Updates," as mentioned earlier, will be updated online at http://www.ciscopress.com/title/1587201410 when appropriate to provide you with the most up to date material. Appendix C covers MPLS, because it was being considered for inclusion in the CCIE Routing and Switching written exam blueprint at the time of publication. Please check http://www.cisco.com and the web page for this book at http://www.ciscopress.com/title/1587201410 to see the latest information about whether or not you need to read the MPLS appendix. Also included in Part X is a decimal to binary conversion chart for reference in Appendix D.

Following is a description of each part's coverage:

- **Part I, "Bridging and LAN Switching" (Chapters 1–3)**

  This part focuses on LAN Layer 2 features, specifically Ethernet (Chapter 1), VLANs and trunking (Chapter 2), and Spanning Tree Protocol (Chapter 3).

- **Part II, "TCP/IP" (Chapters 4–6)**

  This part is titled "IP" to match the blueprint, but it might be better titled "TCP/IP" because it covers details across the spectrum of the TCP/IP protocol stack. It includes IP addressing (Chapter 4), IP services like DHCP, ARP, and ICMP (Chapter 5), and protocol details for TCP, UDP, and application layer protocols (Chapter 6).

- **Part III, "IP Routing" (Chapters 7–13)**

  This part covers some of the more important topics on the exam, and is easily the largest part of the book. It covers Layer 3 forwarding concepts (Chapter 7), followed by three routing protocol chapters, one each about RIP, EIGRP, and OSPF (Chapters 8 through 10, respectively). Following that, Chapter 11 covers route redistribution between IGPs. At the end, two chapters (12 and 13) hit the details of BGP.

- **Part IV, "Quality of Service" (Chapters 14–16)**

  This part covers the more popular QoS tools, including some MQC-based tools, as well as several older tools, particularly FRTS. The chapters include coverage of classification and marking (Chapter 14), queuing and congestion avoidance (Chapter 15), plus shaping, policing, and link efficiency (Chapter 16).

- **Part V, "WAN" (Chapters 17–18)**

  The WAN coverage in the blueprint shrunk in the summer of 2005 with the removal of ATM, SONET, ISDN, and DDR. The potential addition of MPLS back into the CCIE Routing and Switching written blueprint (see http://www.cisco.com for the latest, or this book's page at http://www.ciscopress.com/title/1587201410) would add another WAN-oriented topic. The book's WAN section covers two main topics: point-to-point protocols and concepts (Chapter 17) and Frame Relay (Chapter 18).

- **Part VI, "IP Multicast" (Chapters 19–20)**

  This is one of the two parts of the book that cover topics that are mostly ignored for the CCNP exam. As a result, the text assumes that the reader has no knowledge of multicast before beginning this part. Chapter 19 covers multicast on LANs, including IGMP and how hosts join multicast groups. Chapter 20 covers multicast WAN topics.

- **Part VII, "Security" (Chapter 21)**

  Given the CCIE tracks for both Security and Voice, Cisco has a small dilemma regarding whether to cover those topics on CCIE Routing and Switching, and if so, in how much detail. This part covers a variety of security topics appropriate for CCIE Routing and Switching, in a single chapter. This chapter focuses on switch and router security. (Note that Voice, whose protocols were formerly covered on CCIE Routing and Switching, is not covered in the current blueprint or in this book.)

- **Part VIII, "Enterprise Wireless Mobility" (Chapters 22–23)**

  Cisco added wireless LAN coverage to the blueprint in summer 2004. The coverage focuses on wireless LAN concepts and protocols, along with RF properties of the wireless signals. The coverage is comprised of two chapters: Chapter 22, covering 802.11 wireless LAN fundamentals, and Chapter 23, covering deployment solutions.

- **Part IX, "OSI and Cisco Device Basics" (Chapter 24)**

  The final part of the book covers a few topics from the first part of the blueprint, and is mainly a catch-all chapter for a few small topics that were not appropriate for any other part of the book.

- **Part X, "Appendixes"**
  - **Appendix A, "Answers to the 'Do I Know This Already?' Quizzes"**

    This appendix lists the questions covered at the beginning of each chapter and their corresponding answers.

  - **Appendix B, "CCIE Routing and Switching Exam Updates: Version 1.0"**

    As of the first printing of the book, this appendix contains only a few words that reference the web page for this book at http://www.ciscopress.com/title/1587201410. As the blueprint evolves over time, the authors will post new materials at the website. Any future printings of the book will include the latest newly added materials in printed form inside Appendix B.

  - **Appendix C, "MPLS"**

    This appendix covers many of the basics of MPLS, with some focus on the issues between the CE and PE routers. This coverage is an appendix because, as of press time, Cisco had not made a final decision about whether to add MPLS coverage back to the CCIE Routing and Switching exam. Please check http://www.ciscopress.com/title/1587201410 for information about whether you should study this section.

  - **Appendix D, "Decimal to Binary Conversion Table"**

    This appendix lists the decimal values 0 through 255, with their binary equivalents.

  - **(CD-only) Appendix E, "IP Addressing Practice"**

    (This appendix is in a PDF on the CD, in printable format.) This appendix lists several practice problems for IP subnetting and finding summary routes. The explanations to the answers use the shortcuts described in the book.

  - **(CD-only) Appendix F, "Key Tables for CCIE Study"**

    (This appendix is in a PDF on the CD, in printable format.) This appendix lists the most important tables from the core chapters of the book. The tables have much of the content removed. You can print the PDF, and then fill in the table from memory, checking your answers against the tables in the book.

Book Features

The core chapters of this book have several features that help you make the best use of your time:

- **"Do I Know This Already?" Quizzes**—Each chapter begins with a quiz that helps you to determine the amount of time you need to spend studying that chapter. If you follow the directions at the beginning of the chapter, the "Do I Know This Already?" quiz directs you to study all or particular parts of the chapter.
- **Foundation Topics**—These are the core sections of each chapter. They explain the protocols, concepts, and configuration for the topics in that chapter.
- **Foundation Summary**—The "Foundation Summary" section of this book departs from the typical features of the "Foundation Summary" section of other Cisco Press *Official Exam Certification Guides*. This section does not repeat any details from the "Foundation Topics" section; instead, it simply summarizes and lists facts related to the chapter, but for which a longer or more detailed explanation is not warranted.
- **Key Points**—Throughout the "Foundation Topics" section, a Key Point icon has been placed beside the most important areas for review. After reading a chapter, when doing your final preparation for the exam, take the time to flip through the chapters, looking for the Key Point icons, and review those paragraphs, tables, figures, and lists.
- **Fill in Key Tables from Memory**—The more important tables from the chapters have been copied to PDF files available on the CD as Appendix F. The tables have most of the information removed. After printing these mostly-empty tables, you can use them to improve your memory of the facts in the table by trying to fill them out. This tool should be useful for memorizing key facts.
- **CD-based practice exam**—The companion CD contains multiple-choice questions and a testing engine. The CD includes two question banks: one that consists of all the "Do I Know This Already?" quiz questions, and another set that includes questions unique to the CD. As part of your final preparation, you should practice with these questions to help you get used to the exam-taking process, as well as help refine and prove your knowledge of the exam topics.
- **Key Terms and Glossary**—The more important terms mentioned in each chapter are listed at the end of each chapter under the heading "Definitions." The glossary, found at the end of the book, lists all the terms from the chapters. When studying each chapter, you should review the key terms, and for those terms about which you are unsure of the definition, you can review the short definitions from the glossary.
- **Further Reading**—Each chapter includes a suggested set of books and websites for additional study on the same topics covered in that chapter. Often, these references will be useful tools for preparation for the CCIE Routing and Switching lab exam.

Part I: Bridging and LAN Switching

Chapter 1 Ethernet Basics

Chapter 2 Virtual LANs and VLAN Trunking

Chapter 3 Spanning Tree Protocol

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- Bridging and LAN Switching
  - — Transparent
  - — LAN Switching
  - — Data Link Layer
  - — Ethernet
  - — Catalyst IOS Configuration Commands

In addition, this chapter covers information related to the following specific CCIE Routing and Switching written exam topics:

- Ethernet cabling
- Ethernet framing and addressing
- CSMA/CD
- Switch forwarding logic

CHAPTER 1

Ethernet Basics

It's no surprise that the concepts, protocols, and commands related to Ethernet are a key part of the CCIE Routing and Switching written exam. Almost all campus networks today are built using Ethernet technology. Also, Ethernet technology is moving into the WAN with the emergence of metro Ethernet. Even in an IT world, where technology changes rapidly, you can expect that ten years from now, Ethernet will still be an important part of the CCIE Routing and Switching written and lab exams.

For this chapter, if I had to venture a guess, probably 100 percent of you reading this book know a fair amount about Ethernet basics already. I must admit, I was tempted to leave it out. However, I would also venture a guess that at least some of you have forgotten a few facts about Ethernet. So you can read the whole chapter if your Ethernet recollections are a bit fuzzy—or you could just hit the highlights. For exam preparation, it is typically useful to use all the refresher tools: take the "Do I Know This Already?" quiz, complete the definitions of the terms listed at the end of the chapter, print and complete the tables in Appendix F, "Key Tables for CCIE Study," and certainly answer all the CD-ROM questions concerning Ethernet.

"Do I Know This Already?" Quiz

Table 1-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 1-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Ethernet Layer 1: Wiring, Speed, and Duplex	1–5	
Ethernet Layer 2: Framing and Addressing	6–7	
Switching and Bridging Logic	8	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. Which of the following denotes the correct usage of pins on the RJ-45 connectors at the opposite ends of an Ethernet cross-over cable?
   a. 1 to 1
   b. 1 to 2
   c. 1 to 3
   d. 6 to 1
   e. 6 to 2
   f. 6 to 3

2. Which of the following denotes the correct usage of pins on the RJ-45 connectors at the opposite ends of an Ethernet straight-through cable?
   a. 1 to 1
   b. 1 to 2
   c. 1 to 3
   d. 6 to 1
   e. 6 to 2
   f. 6 to 3

3. Which of the following commands must be configured on a Cisco IOS switch interface to disable Ethernet auto-negotiation?
   a. **no auto-negotiate**
   b. **no auto**
   c. Both **speed** and **duplex**
   d. **duplex**
   e. **speed**

4. Consider an Ethernet cross-over cable between two 10/100 ports on Cisco switches. One switch has been configured for 100-Mbps full duplex. Which of the following is true about the other switch?
   a. It will use a speed of 10 Mbps.
   b. It will use a speed of 100 Mbps.
   c. It will use a duplex setting of half duplex.
   d. It will use a duplex setting of full duplex.

5. Consider an Ethernet cross-over cable between two 10/100/1000 ports on Cisco switches. One switch has been configured for half duplex, and the other for full duplex. The ports successfully negotiate a speed of 1 Gbps. Which of the following could occur as a result of the duplex mismatch?

   a. No frames can be received by the half-duplex switch without it believing an FCS error has occurred.

   b. CDP would detect the mismatch and change the full-duplex switch to half duplex.

   c. CDP would detect the mismatch and issue a log message to that effect.

   d. The half-duplex switch will erroneously believe collisions have occurred.

6. Which of the following Ethernet header type fields is a 2-byte field?

   a. DSAP

   b. Type (in SNAP header)

   c. Type (in Ethernet V2 header)

   d. LLC Control

7. Which of the following standards defines a Fast Ethernet standard?

   a. IEEE 802.1Q

   b. IEEE 802.1U

   c. IEEE 802.1X

   d. IEEE 802.1Z

   e. IEEE 802.1AB

   f. IEEE 802.1AD

8. Suppose a brand-new Cisco IOS–based switch has just been taken out of the box and cabled to several devices. One of the devices sends a frame. For which of the following destinations would a switch flood the frames out all ports (except the port upon which the frame was received)?

   a. Broadcasts

   b. Unknown unicasts

   c. Known unicasts

   d. Multicasts

Foundation Topics

Ethernet Layer 1: Wiring, Speed, and Duplex

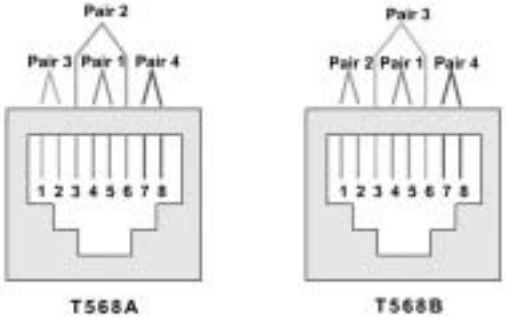
Before making an Ethernet LAN functional, end-user devices, routers, and switches must be cabled correctly. To run with fewer transmission errors at higher speeds, and to support longer cable distances, variations of copper and optical cabling can be used. The different Ethernet specifications, cable types, and cable lengths per the various specifications are important for the exam, and are listed in the "Foundation Summary" section.

RJ-45 Pinouts and Category 5 Wiring

You should know the details of cross-over and straight-through Category 5 (Cat 5) or Cat 5e cabling for most any networking job. The EIA/TIA defines the cabling specifications for Ethernet LANs (http://www.eia.org and http://www.tiaonline.org), including the pinouts for the RJ-45 connects, as shown in Figure 1-1.

Figure 1-1 *RJ-45 Pinouts with Four-Pair UTP Cabling*

KEY POINT



The most popular Ethernet standards (10BASE-T, 100BASE-T, and 1000BASE-T) each use two twisted pairs (specifically pairs 2 and 3 shown in Figure 1-1), with one pair used for transmission in each direction. Depending on which pair a device uses to transmit and receive, either a straight-through or cross-over cable is required. Table 1-2 summarizes how the cabling and pinouts work.

Table 1-2 *Ethernet Cabling Types*

KEY POINT

Type of Cable	Pinouts	Key Pins Connected
Straight-through	T568A (both ends) or T568B (both ends)	1 – 1; 2 – 2; 3 – 3; 6 – 6
Cross-over	T568A on one end, T568B on the other	1 – 3; 2 – 6; 3 – 1; 6 – 2

Many Ethernet standards use two twisted pairs, with one pair being used for transmission in each direction. For instance, a PC network interface card (NIC) transmits on pair 1,2 and receives on pair 3,6; switch ports do the opposite. So, a straight-through cable works well, connecting pair 1,2 on the PC (PC transmit pair) to the switch port's pair 1,2, on which the switch receives. When the two devices on the ends of the cable both transmit using the same pins, a cross-over cable is required. For instance, if two connected switches send using the pair at pins 3,6 and receive on pins 1,2, then the cable needs to connect the pair at 3,6 on one end to pins 1,2 at the other end, and vice versa.

NOTE Cross-over cables can also be used between a pair of PCs, swapping the transmit pair on one end (1,2) with the receive pins at the other end (3,6).

Cisco also supports a switch feature that lets the switch figure out if the wrong cable is installed: *Auto-MDIX* (automatic medium-dependent interface crossover) detects the wrong cable and causes the switch to swap the pair it uses for transmitting and receiving, which solves the cabling problem. (As of publication, this feature is not supported on all Cisco switch models.)

Auto-negotiation, Speed, and Duplex

By default, each Cisco switch port uses *Ethernet auto-negotiation* to determine the speed and duplex setting (half or full). The switches can also set their duplex setting with the **duplex** interface subcommand, and their speed with—you guessed it—the **speed** interface subcommand.

Switches can dynamically detect the speed setting on a particular Ethernet segment by using a few different methods. Cisco switches (and many other devices) can sense the speed using the *Fast Link Pulses (FLP)* of the auto-negotiation process. However, if auto-negotiation is disabled on either end of the cable, the switch detects the speed anyway based on the incoming electrical signal. You can force a speed mismatch by statically configuring different speeds on either end of the cable, causing the link to no longer function.

Switches detect duplex settings through auto-negotiation only. If both ends have auto-negotiation enabled, the duplex is negotiated. However, if either device on the cable disables auto-negotiation, the devices without a configured duplex setting must assume a default. Cisco switches use a default duplex setting of half duplex (HDX) (for 10-Mbps and 100-Mbps interfaces) or full duplex (FDX) (for 1000-Mbps interfaces). To disable auto-negotiation on a Cisco switch port, you simply need to statically configure the speed and the duplex settings.

Ethernet devices can use FDX only when collisions cannot occur on the attached cable; a collision-free link can be guaranteed only when a shared hub is not in use. The next few topics review how Ethernet deals with collisions when they do occur, as well as what is different with Ethernet logic in cases where collisions cannot occur and FDX is allowed.

CSMA/CD

The original Ethernet specifications expected collisions to occur on the LAN. The media was shared, creating a literal electrical bus. Any electrical signal induced onto the wire could collide with a signal induced by another device. When two or more Ethernet frames overlap on the transmission medium at the same instant in time, a collision occurs; the collision results in bit errors and lost frames.

The original Ethernet specifications defined the *Carrier Sense Multiple Access with Collision Detection (CSMA/CD)* algorithm to deal with the inevitable collisions. CSMA/CD minimizes the number of collisions, but when they occur, CSMA/CD defines how the sending stations can recognize the collisions and retransmit the frame. The following list outlines the steps in the CSMA/CD process:

KEY POINT

1. A device with a frame to send listens until the Ethernet is not busy (in other words, the device cannot sense a carrier signal on the Ethernet segment).
2. When the Ethernet is not busy, the sender begins sending the frame.
3. The sender listens to make sure that no collision occurred.
4. If there was a collision, all stations that sent a frame send a jamming signal to ensure that all stations recognize the collision.
5. After the jamming is complete, each sender of one of the original collided frames randomizes a timer and waits that long before resending. (Other stations that did not create the collision do not have to wait to send.)
6. After all timers expire, the original senders can begin again with Step 1.

Collision Domains and Switch Buffering

A *collision domain* is a set of devices that can send frames that collide with frames sent by another device in that same set of devices. Before the advent of LAN switches, Ethernets were either physically shared (10BASE2 and 10BASE5) or shared by virtue of shared hubs and their Layer 1 "repeat out all other ports" logic. Ethernet switches greatly reduce the number of possible collisions, both through frame buffering and through their more complete Layer 2 logic.

By definition of the term, Ethernet hubs:

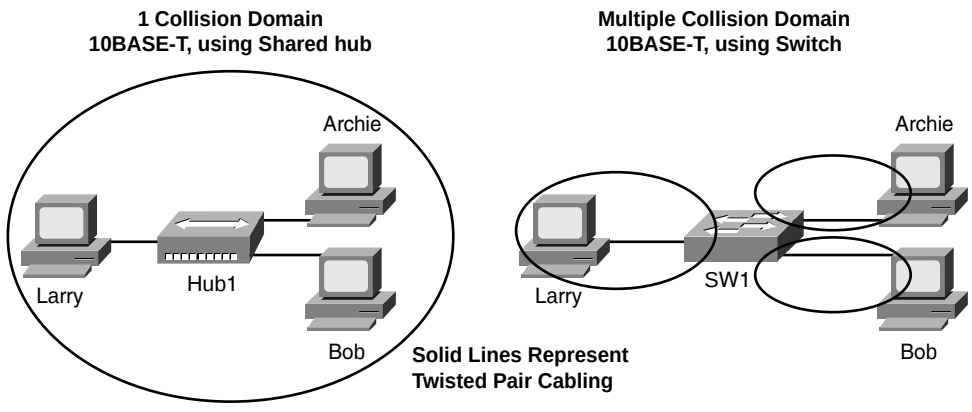
KEY POINT

- Operate solely at Ethernet Layer 1
- Repeat (regenerate) electrical signals to improve cabling distances
- Forward signals received on a port out all other ports (no buffering)

As a result of a hub's logic, a hub creates a single *collision domain*. Switches, however, create a different collision domain per switch port, as shown in Figure 1-2.

Figure 1-2 *Collision Domains with Hubs and Switches*

KEY POINT



Switches have the same cabling and signal regeneration benefits as hubs, but switches do a lot more—including sometimes reducing or even eliminating collisions by buffering frames. When switches receive multiple frames on different switch ports, they store the frames in memory buffers to prevent collisions.

For instance, imagine that a switch receives three frames at the same time, entering three different ports, and they all must exit the same switch port. The switch simply stores two of the frames in memory, forwarding the frames sequentially. As a result, in Figure 1-2, the switch prevents any frame sent by Larry from colliding with a frame sent by Archie or Bob—which by definition puts each of the PCs attached to the switch in Figure 1-2 in different collision domains.

When a switch port connects via cable to a single other non-hub device—for instance, like the three PCs in Figure 1-2—no collisions can possibly occur. The only devices that could create a collision are the switch port and the one connected device—and they each have a separate twisted pair on which to transmit. Because collisions cannot occur, such segments can use full-duplex logic.

When a switch port connects to a hub, it needs to operate in HDX mode, because collisions might occur due to the logic used by the hub.

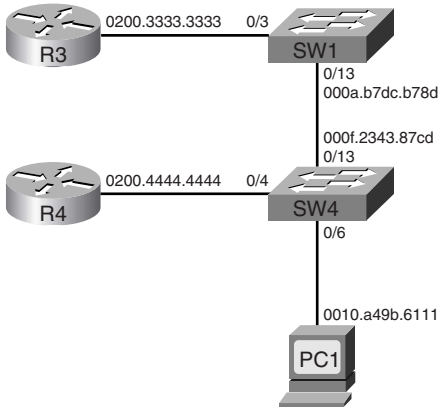
NOTE NICs operating in HDX mode use *loopback circuitry* when transmitting a frame. This circuitry loops the transmitted frame back to the receive side of the NIC, so that when the NIC receives a frame over the cable, the combined looped-back signal and received signal allows the NIC to notice that a collision has occurred.

Basic Switch Port Configuration

The three key configuration elements on a Cisco switch port are auto-negotiation, speed, and duplex. Cisco switches use auto-negotiation by default; it is then disabled if both the speed and duplex are manually configured. You can set the speed using the **speed** {**auto** | **10** | **100** | **1000**} interface subcommand, assuming the interface supports multiple speeds. You configure the duplex setting using the **duplex** {**auto** | **half** | **full**} interface subcommand.

Example 1-1 shows the manual configuration of the speed and duplex on the link between Switch1 and Switch4 from Figure 1-3, and the results of having mismatched duplex settings. (The book refers to specific switch commands used on IOS-based switches, referred to as "Catalyst IOS" by the Cisco CCIE blueprint.)

Figure 1-3 *Simple Switched Network with Trunk*



Example 1-1 *Manual Setting for Duplex and Speed, with Mismatched Duplex*

```
switch1# show interface fa 0/13
FastEthernet0/13 is up, line protocol is up
  Hardware is Fast Ethernet, address is 000a.b7dc.b78d (bia 000a.b7dc.b78d)
  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
! remaining lines omitted for brevity
! Below, Switch1's interface connecting to Switch4 is configured for 100 Mbps,
! HDX. Note that IOS rejects the first duplex command; you cannot set duplex until
! the speed is manually configured.
switch1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
switch1(config)# int fa 0/13
switch1(config-if)# duplex half
Duplex will not be set until speed is set to non-auto value
switch1(config-if)# speed 100
```

Example 1-1 *Manual Setting for Duplex and Speed, with Mismatched Duplex (Continued)*

```
05:08:41: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/13, changed state
 to down
05:08:46: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/13, changed state
 to up
switch1(config-if)# duplex half
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! NOT SHOWN: Configuration for 100/half on Switch4's int fa 0/13.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Now with both switches manually configured for speed and duplex, neither will be
! using Ethernet auto-negotiation. As a result, below the duplex setting on Switch1
! can be changed to FDX with Switch4 remaining configured to use HDX.
switch1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
switch1(config)# int fa 0/13
switch1(config-if)# duplex full
05:13:03: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/13, changed state
 to down
05:13:08: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/13, changed state
 to up
switch1(config-if)#^Z
switch1# sh int fa 0/13
FastEthernet0/13 is up, line protocol is up
! Lines omitted for brevity
  Full-duplex, 100Mb/s
! remaining lines omitted for brevity
! Below, Switch4 is shown to be HDX. Note
! the collisions counters at the end of the show interface command.
switch4# sh int fa 0/13
FastEthernet0/13 is up, line protocol is up (connected)
  Hardware is Fast Ethernet, address is 000f.2343.87cd (bia 000f.2343.87cd)
  MTU 1500 bytes, BW 100000 Kbit, DLY 1000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Half-duplex, 100Mb/s
! Lines omitted for brevity
  5 minute output rate 583000 bits/sec, 117 packets/sec
     25654 packets input, 19935915 bytes, 0 no buffer
     Received 173 broadcasts (0 multicast)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 173 multicast, 0 pause input
     0 input packets with dribble condition detected
     26151 packets output, 19608901 bytes, 0 underruns
     54 output errors, 5 collisions, 0 interface resets
     0 babbles, 54 late collision, 59 deferred
     0 lost carrier, 0 no carrier, 0 PAUSE output
     0 output buffer failures, 0 output buffers swapped out
```

continues

Example 1-1 *Manual Setting for Duplex and Speed, with Mismatched Duplex (Continued)*

KEY POINT

```
02:40:49: %CDP-4-DUPLEX_MISMATCH: duplex mismatch discovered on FastEthernet0/13
(not full duplex), with Switch1 FastEthernet0/13 (full duplex).
! Above, CDP messages have been exchanged over the link between switches. CDP
! exchanges information about Duplex on the link, and can notice (but not fix)
! the mismatch.
```

The statistics on switch4 near the end of the example show collisions (detected in the time during which the first 64 bytes were being transmitted) and late collisions (after the first 64 bytes were transmitted). In an Ethernet that follows cabling length restrictions, collisions should be detected while the first 64 bytes are being transmitted. In this case, Switch1 is using FDX logic, meaning it sends frames anytime—including when Switch4 is sending frames. As a result, Switch4 receives frames anytime, and if sending at the time, it believes a collision has occurred. Switch4 has deferred 59 frames, meaning that it chose to wait before sending frames because it was currently receiving a frame. Also, the retransmission of the frames that Switch4 thought were destroyed due to a collision, but may not have been, causes duplicate frames to be received, occasionally causing application connections to fail and routers to lose neighbor relationships.

Ethernet Layer 2: Framing and Addressing

In this book, as in many Cisco courses and documents, the word *frame* refers to the bits and bytes that include the Layer 2 header and trailer, along with the data encapsulated by that header and trailer. The term *packet* is most often used to describe the Layer 3 header and data, without a Layer 2 header or trailer. Ethernet's Layer 2 specifications relate to the creation, forwarding, reception, and interpretation of Ethernet frames.

The original Ethernet specifications were owned by the combination of Digital Equipment Corp., Intel, and Xerox—hence the name "Ethernet (DIX)" shown in Figure 1-4, which shows the various Ethernet frame formats. Later, in the early 1980s, the IEEE standardized Ethernet, defining parts (Layer 1 and some of Layer 2) in the 802.3 *Media Access Control (MAC)* standard, and other parts of Layer 2 in the 802.2 *Logical Link Control (LLC)* standard. Later, the IEEE realized that the 1-byte DSAP field in the 802.2 LLC header was too small. As a result, the IEEE introduced a new frame format with a *Sub-Network Access Protocol (SNAP)* header after the 802.2 header, as shown in the third style of header in Figure 1-4. Later, in 1997, the IEEE added the original DIX V2 framing to the 802.3 standard as well.

Table 1-3 lists the header fields, along with a brief explanation. The more important fields are explained in more detail after the table.

Figure 1-4 *Ethernet Framing Options*

KEY POINT

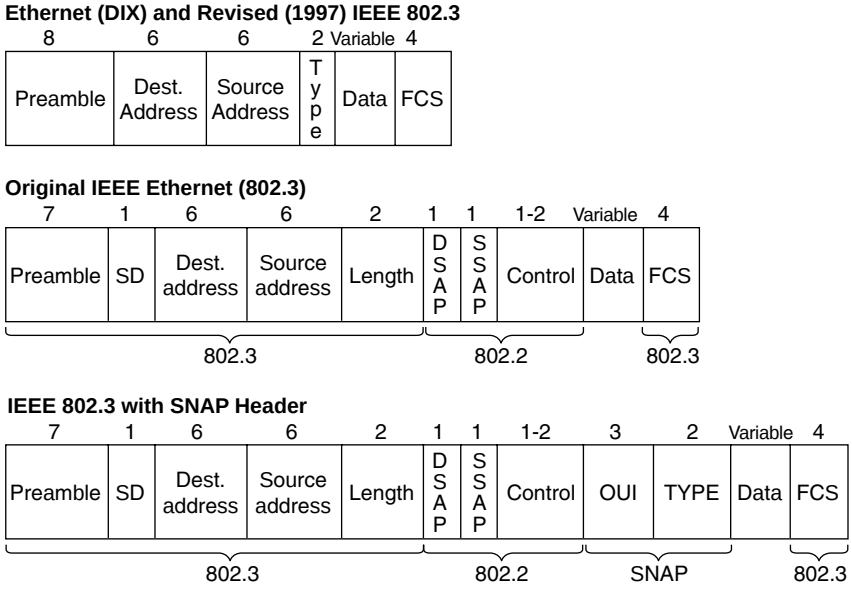


Table 1-3 *Ethernet Header Fields*

KEY POINT

Field	Description
Preamble (DIX)	Provides synchronization and signal transitions to allow proper clocking of the transmitted signal. Consists of 62 alternating 1s and 0s, and ends with a pair of 1s.
Preamble and Start of Frame Delimiter (802.3)	Same purpose and binary value as DIX preamble; 802.3 simply renames the 8-byte DIX preamble as a 7-byte preamble and a 1-byte Start of Frame Delimiter (SFD).
Type (or Protocol Type) (DIX)	2-byte field that identifies the type of protocol or protocol header that follows the header. Allows the receiver of the frame to know how to process a received frame.
Length (802.3)	Describes the length, in bytes, of the data following the Length field, up to the Ethernet trailer. Allows an Ethernet receiver to predict the end of the received frame.
Destination Service Access Point (802.2)	DSAP; 1-byte protocol type field. The size limitations, along with other uses of the low-order bits, required the later addition of SNAP headers.
Source Service Access Point (802.2)	SSAP; 1-byte protocol type field that describes the upper-layer protocol that created the frame.

continues

Table 1-3 *Ethernet Header Fields (Continued)*

Field	Description
Control (802.2)	1- or 2-byte field that provides mechanisms for both connectionless and connection-oriented operation. Generally used only for connectionless operation by modern protocols, with a 1-byte value of 0x03.
Organizationally Unique Identifier (SNAP)	OUI; 3-byte field, generally unused today, providing a place for the sender of the frame to code the OUI representing the manufacturer of the Ethernet NIC.
Type (SNAP)	2-byte Type field, using same values as the DIX Type field, overcoming deficiencies with size and use of the DSAP field.

Types of Ethernet Addresses

Ethernet addresses, also frequently called MAC addresses, are 6 bytes in length, typically listed in hexadecimal form. There are three main types of Ethernet address, as listed in Table 1-4.

Table 1-4 *Three Types of Ethernet/MAC Address*

Type of Ethernet/MAC Address	Description and Notes
Unicast	Fancy term for an address that represents a single LAN interface. The I/G bit, the most significant bit in the most significant byte, is set to 0.
Broadcast	An address that means “all devices that reside on this LAN right now.” Always a value of hex FFFFFFFFFFFF.
Multicast	A MAC address that implies some subset of all devices currently on the LAN. By definition, the I/G bit is set to 1.

Most engineers instinctively know how unicast and broadcast addresses are used in a typical network. When an Ethernet NIC needs to send a frame, it puts its own unicast address in the Source Address field of the header. If it wants to send the frame to a particular device on the LAN, the sender puts the other device’s MAC address in the Ethernet header’s Destination Address field. If the sender wants to send the frame to every device on the LAN, it sends the frame to the FFFF.FFFF.FFFF broadcast destination address. (A frame sent to the broadcast address is named a *broadcast* or *broadcast frame*, and frames sent to unicast MAC addresses are called *unicasts* or *unicast frames*.)

Multicast Ethernet frames are used to communicate with a possibly dynamic subset of the devices on a LAN. The most common use for Ethernet multicast addresses involves the use of IP multicast. For example, if only 3 of 100 users on a LAN want to watch the same video stream using an IP multicast–based video application, the application can send a single multicast frame. The three interested devices prepare by listening for frames sent to a particular multicast Ethernet address,

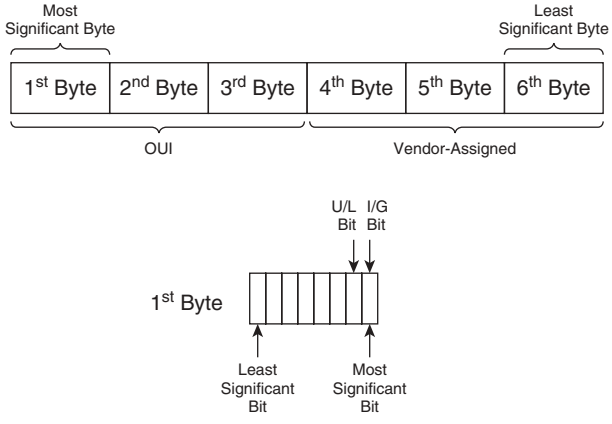
processing frames destined for that address. Other devices may receive the frame, but they ignore its contents. Because the concept of Ethernet multicast is most often used today with IP multicast, most of the rest of the details of Ethernet multicast will be covered in Chapter 19, "Introduction to IP Multicasting."

Ethernet Address Formats

The IEEE intends for unicast addresses to be unique in the universe by administering the assignment of MAC addresses. The IEEE assigns each vendor a code to use as the first 3 bytes of its MAC addresses; that first half of the addresses is called the *Organizationally Unique Identifier (OUI)*. The IEEE expects each manufacturer to use its OUI for the first 3 bytes of the MAC assigned to any Ethernet product created by that vendor. The vendor then assigns a unique value in the low-order 3 bytes for each Ethernet card that it manufactures—thereby ensuring global uniqueness of MAC addresses. Figure 1-5 shows the basic Ethernet address format, along with some additional details.

Figure 1-5 *Ethernet Address Format*

KEY POINT



Note that Figure 1-5 shows the location of the most significant byte and most significant bit in each byte. IEEE documentation lists Ethernet addresses with the most significant byte on the left. However, inside each byte, the leftmost bit is the least significant bit, and the rightmost bit is the most significant bit. Many documents refer to the bit order as *noncanonical;* other documents refer to it as *little-endian.* Regardless of the term, the bit order inside each byte is important for understanding the meaning of the two most significant bits in an Ethernet address:

- The Individual/Group (I/G) bit
- The Universal/Local (U/L) bit

Table 1-5 summarizes the meaning of each bit.

Table 1-5 *I/G and U/L Bits*

Field	Meaning
I/G	Binary 0 means the address is a unicast; Binary 1 means the address is a multicast or broadcast.
U/L	Binary 0 means the address is vendor assigned; Binary 1 means the address has been administratively assigned, overriding the vendor-assigned address.

The I/G bit signifies whether the address represents an individual device or a group of devices, and the U/L bit identifies locally configured addresses. For instance, the Ethernet multicast addresses used by IP multicast implementations always start with 0x01005E. Hex 01 (the first byte of the address) converts to binary 00000001, with the most significant bit being 1, confirming the use of the I/G bit.

NOTE Often, when overriding the MAC address to use a local address, the device or device driver does not enforce the setting of the U/L bit to a value of 1.

Protocol Types and the 802.3 Length Field

Each of the three types of Ethernet header shown in Figure 1-4 has a field identifying the format of the Data field in the frame. Generically called a *Type* field, these fields allow the receiver of an Ethernet frame to know how to interpret the data in the received frame. For instance, a router might want to know whether the frame contains an IP packet, an IPX packet, and so on.

DIX and the revised IEEE framing use the Type field, also called the Protocol Type field. The originally-defined IEEE framing uses those same 2 bytes as a Length field. To distinguish the style of Ethernet header, the Ethernet Type field values begin at 1536, and the length of the Data field in an IEEE frame is limited to decimal 1500 or less. That way, an Ethernet NIC can easily determine whether the frame follows the DIX or original IEEE format.

The original IEEE frame used a 1-byte Protocol Type field (DSAP) for the 802.2 LLC standard type field. It also reserved the high-order 2 bits for other uses, similar to the I/G and U/L bits in MAC addresses. As a result, there were not enough possible combinations in the DSAP field for the needs of the market—so the IEEE had to define yet another type field, this one inside an additional IEEE SNAP header. Table 1-6 summarizes the meaning of the three main Type field options with Ethernet.

Table 1-6 *Ethernet Type Fields*

KEY POINT

Type Field	Description
Protocol Type	DIX V2 Type field; 2 bytes; registered values now administered by the IEEE
DSAP	802.2 LLC; 1 byte, with 2 high-order bits reserved for other purposes; registered values now administered by the IEEE
SNAP	SNAP header; 2 bytes; uses same values as Ethernet Protocol Type; signified by an 802.2 DSAP of 0xAA

Switching and Bridging Logic

In this chapter so far, you have been reminded about the cabling details for Ethernet along with the formats and meanings of the fields inside Ethernet frames. A switch's ultimate goal is to deliver those frames to the appropriate destination(s) based on the destination MAC address in the frame header. Table 1-7 summarizes the logic used by switches when forwarding frames, which differs based on the type of destination Ethernet address and on whether the destination address has been added to its MAC address table.

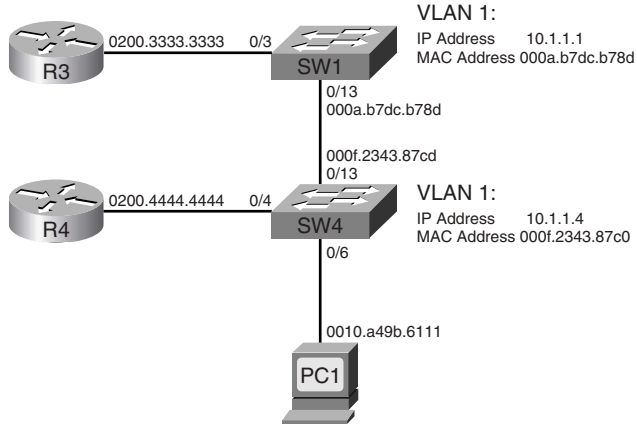
Table 1-7 *LAN Switch Forwarding Behavior*

KEY POINT

Type of Address	Switch Action
Known unicast	Forwards frame out the single interface associated with the destination address
Unknown unicast	Floods frame out all interfaces, except the interface on which the frame was received
Broadcast	Floods frame identically to unknown unicasts
Multicast	Floods frame identically to unknown unicasts, unless multicast optimizations are configured

For unicast forwarding to work most efficiently, switches need to know about all the unicast MAC addresses and out which interface the switch should forward frames sent to each MAC address. Switches learn MAC addresses, and the port to associate with them, by reading the source MAC address of received frames. You can see the learning process in Example 1-2, along with several other details of switch operation. Figure 1-6 lists the devices in the network associated with Example 1-2, along with their MAC addresses.

Figure 1-6 *Sample Network with MAC Addresses Shown*



Example 1-2 *Command Output Showing MAC Address Table Learning*

```
Switch1# show mac-address-table dynamic
          Mac Address Table
----------------------------------------

Vlan    Mac Address       Type       Ports
----    -----------       ----       -----
   1    000f.2343.87cd    DYNAMIC    Fa0/13
   1    0200.3333.3333    DYNAMIC    Fa0/3
   1    0200.4444.4444    DYNAMIC    Fa0/13
Total Mac Addresses for this criterion: 3
! Above, Switch1's MAC address table lists three dynamically learned addresses,
! including Switch4's FA 0/13 MAC.
! Below, Switch1 pings Switch4's management IP address.
Switch1# ping 10.1.1.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.1.1.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
! Below Switch1 now knows the MAC address associated with Switch4's management IP
! address. Each switch has a range of reserved MAC addresses, with the first MAC
! being used by the switch IP address, and the rest being assigned in sequence to
! the switch interfaces - note 0xcd (last byte of 2nd address in the table above)
! is for Switch4's FA 0/13 interface, and is 13 (decimal) larger than Switch4's
! base MAC address.
Switch1# show mac-address-table dynamic
          Mac Address Table
----------------------------------------

Vlan    Mac Address       Type       Ports
----    -----------       ----       -----
   1    000f.2343.87c0    DYNAMIC    Fa0/13
   1    000f.2343.87cd    DYNAMIC    Fa0/13
   1    0200.3333.3333    DYNAMIC    Fa0/3
   1    0200.4444.4444    DYNAMIC    Fa0/13
Total Mac Addresses for this criterion: 4
! Not shown: PC1 ping 10.1.1.23 (R3) PC1's MAC in its MAC address table
----------------------------------------

Vlan    Mac Address       Type       Ports
----    -----------       ----       -----
   1    000f.2343.87c0    DYNAMIC    Fa0/13
   1    000f.2343.87cd    DYNAMIC    Fa0/13
   1    0010.a49b.6111    DYNAMIC    Fa0/13
```

Example 1-2 *Command Output Showing MAC Address Table Learning (Continued)*

```
   1    0200.3333.3333    DYNAMIC    Fa0/3
   1    0200.4444.4444    DYNAMIC    Fa0/13
Total Mac Addresses for this criterion: 5
! Above, Switch1 learned the PC's MAC address, associated with FA 0/13,
! because the frames sent by the PC came into Switch1 over its FA 0/13.
! Below, Switch4's MAC address table shows PC1's MAC off its FA 0/6
switch4# show mac-address-table dynamic
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    000a.b7dc.b780    DYNAMIC     Fa0/13
   1    000a.b7dc.b78d    DYNAMIC     Fa0/13
   1    0010.a49b.6111    DYNAMIC     Fa0/6
   1    0200.3333.3333    DYNAMIC     Fa0/13
   1    0200.4444.4444    DYNAMIC     Fa0/4
Total Mac Addresses for this criterion: 5
! Below, for example, the aging timeout (default 300 seconds) is shown, followed
! by a command just listing the mac address table entry for a single address.
switch4# show mac-address-table aging-time
Vlan    Aging Time
----    ----------
  1     300
switch4# show mac-address-table address 0200.3333.3333
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    0200.3333.3333    DYNAMIC     Fa0/13
Total Mac Addresses for this criterion: 1
```

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most of the Cisco Press *Exam Certification Guides*, this book does not repeat information presented in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in the "Foundation Topics" section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 1-8 lists the different types of Ethernet and some distinguishing characteristics of each type.

Table 1-8 *Ethernet Standards*

Type of Ethernet	General Description
10BASE5	Commonly called "thick-net"; uses coaxial cabling
10BASE2	Commonly called "thin-net"; uses coaxial cabling
10BASE-T	First type of Ethernet to use twisted-pair cabling
DIX Ethernet Version 2	Layer 1 and Layer 2 specifications for original Ethernet, from Digital/Intel/Xerox; typically called DIX V2
IEEE 802.3	Called MAC due to the name of the IEEE committee (Media Access Control); original Layer 1 and 2 specifications, standardized using DIX V2 as a basis
IEEE 802.2	Called LLC due to the name of the IEEE committee (Logical Link Control); Layer 2 specification for header common to multiple IEEE LAN specifications
IEEE 802.3u	IEEE standard for Fast Ethernet (100 Mbps) over copper and optical cabling; typically called FastE
IEEE 802.3z	Gigabit Ethernet over optical cabling; typically called GigE
IEEE 802.3ab	Gigabit Ethernet over copper cabling

Switches forward frames when necessary, and do not forward when there is no need to do so, thus reducing overhead. To accomplish this, switches perform three actions:

- Learn MAC addresses by examining the source MAC address of each received frame
- Decide when to forward a frame or when to filter (not forward) a frame, based on the destination MAC address
- Create a loop-free environment with other bridges by using the Spanning Tree Protocol

The internal processing algorithms used by switches vary among models and vendors; regardless, the internal processing can be categorized as one of the methods listed in Table 1-9.

Table 1-9 *Switch Internal Processing*

Switching Method	Description
Store-and-forward	The switch fully receives all bits in the frame (store) before forwarding the frame (forward). This allows the switch to check the FCS before forwarding the frame, thus ensuring that errored frames are not forwarded.
Cut-through	The switch performs the address table lookup as soon as the Destination Address field in the header is received. The first bits in the frame can be sent out the outbound port before the final bits in the incoming frame are received. This does not allow the switch to discard frames that fail the FCS check, but the forwarding action is faster, resulting in lower latency.
Fragment-free	This performs like cut-through switching, but the switch waits for 64 bytes to be received before forwarding the first bytes of the outgoing frame. According to Ethernet specifications, collisions should be detected during the first 64 bytes of the frame, so frames that are in error because of a collision will not be forwarded.

Table 1-10 lists some of the most popular Cisco IOS commands related to the topics in this chapter.

Table 1-10 *Catalyst IOS Commands for Catalyst Switch Configuration*

Command	Description
interface vlan 1	Global command; moves user to interface configuration mode for a VLAN interface
interface fastethernet *0/x*	Puts user in interface configuration mode for that interface
duplex {**auto** | **full** | **half**}	Used in interface configuration mode; sets duplex mode for the interface
speed {**10** | **100** | **1000** | **auto** | **nonegotiate**}	Used in interface configuration mode; sets speed for the interface
show mac address-table [**aging-time** | **count** | **dynamic** | **static**] [**address** *hw-addr*] [**interface** *interface-id*] [**vlan** *vlan-id*]	Displays the MAC address table; the security option displays information about the restricted or static settings
show interface fastethernet 0/x	Displays interface status for a physical 10/100 interface
show interface vlan 1	Displays IP address configuration for VLAN

Table 1-11 outlines the types of UTP cabling.

Table 1-11 *UTP Cabling Reference*

UTP Category	Max Speed Rating	Description
1	—	Used for telephones, and not for data
2	4 Mbps	Originally intended to support Token Ring over UTP
3	10 Mbps	Can be used for telephones as well; popular option for Ethernet in years past, if Cat 3 cabling for phones was already in place
4	16 Mbps	Intended for the fast Token Ring speed option
5	1 Gbps	Very popular for cabling to the desktop
5e	1 Gbps	Added mainly for the support of copper cabling for Gigabit Ethernet
6	1 Gbps+	Intended as a replacement for Cat 5e, with capabilities to support multigigabit speeds

Table 1-12 lists the pertinent details of the Ethernet standards and the related cabling.

Table 1-12 *Ethernet Types and Cabling Standards*

Standard	Cabling	Maximum Single Cable Length
10BASE5	Thick coaxial	500 m
10BASE2	Thin coaxial	185 m
10BASE-T	UTP Cat 3, 4, 5, 5e, 6	100 m
100BASE-FX	Two strands, multimode	400 m
100BASE-T	UTP Cat 3, 4, 5, 5e, 6, 2 pair	100 m
100BASE-T4	UTP Cat 3, 4, 5, 5e, 6, 4 pair	100 m
100BASE-TX	UTP Cat 3, 4, 5, 5e, 6, or STP, 2 pair	100 m
1000BASE-LX	Long-wavelength laser, MM or SM fiber	10 km (SM) 3 km (MM)
1000BASE-SX	Short-wavelength laser, MM fiber	220 m with 62.5-micron fiber; 550 m with 50-micron fiber
1000BASE-ZX	Extended wavelength, SM fiber	100 km
1000BASE-CS	STP, 2 pair	25 m
1000BASE-T	UTP Cat 5, 5e, 6, 4 pair	100 m

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

> Auto-negotiation, half duplex, full duplex, cross-over cable, straight-through cable, unicast address, multicast address, broadcast address, loopback circuitry, I/G bit, U/L bit, CSMA/CD

Refer to the CD-based glossary to check your answers.

Further Reading

For a good reference for more information on the actual FLPs used by auto-negotiation, refer to the Fast Ethernet web page of the University of New Hampshire Research Computing Center's InterOperability Laboratory, at http://www.iol.unh.edu/training/fe/.

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- Bridging and LAN Switching
- LAN Switching
- MLS
- Catalyst IOS Configuration Commands

In addition, this chapter covers information related to the following specific CCIE Routing and Switching written exam topics:

- VLANs
- VTP
- VLAN trunking

CHAPTER 2

Virtual LANs and VLAN Trunking

This chapter continues with the coverage of some of the most fundamental and important LAN topics with coverage of VLANs and VLAN trunking. As usual, for those of you current in your knowledge of the topics in this chapter, review the items next to the Key Point icons spread throughout the chapter, plus the "Foundation Summary," "Memory Builders," and "Q&A" sections at the end of the chapter.

"Do I Know This Already?" Quiz

Table 2-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 2-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Virtual LANs	1–2	
VLAN Trunking Protocol	3–5	
VLAN Trunking: ISL and 802.1Q	6–9	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. Assume that VLAN 28 does not yet exist on Switch1. Which of the following commands, issued from any part of global configuration mode (reached with the **configure terminal** exec command) would cause the VLAN to be created?

a. vlan 28

b. vlan 28 name fred

c. switchport vlan 28

d. switchport access vlan 28

e. switchport access 28

2. Which of the following are the two primary motivations for using private VLANs?
   a. Better LAN security
   b. IP subnet conservation
   c. Better consistency in VLAN configuration details
   d. Reducing the impact of broadcasts on end-user devices
   e. Reducing the unnecessary flow of frames to switches that do not have any ports in the VLAN to which the frame belongs

3. Which of the following VLANs can be pruned by VTP on an 802.1Q trunk?
   a. 1–1023
   b. 1–1001
   c. 2–1001
   d. 1–1005
   e. 2–1005

4. An existing switched network has ten switches, with Switch1 and Switch2 being the only VTP servers in the network. The other switches are all VTP clients and have successfully learned about the VLANs from the VTP servers. The only configured VTP parameter on all switches is the VTP domain name (Larry). The VTP revision number is 201. What happens when a new, already-running VTP client switch, named Switch11, with domain name Larry and revision number 301, connects via a trunk to any of the other ten switches?
   a. No VLAN information changes; Switch11 ignores the VTP updates sent from the two existing VTP servers until the revision number reaches 302.
   b. The original ten switches replace their old VLAN configuration with the configuration in Switch11.
   c. Switch11 replaces its own VLAN configuration with the configuration sent to it by one of the original VTP servers.
   d. Switch11 merges its existing VLAN database with the database learned from the VTP servers, because Switch11 had a higher revision number.

5. An existing switched network has ten switches, with Switch1 and Switch2 being the only VTP servers in the network. The other switches are all VTP clients, and have successfully learned about the VLANs from the VTP server. The only configured VTP parameter is the VTP domain name (Larry). The VTP revision number is 201. What happens when an already-running VTP server switch, named Switch11, with domain name Larry and revision number 301, connects via a trunk to any of the other ten switches?

   a. No VLAN information changes; all VTP updates between the original VTP domain and the new switch are ignored.

   b. The original ten switches replace their old VLAN configuration with the configuration in Switch11.

   c. Switch11 replaces its old VLAN configuration with the configuration sent to it by one of the original VTP servers.

   d. Switch11 merges its existing VLAN database with the database learned from the VTP servers, because Switch11 had a higher revision number.

   e. None of the other answers is correct.

6. Assume that two brand-new Cisco switches were removed from their cardboard boxes. PC1 was attached to one switch, PC2 was attached to the other, and the two switches were connected with a cross-over cable. The switch connection dynamically formed an 802.1Q trunk. When PC1 sends a frame to PC2, how many additional bytes of header are added to the frame before it passes over the trunk?

   a. 0

   b. 4

   c. 8

   d. 26

7. Assume that two brand-new Cisco Catalyst 3550 switches were connected with a cross-over cable. Before attaching the cable, one switch interface was configured with the **switchport trunk encapsulation dot1q**, **switchport mode trunk**, and **switchport nonegotiate** subcommands. Which of the following must be configured on the other switch before trunking will work between the switches?

   a. **switchport trunk encapsulation dot1q**

   b. **switchport mode trunk**

   c. **switchport nonegotiate**

   d. No configuration is required.

8. When configuring trunking on a Cisco router fa0/1 interface, under which configuration modes could the IP address associated with the native VLAN (VLAN 1 in this case) be configured?

   a. Interface fa 0/1 configuration mode
   b. Interface fa 0/1.1 configuration mode
   c. Interface fa 0/1.2 configuration mode
   d. None of the other answers is correct

9. Which of the following is false about 802.1Q?

   a. Encapsulates the entire frame inside an 802.1Q header and trailer
   b. Supports the use of a native VLAN
   c. Allows VTP to operate only on extended-range VLANs
   d. Is chosen over ISL by DTP

Foundation Topics

Virtual LANs

In an Ethernet LAN, a set of devices that receive a broadcast sent by any one of the devices in the same set is called a *broadcast domain*. On switches that have no concept of virtual LANs (VLAN), a switch simply forwards all broadcasts out all interfaces, except the interface on which it received the frame. As a result, all the interfaces on an individual switch are in the same broadcast domain. Also, if the switch connects to other switches and hubs, the interfaces on those switches and hubs are also in the same broadcast domain.

A *VLAN* is simply an administratively defined subset of switch ports that are in the same broadcast domain. Ports can be grouped into different VLANs on a single switch, and on multiple interconnected switches as well. By creating multiple VLANs, the switches create multiple broadcast domains. By doing so, a broadcast sent by a device in one VLAN is forwarded to the other devices in that same VLAN; however, the broadcast is not forwarded to devices in the other VLANs.

KEY POINT

With VLANs and IP, best practices dictate a one-to-one relationship between VLANs and IP subnets. Simply put, the devices in a single VLAN are typically also in the same single IP subnet. Alternately, it is possible to put multiple subnets in one VLAN, and use secondary IP addresses on routers to route between the VLANs and subnets. Also, although not typically done, you can design a network to use one subnet on multiple VLANs, and use routers with proxy ARP enabled to forward traffic between hosts in those VLANs. (Private VLANs might be considered to consist of one subnet over multiple VLANs as well, as covered later in this chapter.) Ultimately, the CCIE written exams tend to focus more on the best use of technologies, so this book will assume that one subnet sits on one VLAN, unless otherwise stated.

Layer 2 switches forward frames between devices in the same VLAN, but they do not forward frames between two devices in different VLANs. To forward data between two VLANs, a multilayer switch (MLS) or router is needed. Chapter 7, "IP Forwarding (Routing)," covers the details of MLS.

VLAN Configuration

Configuring VLANs in a network of Cisco switches requires just a few simple steps:

Step 1 Create the VLAN itself.

Step 2 Associate the correct ports with that VLAN.

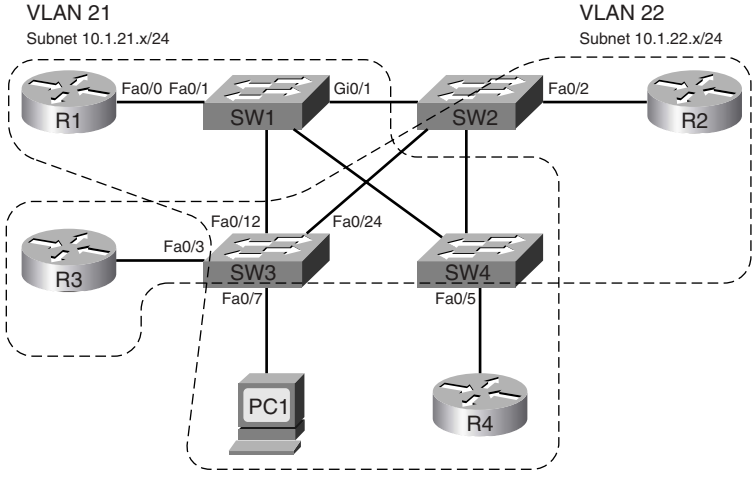
The challenge relates to how some background tasks differ depending on how the Cisco *VLAN Trunking Protocol (VTP)* is configured, and whether normal-range or extended-range VLANs are being used.

Using VLAN Database Mode to Create VLANs

To begin, consider Example 2-1, which shows some of the basic mechanics of VLAN creation in *VLAN database configuration mode*. VLAN database configuration mode allows the creation of VLANs, basic administrative settings for each VLAN, and verification of VTP configuration information. Only normal-range (VLANs 1–1005) VLANs can be configured in this mode, and the VLAN configuration is stored in a Flash file called VLAN.DAT.

Example 2-1 demonstrates VLAN database configuration mode, showing the configuration on Switch3 from Figure 2-1. The example shows VLANs 21 and 22 being created.

Figure 2-1 *Simple Access and Distribution*



Example 2-1 *VLAN Creation in VLAN Database Mode–Switch3*

```
! Below, note that FA 0/12 and FA0/24 missing from the list, because they have
! dynamically become trunks, supporting multiple VLANs.
Switch3# show vlan brief
VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                Fa0/9, Fa0/10, Fa0/11, Fa0/13
                                                Fa0/14, Fa0/15, Fa0/16, Fa0/17
                                                Fa0/18, Fa0/19, Fa0/20, Fa0/21
                                                Fa0/22, Fa0/23
```

Example 2-1 *VLAN Creation in VLAN Database Mode–Switch3 (Continued)*

```
! Below, "unsup" means that this 2950 switch does not support FDDI and TR
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup
! Below, vlan database moves user to VLAN database configuration mode.
! The vlan 21 command defines the VLAN, as seen in the next command output
! (show current), VLAN 21 is not in the "current" VLAN list.
Switch3# vlan database
Switch3(vlan)# vlan 21
VLAN 21 added:
    Name: VLAN0021
! The show current command lists the VLANs available to the IOS when the switch
! is in VTP Server mode. The command lists the VLANs in numeric order, with
! VLAN 21 missing.
Switch3(vlan)# show current
  VLAN ISL Id: 1
    Name: default
    Media Type: Ethernet
    VLAN 802.10 Id: 100001
    State: Operational
    MTU: 1500
    Backup CRF Mode: Disabled
    Remote SPAN VLAN: No

  VLAN ISL Id: 1002
    Name: fddi-default
    Media Type: FDDI
    VLAN 802.10 Id: 101002
    State: Operational
    MTU: 1500
    Backup CRF Mode: Disabled
    Remote SPAN VLAN: No
! Lines omitted for brevity
! Next, note that show proposed lists VLAN 21. The vlan 21 command
! creates the definition, but it must be "applied" before it is "current".
Switch3(vlan)# show proposed
  VLAN ISL Id: 1
    Name: default
    Media Type: Ethernet
    VLAN 802.10 Id: 100001
    State: Operational
    MTU: 1500
    Backup CRF Mode: Disabled
    Remote SPAN VLAN: No
```

continues

Example 2-1 *VLAN Creation in VLAN Database Mode–Switch3 (Continued)*

```
  VLAN ISL Id: 21
    Name: VLAN0021
    Media Type: Ethernet
    VLAN 802.10 Id: 100021
    State: Operational
    MTU: 1500
    Backup CRF Mode: Disabled
    Remote SPAN VLAN: No
! Lines omitted for brevity
! Next, you could apply to complete the addition of VLAN 21,
! abort to not make the changes and exit VLAN database mode, or
! reset to not make the changes but stay in VLAN database mode.
Switch3(vlan)# ?
VLAN database editing buffer manipulation commands:
  abort  Exit mode without applying the changes
  apply  Apply current changes and bump revision number
  exit   Apply changes, bump revision number, and exit mode
  no     Negate a command or set its defaults
  reset  Abandon current changes and reread current database
  show   Show database information
  vlan   Add, delete, or modify values associated with a single VLAN
  vtp    Perform VTP administrative functions.
! The apply command was used, making the addition of VLAN 21 complete.
Switch3(vlan)# apply
APPLY completed.
! A show current now would list VLAN 21.
Switch3(vlan)# vlan 22 name ccie-vlan-22
VLAN 22 added:
    Name: ccie-vlan-22
! Above and below, some variations on commands are shown, along with the
! creation of VLAN 22, with name ccie-vlan-22.
! Below, the vlan 22 option is used on show current and show proposed
! detailing the fact that the apply has not been done yet.
Switch3(vlan)# show current 22
VLAN 22 does not exist in current database
Switch3(vlan)# show proposed 22
  VLAN ISL Id: 22
! Lines omitted for brevity
! Finally, the user exits VLAN database mode using CTRL-Z, which does
! not inherently apply the change. CTRL-Z actually executes an abort.
Switch3(vlan)# ^Z
```

KEY POINT

Using Configuration Mode to Put Interfaces into VLANs

To make a VLAN operational, the VLAN must be created, and then switch ports must be assigned to the VLAN. Example 2-2 shows how to associate the interfaces with the correct VLANs, once again on Switch3.

NOTE At the end of Example 2-1, VLAN 22 had not been successfully created. The assumption for Example 2-2 is that VLAN 22 has been successfully created.

Example 2-2 *Assigning Interfaces to VLANs–Switch3*

```
! First, the switchport access command assigns the VLAN numbers to the
! respective interfaces.

Switch3# config t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch3(config)# int fa 0/3
Switch3(config-if)# switchport access vlan 22
Switch3(config-if)# int fa 0/7
Switch3(config-if)# switchport access vlan 21
Switch3(config-if)# ^Z
! Below, show vlan brief lists these same two interfaces as now being in
! VLANs 21 and 22, respectively.
Switch3# show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/1, Fa0/2, Fa0/4, Fa0/5
                                                Fa0/6, Fa0/8, Fa0/9, Fa0/10
                                                Fa0/11, Fa0/13, Fa0/14, Fa0/15
                                                Fa0/16, Fa0/17, Fa0/18, Fa0/19
                                                Fa0/20, Fa0/21, Fa0/22, Fa0/23
21   VLAN0021                         active    Fa0/7
22   ccie-vlan-22                     active    Fa0/3
! Lines omitted for brevity
! While the VLAN configuration is not shown in the running-config at this point,
! the switchport access command that assigns the VLAN for the interface is in the
! configuration, as seen with the show run int fa 0/3 command.
Switch3# show run int fa 0/3
interface FastEthernet0/3
switchport access vlan 22
```

Using Configuration Mode to Create VLANs

At this point, the two new VLANs (21 and 22) have been created on Switch3, and the two interfaces are now in the correct VLANs. However, Cisco IOS switches support a different way to create VLANs, using configuration mode, as shown in Example 2-3.

Example 2-3 *Creating VLANs in Configuration Mode–Switch3*

KEY POINT

```
! First, VLAN 31 did not exist when the switchport access vlan 31 command was
! issued. As a result, the switch both created the VLAN and put interface fa0/8
! into that VLAN. Then, the vlan 32 global command was used to create a
```

continues

Example 2-3 *Creating VLANs in Configuration Mode–Switch3 (Continued)*

```
! VLAN from configuration mode, and the name subcommand was used to assign a
! non-default name.
Switch3# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch3(config)# int fa 0/8
Switch3(config-if)# switchport access vlan 31
% Access VLAN does not exist. Creating vlan 31
Switch3(config-if)# exit
Switch3(config)# vlan 32
Switch3(config-vlan)# name ccie-vlan-32
Switch3(config-vlan)# ^Z
Switch3# show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/1, Fa0/2, Fa0/4, Fa0/5
                                                Fa0/6, Fa0/9, Fa0/10, Fa0/11
                                                Fa0/13, Fa0/14, Fa0/15, Fa0/16
                                                Fa0/17, Fa0/18, Fa0/19, Fa0/20
                                                Fa0/21, Fa0/22, Fa0/23
21   VLAN0021                         active    Fa0/7
22   ccie-vlan-22                     active    Fa0/3
31   VLAN0031                         active    Fa0/8
32   ccie-vlan-32                     active
! Portions omitted for brevity
```

Example 2-3 shows how the **switchport access vlan** subcommand creates the VLAN, as needed, and assigns the interface to that VLAN. Note that in Example 2-3, the **show vlan brief** output lists fa0/8 as being in VLAN 31. Because no ports have been assigned to VLAN 32 as of yet, the final line in Example 2-3 simply does not list any interfaces.

The VLAN creation process is simple but laborious in a large network. If many VLANs exist, and they exist on multiple switches, instead of manually configuring the VLANs on each switch, you can use VTP to distribute the VLAN configuration of a VLAN to the rest of the switches. VTP will be discussed after a brief discussion of private VLANs.

Private VLANs

Engineers may design VLANs with many goals in mind. In many cases today, devices end up in the same VLAN just based on the physical locations of the wiring drops. Security is another motivating factor in VLAN design: devices in different VLANs do not overhear each other's

broadcasts. Additionally, the separation of hosts into different VLANs and subnets requires an intervening router or multilayer switch between the subnets, and these types of devices typically provide more robust security features.

Regardless of the design motivations behind grouping devices into VLANs, good design practices typically call for the use of a single IP subnet per VLAN. In some cases, however, the need to increase security by separating devices into many small VLANs conflicts with the design goal of conserving the use of the available IP subnets. The Cisco private VLAN feature addresses this issue. Private VLANs allow a switch to separate ports as if they were on different VLANs, while consuming only a single subnet.

A common place to implement private VLANs is in the multitenant offerings of a service provider (SP). The SP can install a single router and a single switch. Then, the SP attaches devices from multiple customers to the switch. Private VLANs then allow the SP to use only a single subnet for the whole building, separating different customers' switch ports so that they cannot communicate directly, while supporting all customers with a single router and switch.

Conceptually, a private VLAN includes the following general characterizations of how ports communicate:

- Ports that need to communicate with all devices
- Ports that need to communicate with each other, and with shared devices, typically routers
- Ports that need to communicate only with shared devices

To support each category of allowed communications, a single private VLAN features a *primary VLAN* and one or more *secondary VLANs*. The ports in the primary VLAN are *promiscuous* in that they can send and receive frames with any other port, including ports assigned to secondary VLANs. Commonly accessed devices, such as routers and servers, are placed into the primary VLAN. Other ports, such as customer ports in the SP multitenant model, attach to one of the secondary VLANs.

Secondary VLANs are either *community VLANs* or *isolated VLANs*. The engineer picks the type based on whether the device is part of a set of ports that should be allowed to send frames back and forth (community VLAN ports), or whether the device port should not be allowed to talk to any other ports besides those on the primary VLAN (isolated VLAN). Table 2-2 summarizes the behavior of private VLAN communications between ports.

Table 2-2 *Private VLAN Communications Between Ports*

KEY POINT	Description of Who can Talk to Whom	Primary VLAN Ports	Community VLAN Ports[1]	Isolated VLAN Ports[1]
	Talk to ports in primary VLAN (promiscuous ports)	Yes	Yes	Yes
	Talk to ports in the same secondary VLAN (host ports)	N/A[2]	Yes	No
	Talks to ports in another secondary VLAN	N/A[2]	No	No

[1]Community and isolated VLANs are secondary VLANs.

[2]Promiscuous ports, by definition in the primary VLAN, can talk to all other ports.

VLAN Trunking Protocol

VTP advertises VLAN configuration information to neighboring switches so that the VLAN configuration can be made on one switch, with all the other switches in the network learning the VLAN information dynamically. VTP advertises the VLAN ID, VLAN name, and VLAN type for each VLAN. However, VTP does not advertise any information about which ports (interfaces) should be in each VLAN, so the configuration to associate a switch interface with a particular VLAN (using the **switchport access vlan** command) must still be configured on each individual switch. Also, the existence of the VLAN IDs used for private VLANs is advertised, but the rest of the detailed private VLAN configuration is not advertised by VTP.

Each Cisco switch uses one of three VTP modes, as outlined in Table 2-3.

Table 2-3 *VTP Modes and Features**

KEY POINT	Function	Server Mode	Client Mode	Transparent Mode
	Originates VTP advertisements	Yes	No	No
	Processes received advertisements in order to update its VLAN configuration	Yes	Yes	No
	Forwards received VTP advertisements	Yes	Yes	Yes
	Saves VLAN configuration in NVRAM or VLAN.DAT	Yes	No	Yes
	Can create, modify, or delete VLANs using configuration commands	Yes	No	Yes

*CatOS switches support a fourth VTP mode (off), meaning that the switch does not create, listen to, or forward VTP updates.

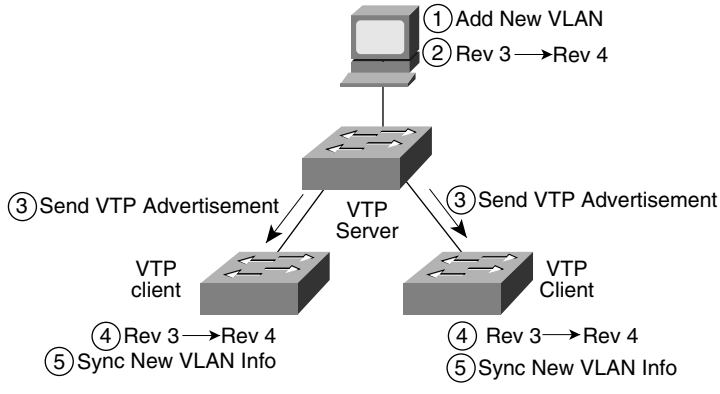
VTP Process and Revision Numbers

The VTP update process begins when a switch administrator, from a VTP server switch, adds, deletes, or updates the configuration for a VLAN. When the new configuration occurs, the VTP server increments the old VTP *revision number* by 1, and advertises the entire VLAN configuration database along with the new revision number.

The VTP revision number concept allows switches to know when VLAN database changes have occurred. Upon receiving a VTP update, if the revision number in a received VTP update is larger than a switch's current revision number, it believes that there is a new version of the VLAN database. Figure 2-2 shows an example in which the old VTP revision number was 3, the server adds a new VLAN, incrementing the revision number to 4, and then propagates the VTP database to the other switches.

Figure 2-2 *VTP Revision Number Basic Operation*

KEY POINT



Cisco switches default to use VTP server mode, but they do not start sending VTP updates until the switch has been configured with a VTP domain name. At that point, the server begins to send its VTP updates, with a different database and revision number each time its VLAN configuration changes. However, the VTP clients in Figure 2-2 actually do not have to have the VTP domain name configured. If not configured, the client will assume it should use the VTP domain name in the first received VTP update. However, the client does need one small bit of configuration, namely, the VTP mode, as configured with the **vtp mode** global configuration command.

VTP clients and servers alike will accept VTP updates from other VTP server switches. When using VTP, for better availability, a switched network using VTP needs at least two VTP server switches. Under normal operations, a VLAN change could be made on one server switch, and the other VTP server (plus all the clients) would learn about the changes to the VLAN database. Once learned, VTP servers would then store the VLAN configuration permanently (for example, in NVRAM), whereas clients do not permanently store the configuration.

The support of multiple VTP servers introduces the possibility of accidentally changing the VLAN configuration for the network. When a VTP client or VTP transparent switch first connects via a trunk to the network, it cannot affect the existing configuration, because in these modes the switch does not originate VTP updates. However, if a newly added switch in VTP server mode attaches via a trunk, it has the capability to change the other switch VLAN configurations with its own. It will change the other switch VLAN configurations, assuming the following are true about the newly added switch:

KEY POINT

- The new link connecting the new switch is trunking.
- The new switch has the same VTP domain name as the other switches.
- The new switch's revision number is larger than that of the existing switches.
- Same password, if configured on the existing switches.

The revision number and VTP domain name can be easily seen with a Sniffer trace; to prevent DoS attacks with VTP, set VTP passwords, which are encoded as message digests (MD5) in the VTP updates. Also, some installations simply use VTP transparent mode on all switches, which prevents switches from ever listening to other switch VTP updates and erroneously deleting their VLAN configuration databases.

VTP Configuration

VTP sends updates out all active trunk interfaces (ISL or 802.1Q). However, with all default settings from Cisco, switches are in server mode, with no VTP domain name configured, and they do not send any VTP updates. Before any switches can learn VLAN information from another switch, at least one switch must have a bare-minimum VTP server configuration—specifically, a domain name.

Example 2-4 shows Switch3 configuring a VTP domain name to become a VTP server and advertise the VLANs it has configured. The example also lists several key VTP **show** commands. (Note that the example begins with VLANs 21 and 22 configured on Switch3, and all default settings for VTP on all four switches.)

Example 2-4 *VTP Configuration and* **show** *Command Example*

```
! First, Switch3 is configured with a VTP domain ID of CCIE-domain.
Switch3# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch3(config)# vtp domain CCIE-domain
Changing VTP domain name from NULL to CCIE-domain
Switch3(config)# ^Z
```

```
! Next, on Switch1, the VTP status shows the same revision as Switch3, and it
! learned the VTP domain name CCIE-domain. Note that Switch1 has no VTP-related
```

Example 2-4 *VTP Configuration and* **show** *Command Example (Continued)*

```
! configuration, so it is a VTP server; it learned the VTP domain name from.
! Switch3.
Switch1# sh vtp status
VTP Version                     : 2
Configuration Revision          : 2
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 7
VTP Operating Mode              : Server
VTP Domain Name                 : CCIE-domain
VTP Pruning Mode                : Disabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x0E 0x07 0x9D 0x9A 0x27 0x10 0x6C 0x0B
Configuration last modified by 10.1.1.3 at 3-1-93 00:02:55
Local updater ID is 10.1.1.1 on interface Vl1 (lowest numbered VLAN interface found)
! The show vlan brief command lists the VLANs learned from Switch3.
Switch1# show vlan brief
VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/6, Fa0/7, Fa0/10
                                                Fa0/11, Fa0/13, Fa0/14, Fa0/15
                                                Fa0/16, Fa0/17, Fa0/18, Fa0/19
                                                Fa0/20, Fa0/21, Fa0/22, Fa0/23
                                                Gi0/2
21   VLAN0021                         active
22   ccie-vlan-22                     active
1002 fddi-default                     active
1003 token-ring-default               active
1004 fddinet-default                  active
1005 trnet-default                    active
```

Example 2-4 shows examples of a few VTP configuration options. Table 2-4 provides a complete list, along with explanations.

Table 2-4 *VTP Configuration Options*

Option	Meaning
domain	Sends domain name in VTP updates. Received VTP update is ignored if it does not match a switch's domain name. One VTP domain name per switch is allowed.
password	Used to generate an MD5 hash that is included in VTP updates. Received VTP updates are ignored if the passwords on the sending and receiving switch do not match.
mode	Sets server, client, or transparent mode on the switch.

continues

Table 2-4 *VTP Configuration Options (Continued)*

Option	Meaning
version	Sets version 1 or 2. Servers and clients must match version to exchange VLAN configuration data. Transparent mode switches at version 2 forward version 1 or version 2 VTP updates.
pruning	Enables VTP pruning, which prevents broadcasts from being propagated on a per-VLAN basis to switches that do not have any ports configured as members of that VLAN.
interface	Specifies from which interface a switch picks the source MAC address for VTP updates.

Normal-Range and Extended-Range VLANs

Some VLAN numbers are considered to be *normal*, whereas some others are considered to be *extended*. Normal-range VLANs are VLANs 1–1005, and can be advertised via VTP versions 1 and 2. These VLANs can be configured in VLAN database mode, with the details being stored in the VLAN.DAT file in Flash.

Extended-range VLANs range from 1006–4094, inclusive. However, these additional VLANs cannot be configured in VLAN database mode, nor stored in the VLAN.DAT file, nor advertised via VTP. In fact, to configure them, the switch must be in VTP transparent mode. (Also, you should take care to avoid using VLANs 1006–1024 for compatibility with CatOS-based switches.)

Both ISL and 802.1Q support extended-range VLANs today. Originally, ISL began life only supporting normal-range VLANs, using only 10 of the 15 bits reserved in the ISL header to identify the VLAN ID. The later-defined 802.1Q used a 12-bit VLAN ID field, thereby allowing support of the extended range. Following that, Cisco changed ISL to use 12 of its reserved 15 bits in the VLAN ID field, thereby supporting the extended range.

Table 2-5 summarizes VLAN numbers and provides some additional notes.

Table 2-5 *Valid VLAN Numbers, Normal and Extended*

KEY POINT

VLAN Number	Normal or Extended?	Can be Advertised and Pruned by VTP Versions 1 and 2?	Comments
0	Reserved	—	Not available for use
1	Normal	No	On Cisco switches, the default VLAN for all access ports; cannot be deleted or changed
2–1001	Normal	Yes	

Table 2-5 *Valid VLAN Numbers, Normal and Extended (Continued)*

VLAN Number	Normal or Extended?	Can be Advertised and Pruned by VTP Versions 1 and 2?	Comments
1002–1005	Normal	No	Defined specifically for use with FDDI and TR translational bridging
1006–4094	Extended	No	

Storing VLAN Configuration

Catalyst IOS stores VLAN and VTP configuration in one of two places—either in a Flash file called VLAN.DAT or in the running configuration. (Remember that the term "Catalyst IOS" refers to a switch that uses IOS, not the Catalyst OS, which is often called CatOS.) IOS chooses the location for the configuration information based in part on whether the switch is in VTP server or transparent mode, and in part based on whether the VLANs are normal-range VLANs or extended-range VLANs. Table 2-6 describes what happens based on what configuration mode is used to configure the VLANs, the VTP mode, and the VLAN range.

Table 2-6 *VLAN Configuration and Storage*

KEY POINT

Function	When in VTP Server Mode	When in VTP Transparent Mode
Normal-range VLANs can be configured from	Both VLAN database and configuration modes	Both VLAN database and configuration modes
Extended-range VLANs can be configured from	Nowhere—cannot be configured	Configuration mode only
VTP and normal-range VLAN configuration commands are stored in	VLAN.DAT in Flash	Both VLAN.DAT in Flash and running configuration[1]
Extended-range VLAN configuration commands stored in	Nowhere—extended range not allowed in VTP server mode	Running configuration only

[1]When a switch reloads, if the VTP mode or domain name in the VLAN.DAT file and the startup-config file differ, the switch uses only the VLAN.DAT file's contents for VLAN configuration.

NOTE The configuration characteristics referenced in Table 2-6 do not include the interface configuration command **switchport access vlan**; it includes the commands that create a VLAN (**vlan** command) and VTP configuration commands.

Of particular interest for those of you stronger with CatOS configuration skills is that when you erase the startup-config file, and reload the Cisco IOS switch, you do not actually erase the normal-

range VLAN and VTP configuration information. To erase the VLAN and VTP configuration, you must use the **delete flash:vlan.dat** exec command. Also note that if multiple switches are in VTP server mode, if you delete VLAN.DAT on one switch and then reload it, as soon as the switch comes back up and brings up a trunk, it learns the old VLAN database via a VTP update from the other VTP server.

VLAN Trunking: ISL and 802.1Q

VLAN trunking allows switches, routers, and even PCs with the appropriate NICs to send traffic for multiple VLANs across a single link. In order to know to which VLAN a frame belongs, the sending switch, router, or PC adds a header to the original Ethernet frame, with that header having a field in which to place the VLAN ID of the associated VLAN. This section describes the protocol details for the two trunking protocols, followed by the details of how to configure trunking.

ISL and 802.1Q Concepts

If two devices are to perform trunking, they must agree to use either ISL or 802.1Q, because there are several differences between the two, as summarized in Table 2-7.

Table 2-7 *Comparing ISL and 802.1Q*

KEY POINT

Feature	ISL	802.1Q
VLANs supported	Normal and extended range[1]	Normal and extended range
Protocol defined by	Cisco	IEEE
Encapsulates original frame or inserts tag	Encapsulates	Inserts tag
Supports native VLAN	No	Yes

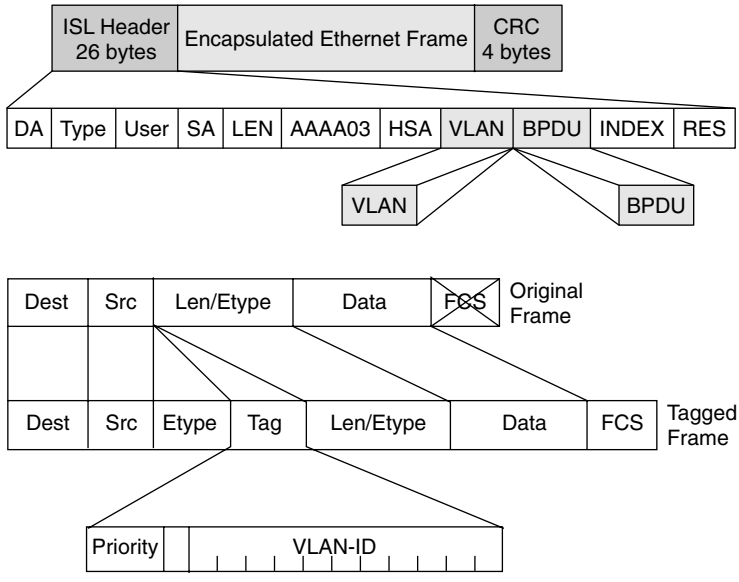
[1]ISL originally supported only normal-range VLANs, but was later improved to support extended-range VLANs as well.

ISL and 802.1Q differ in how they add a header to the Ethernet frame before sending it over a trunk. ISL adds a new 26-byte header, plus a new trailer (to allow for the new FCS value), encapsulating the original frame. This encapsulating header uses the source address (listed as SA in Figure 2-3) of the device doing the trunking, instead of the source MAC of the original frame. ISL uses a multicast destination address (listed as DA in Figure 2-3) of either 0100.0C00.0000 or 0300.0C00.0000.

802.1Q inserts a 4-byte header, called a tag, into the original frame (right after the Source Address field). The original frame's addresses are left intact. Normally, an Ethernet controller would expect to find either an Ethernet Type field or 802.3 Length field right after the Source Address field. With an 802.1Q tag, the first 2 bytes after the Address fields holds a registered Ethernet type value of 0x8100, which implies that the frame includes an 802.1Q header. Because 802.1Q does not actually encapsulate the original frame, it is often called *frame tagging*. Figure 2-3 shows the contents of the headers used by both ISL and 802.1Q.

Figure 2-3 *ISL and 802.1Q Frame Marking Methods*

KEY POINT



Finally, the last row from Table 2-7 refers to the *native VLAN*. 802.1Q does not tag frames sent inside the native VLAN. The native VLAN feature allows a switch to attempt to use 802.1Q trunking on an interface, but if the other device does not support trunking, the traffic for that one native VLAN can still be sent over the link. By default, the native VLAN is VLAN 1.

ISL and 802.1Q Configuration

Cisco switches use the *Dynamic Trunk Protocol (DTP)* to dynamically learn whether the device on the other end of the cable wants to perform trunking and, if so, which trunking protocol to use. DTP learns whether to trunk based on the DTP mode defined for an interface. Cisco switches default to use the DTP *desirable* mode, which means that the switch initiates sending DTP messages, hoping that the device on the other end of the segment replies with another DTP message. If a reply is received, DTP can detect whether both switches can trunk and, if so, which type of trunking to use. If both switches support both types of trunking, they choose to use ISL. (An upcoming section, "Trunk Configuration Compatibility," covers the different DTP modes and how they work.)

With the DTP mode set to desirable, switches can simply be connected, and they should dynamically form a trunk. You can, however, configure trunking details and verify the results with **show** commands. Table 2-8 lists some of the key Catalyst IOS commands related to trunking.

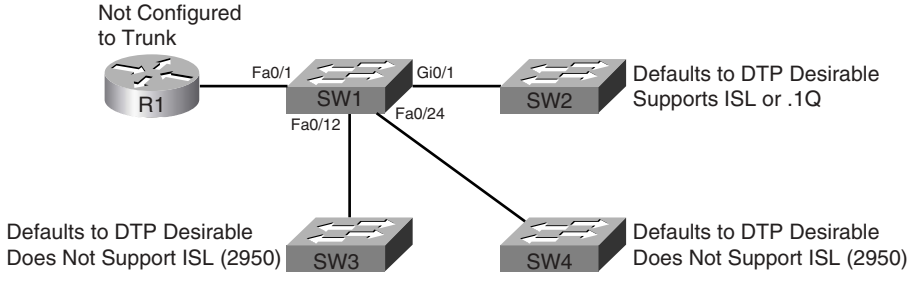
Table 2-8 *VLAN Trunking–Related Commands*

KEY POINT

Command	Function
switchport | **no switchport**	Toggle defining whether to treat the interface as a switch interface (**switchport**) or as a router interface (**no switchport**)
switchport mode	Sets DTP negotiation parameters
switchport trunk	Sets trunking parameters if the interface is trunking
switchport access	Sets nontrunking-related parameters if the interface is not trunking
show interface trunk	Summary of trunk-related information
show interface *type number* **trunk**	Lists trunking details for a particular interface
show interface *type number* **switchport**	Lists nontrunking details for a particular interface

Figure 2-4 lists several details regarding Switch1's trunking configuration and status, as shown in Example 2-5. R1 is not configured to trunk, so Switch1 will fail to negotiate trunking. Switch2 is a Catalyst 3550, which supports both ISL and 802.1Q, so they will negotiate trunking and use ISL. Switch3 and Switch4 are Catalyst 2950s, which support only 802.1Q; as a result, Switch1 negotiates trunking, but picks 802.1Q as the trunking protocol.

Figure 2-4 *Trunking Configuration Reference for Example 2-5*



Example 2-5 *Trunking Configuration and* **show** *Command Example–Switch1*

```
! The administrative mode of dynamic desirable (trunking) and negotiate (trunking
! encapsulation) means that Switch1 attempted to negotiate to trunk, but the
! operational mode of static access means that trunking negotiation failed.
! The reference to "operational trunking encapsulation" of native means that
! no tagging occurs.
```

Example 2-5 *Trunking Configuration and* **show** *Command Example–Switch1 (Continued)*

```
Switch1# show int fa 0/1 switchport
Name: Fa0/1
Switchport: Enabled
Administrative Mode: dynamic desirable
Operational Mode: static access
Administrative Trunking Encapsulation: negotiate
Operational Trunking Encapsulation: native
Negotiation of Trunking: On
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (default)
Administrative private-vlan host-association: none
Administrative private-vlan mapping: none
Operational private-vlan: none
Trunking VLANs Enabled: ALL
Pruning VLANs Enabled: 2-1001

Protected: false
Unknown unicast blocked: disabled
Unknown multicast blocked: disabled

Voice VLAN: none (Inactive)
Appliance trust: none
! Next, the show int gig 0/1 trunk command shows the configured mode
! (desirable), and the current status (N-ISL), meaning negotiated ISL. Note
! that the trunk supports the extended VLAN range as well.
Switch1# show int gig 0/1 trunk
Port      Mode         Encapsulation  Status        Native vlan
Gi0/1     desirable    n-isl          trunking      1

Port      Vlans allowed on trunk
Gi0/1     1-4094

Port      Vlans allowed and active in management domain
Gi0/1     1,21-22

Port      Vlans in spanning tree forwarding state and not pruned
Gi0/1     1,21-22
! Next, Switch1 lists all three trunks - the segments connecting to the other
! three switches - along with the type of encapsulation.
Switch1# show int trunk
Port      Mode         Encapsulation  Status        Native vlan
Fa0/12    desirable    n-802.1q       trunking      1
Fa0/24    desirable    n-802.1q       trunking      1
Gi0/1     desirable    n-isl          trunking      1

Port      Vlans allowed on trunk
Fa0/12    1-4094
```

continues

Example 2-5 *Trunking Configuration and* **show** *Command Example–Switch1 (Continued)*

```
Fa0/24    1-4094
Gi0/1     1-4094

Port      Vlans allowed and active in management domain
Fa0/12    1,21-22
Fa0/24    1,21-22
Gi0/1     1,21-22

Port      Vlans in spanning tree forwarding state and not pruned
Fa0/12    1,21-22
Fa0/24    1,21-22
Gi0/1     1,21-22
```

Allowed, Active, and Pruned VLANs

Although a trunk can support VLANs 1–4094, several mechanisms reduce the actual number of VLANs whose traffic flows over the trunk. First, VLANs can be administratively forbidden from existing over the trunk using the **switchport trunk allowed** interface subcommand. Also, any allowed VLANs must be configured on the switch before they are considered active on the trunk. Finally, VTP can prune VLANs from the trunk, with the switch simply ceasing to forward frames from that VLAN over the trunk.

The **show interface trunk** command lists the VLANs that fall into each category, as shown in the last command in Example 2-5. The categories are summarized as follows:

KEY POINT

- **Allowed VLANs**—Each trunk allows all VLANs by default. However, VLANs can be removed or added to the list of allowed VLANs by using the **switchport trunk allowed** command.
- **Allowed and active**—To be active, a VLAN must be in the allowed list for the trunk (based on trunk configuration), and the VLAN must exist in the VLAN configuration on the switch. With PVST+, an STP instance is actively running on this trunk for the VLANs in this list.
- **Active and not pruned**—This list is a subset of the "allowed and active" list, with any VTP-pruned VLANs removed.

Trunk Configuration Compatibility

In most production networks, switch trunks are configured using the same standard throughout the network. For instance, rather than allow DTP to negotiate trunking,, many engineers configure trunk interfaces to always trunk (**switchport mode trunk**) and disable DTP on ports that should not trunk. IOS includes several commands that impact whether a particular segment becomes a trunk. Because many enterprises use a typical standard, it is easy to forget the nuances of how the related commands work. This section covers those small details.

Two IOS configuration commands impact if and when two switches form a trunk. The **switchport mode** and **switchport nonegotiate** interface subcommands define whether DTP even attempts to negotiate a trunk, and what rules it uses when the attempt is made. Additionally, the settings on the switch ports on either side of the segment dictate whether a trunk forms or not.

Table 2-9 summarizes the trunk configuration options. The first column suggests the configuration on one switch, with the last column listing the configuration options on the other switch that would result in a working trunk between the two switches.

Table 2-9 *Trunking Configuration Options That Lead to a Working Trunk*

KEY POINT

Configuration Command on One Side[1]	Short Name	Meaning	To Trunk, Other Side Must Be
switchport mode trunk	Trunk	Always trunks on this end; sends DTP to help other side choose to trunk	On, desirable, auto
switchport mode trunk; **switchport nonegotiate**	Nonegotiate	Always trunks on this end; does not send DTP messages (good when other switch is a non-Cisco switch)	On
switchport mode dynamic desirable	Desirable	Sends DTP messages, and trunks if negotiation succeeds	On, desirable, auto
switchport mode dynamic auto	Auto	Replies to DTP messages, and trunks if negotiation succeeds	On, desirable
switchport mode access	Access	Never trunks; sends DTP to help other side reach same conclusion	(Never trunks)
switchport mode access; **switchport nonegotiate**	Access (with nonegotiate)	Never trunks; does not send DTP messages	(Never trunks)

[1]When the **switchport nonegotiate** command is not listed in the first column, the default (DTP negotiation is active) is assumed.

NOTE If an interface trunks, then the type of trunking (ISL or 802.1Q) is controlled by the setting on the **switchport trunk encapsulation** command. This command includes an option for dynamically negotiating the type (using DTP) or configuring one of the two types. See Example 2-5 for a sample of the syntax.

Configuring Trunking on Routers

VLAN trunking can be used on routers and hosts as well as on switches. However, routers do not support DTP, so you must manually configure them to support trunking. Additionally, you must manually configure a switch on the other end of the segment to trunk, because the router does not participate in DTP.

The majority of router trunking configurations use subinterfaces, with each subinterface being associated with one VLAN. The subinterface number does not have to match the VLAN ID; rather, the **encapsulation** command sits under each subinterface, with the associated VLAN ID being part of the **encapsulation** command. Also, because good design calls for one IP subnet per VLAN, if the router wants to forward IP packets between the VLANs, the router needs to have an IP address associated with each trunking subinterface.

You can configure 802.1Q native VLANs under a subinterface or under the physical interface on a router. If configured under a subinterface, you use the **encapsulation dot1q** *vlan-id* **native** subcommand, with the inclusion of the **native** keyword meaning that frames exiting this subinterface should not be tagged. As with other router trunking configurations, the associated IP address would be configured on that same subinterface. Alternately, if not configured on a subinterface, the router assumes that the native VLAN is associated with the physical interface. In this case, the **encapsulation** command is not needed under the physical interface; the associated IP address, however, would need to be configured under the physical interface.

Example 2-6 shows an example configuration for Router1 in Figure 2-1, both for ISL and 802.1Q. In this case, Router1 needs to forward packets between the subnets on VLANs 21 and 22. The first part of the example shows ISL configuration, with no native VLANs, and therefore only a subinterface being used for each VLAN. The second part of the example shows an alternative 802.1Q configuration, using the option of placing the native VLAN (VLAN 21) configuration on the physical interface.

Example 2-6 *Trunking Configuration on Router1*

```
! Note the subinterface on the fa 0/0 interface, with the encapsulation
! command noting the type of trunking, as well as the VLAN number. The
! subinterface does not have to be the VLAN ID. Also note the IP addresses for
! each interface, allowing Router1 to route between VLANs.
interface fastethernet 0/0.1
 ip address 10.1.21.1 255.255.255.0
 encapsulation isl 21
!
interface fastethernet 0/0.2
 ip address 10.1.22.1 255.255.255.0
 encapsulation isl 22
! Next, an alternative 802.1Q configuration is shown. Note that this 802.1Q configuration
! places the IP address
! for VLAN 21 on the physical interface; the router simply associates the
! physical interface with the native VLAN. Alternatively, a subinterface could be
! used, with the encapsulation dot1q 21 native command specifying that the router
! should treat this VLAN as the native VLAN.
interface fastethernet 0/0
  ip address 10.1.21.1 255.255.255.0
!
interface fastethernet 0/0.2
  ip address 10.1.22.1 255.255.255.0
  encapsulation dot1q 22
```

KEY POINT

Note also that the router does not have an explicitly defined allowed VLAN list. However, the allowed VLAN list is implied based on the configured VLANs. For instance, in this example, Router1 allows VLAN 1 (because it cannot be deleted), VLAN 21, and VLAN 22. A **show interface trunk** command on Switch1 would show only 1, 21, and 22 as the allowed VLANs on FA0/1.

802.1Q-in-Q Tunneling

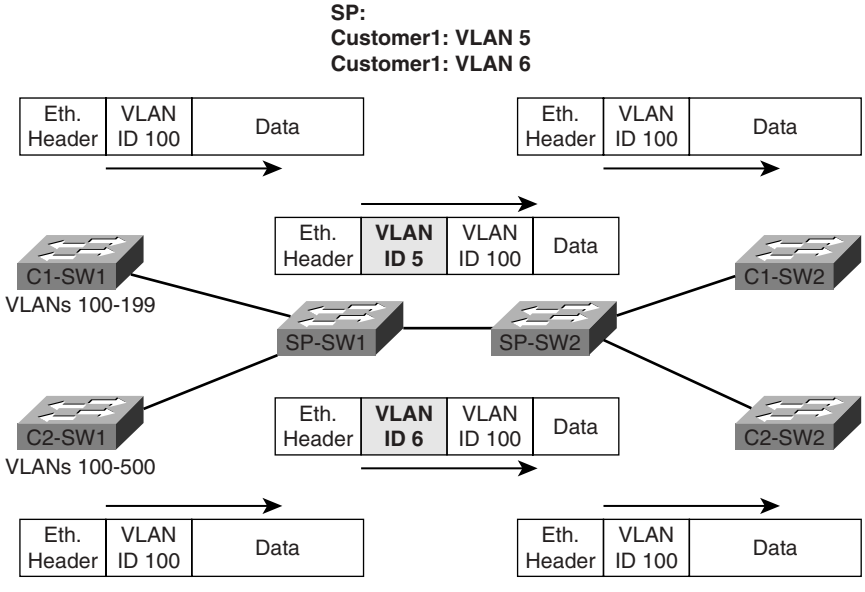
Traditionally, VLANs have not extended beyond the WAN boundary. VLANs in one campus extend to a WAN edge router, but VLAN protocols are not used on the WAN.

Today, several emerging alternatives exist for the passage of VLAN traffic across a WAN, including 802.1Q-in-Q, Ethernet over MPLS (EoMPLS), and VLAN MPLS (VMPLS). While these topics are more applicable to the CCIE Service Provider certification, you should at least know the concept of 802.1 Q-in-Q tunneling.

Also known as Q-in-Q or Layer 2 protocol tunneling, 802.1Q-in-Q allows an SP to preserve 802.1Q VLAN tags across a WAN service. By doing so, VLANs actually span multiple geographically dispersed sites. Figure 2-5 shows the basic idea.

Figure 2-5 *Q-in-Q: Basic Operation*

KEY POINT



The ingress SP switch takes the 802.1Q frame, and then tags each frame entering the interface with an additional 802.1Q header. In this case, all of Customer1's frames are tagged as VLAN 5 as they pass over the WAN; Customer2's frames are tagged with VLAN 6. After removing the tag at egress, the customer switch sees the original 802.1Q frame, and can interpret the VLAN ID correctly. The receiving SP switch (SP-SW2 in this case) can keep the various customers' traffic separate based on the additional VLAN tags.

Using Q-in-Q, an SP can offer VLAN services, even when the customers use overlapping VLAN IDs. Customers get more flexibility for network design options, particularly with metro Ethernet services. Plus, CDP and VTP traffic passes transparently over the Q-in-Q service.

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most of the Cisco Press *Exam Certification Guides*, this book does not repeat information presented in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in the "Foundation Topics" section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 2-10 lists some of the most popular IOS commands related to the topics in this chapter. (The command syntax was copied from the *Catalyst 3550 Multilayer Switch Command Reference, 12.1(20)EA2*. Note that some switch platforms may have differences in the command syntax.)

Table 2-10 *Catalyst IOS Commands Related to Chapter 2*

Command	Description
show mac address-table [**aging-time** | **count** | **dynamic** | **static**] [**address** *hw-addr*] [**interface** *interface-id*] [**vlan** *vlan-id*]	Displays the MAC address table; the security option displays information about the restricted or static settings
show interfaces [*interface-id* | **vlan** *vlan-id*] **switchport** | **trunk**]	Displays detailed information about an interface operating as an access port or a trunk
show vlan [**brief** | **id** *vlan-id* | **name** *vlan-name* | *summary*]	EXEC command that lists information about VLAN
show vlan [*vlan*]	Displays VLAN information
show vtp status	Lists VTP configuration and status information
switchport mode {**access** | **dot1q-tunnel** | **dynamic** {**auto** | **desirable**} | **trunk**}	Configuration command setting nontrunking (**access**), trunking, and dynamic trunking (**auto** and **desirable**) parameters
switchport nonegotiate	Interface subcommand that disables DTP messages; interface must be configured as trunk or access port
switchport trunk {**allowed vlan** *vlan-list*} | {**encapsulation** {**dot1q** | **isl** | **negotiate**}} | {**native vlan** *vlan-id*} | {**pruning vlan** *vlan-list*}	Interface subcommand used to set parameters used when the port is trunking
switchport access vlan *vlan-id*	Interface subcommand that statically configures the interface as a member of that one VLAN

Table 2-11 lists the commands related to VLAN creation—both the VLAN database mode configuration commands (reached with the **vlan database** privileged mode command) and the normal configuration mode commands.

NOTE Some command parameters may not be listed in Table 2-11.

Table 2-11 *Catalyst 3550 VLAN Database and Configuration Mode Command List*

VLAN Database	Configuration
vtp {**domain** *domain-name* | **password** *password* | **pruning** | **v2-mode** | {**server** | **client** | **transparent**}}	**vtp** {**domain** *domain-name* | **file** *filename* | **interface** *name* | **mode** {**client** | **server** | **transparent**} | **password** *password* | **pruning** | **version** *number*}
vlan *vlan-id* [**backupcrf** {**enable** | **disable**}] [**mtu** *mtu-size*] [**name** *vlan-name*] [**parent** *parent-vlan-id*] [**state** {**suspend** | **active**}]	**vlan** *vlan-id*[1]
show {**current** | **proposed** | **difference**}	No equivalent
apply | **abort** | **reset**	No equivalent

[1]Creates the VLAN and places the user in VLAN configuration mode, where commands matching the VLAN database mode options of the **vlan** command are used to set the same parameters.

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

VLAN, broadcast domain, DTP, VTP pruning, 802.1Q, ISL, native VLAN, encapsulation, private VLAN, promiscuous port, community VLAN, isolated VLAN, 802.1Q-in-Q, Layer 2 protocol tunneling

Refer to the CD-based glossary to check your answers.

Further Reading

The topics in this chapter tend to be covered in slightly more detail in CCNP Switching exam preparation books. For more details on these topics, refer to the *CCNP BCMSN Exam Certification Guide* and the *CCNP Self-Study: CCNP BCMSN Exam Certification Guide,* Second Edition, listed in the introduction to this book.

Cisco LAN Switching, by Kennedy Clark and Kevin Hamilton, is an excellent reference for LAN-related topics in general, and certainly very useful for CCIE written and lab exam preparation.

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- Bridging and LAN Switching
- Transparent
- LAN Switching
- Catalyst IOS Configuration Commands

In addition, this chapter covers information related to the following specific CCIE Routing and Switching exam topics:

- IEEE 802.1D Spanning Tree Protocol
- PortFast, UplinkFast, and BackboneFast
- BPDU Guard, Root Guard, UDLD, and Loop Guard
- Cisco PVST+ and IEEE 802.1s MST
- IEEE 802.1w Rapid STP
- EtherChannels

CHAPTER 3

Spanning Tree Protocol

Spanning Tree Protocol (STP) is probably one of the most widely known protocols covered on the CCIE Routing and Switching written exam. STP has been around a long time, is used in most every campus network today, and is covered extensively on the CCNP BCMSN exam. This chapter covers a broad range of topics related to STP.

"Do I Know This Already?" Quiz

Table 3-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 3-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
802.1D Spanning Tree Protocol	1–6	
Optimizing Spanning Tree	7–9	
Protecting STP	10	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. Assume that a non-root 802.1D switch has ceased to receive Hello BPDUs. Which STP setting determines how long a non-root switch waits before trying to choose a new Root Port?

   a. Hello timer setting on the Root

   b. Maxage timer setting on the Root

   c. Forward Delay timer setting on the Root

   d. Hello timer setting on the non-root switch

   e. Maxage timer setting on the non-root switch

   f. Forward Delay timer setting on the non-root switch

2. Assume that a non-root 802.1D switch receives a Hello BPDU with the TCN flag set. Which STP setting determines how long the non-root switch waits before timing out CAM entries?
   a. Hello timer setting on the Root
   b. Maxage timer setting on the Root
   c. Forward Delay timer setting on the Root
   d. Hello timer setting on the non-root switch
   e. Maxage timer setting on the non-root switch
   f. Forward Delay timer setting on the non-root switch

3. Assume that non-root Switch1 (SW1) is blocking on a 802.1Q trunk connected to Switch2 (SW2). Both switches are in the same MST region. SW1 ceases to receive Hellos from SW2. What timers have an impact on how long Switch1 takes to both become the Designated Port on that link and reach forwarding state?
   a. Hello timer setting on the Root
   b. Maxage timer setting on the Root
   c. Forward Delay timer on the Root
   d. Hello timer setting on SW1
   e. Maxage timer setting on SW1
   f. Forward Delay timer on SW1

4. Which of the following statements are true regarding support of multiple spanning trees over an 802.1Q trunk?
   a. Only one common spanning tree can be supported.
   b. Cisco PVST+ supports multiple spanning trees if the switches are Cisco switches.
   c. 802.1Q supports multiple spanning trees when using IEEE 802.1s MST.
   d. Two PVST+ domains can pass over a region of non-Cisco switches using 802.1Q trunks by encapsulating non-native VLAN Hellos inside the native VLAN Hellos.

5. When a switch notices a failure, and the failure requires STP convergence, it notifies the Root by sending a TCN BPDU. Which of the following best describes why the notification is needed?
   a. To speed STP convergence by having the Root converge quickly.
   b. To allow the Root to keep accurate count of the number of topology changes.
   c. To trigger the process that causes all switches to use a short timer to help flush the CAM.
   d. There is no need for TCN today; it is a holdover from DEC's STP specification.

6. Two switches have four parallel Ethernet segments, none of which forms into an EtherChannel. Assuming 802.1D is in use, what is the maximum number of the eight ports (four on each switch) that stabilize into a forwarding state?

   a. 1
   b. 3
   c. 4
   d. 5
   e. 7

7. Two switches have four Ethernet segments connecting them, with the intention of using an EtherChannel. Port fa 0/1 on one switch is connected to port fa0/1 on the other switch; port fa0/2 is connected to the other switch's port fa0/2; and so on. An EtherChannel can still form using these four segments, even though some configuration settings do not match on the corresponding ports on each switch. Which settings do not have to match?

   a. DTP negotiation settings (auto/desirable/on)
   b. Allowed VLAN list
   c. STP per-VLAN port cost on the ports on a single switch
   d. If 802.1Q, native VLAN

8. IEEE 802.1w does not use the exact same port states as does 802.1D. Which of the following are valid 802.1w port states?

   a. Blocking
   b. Listening
   c. Learning
   d. Forwarding
   e. Disabled
   f. Discarding

9. What STP tools or protocols supply a "maxage optimization," allowing a switch to bypass the wait for maxage to expire when its Root Port stops receiving Hellos?

   a. Loop Guard
   b. UDLD
   c. UplinkFast
   d. BackboneFast
   e. IEEE 802.1w

10. A trunk between switches lost its physical transmit path in one direction only. Which of the following features protect against the STP problems caused by such an event?

a. Loop Guard

b. UDLD

c. UplinkFast

d. PortFast

e. UplinkFast

Foundation Topics

802.1D Spanning Tree Protocol

Although many CCIE candidates already know STP well, the details are easily forgotten. For instance, you can install a campus LAN, possibly turn on a few STP optimizations and security features out of habit, and have a working LAN using STP—without ever really contemplating how STP does what it does. And in a network that makes good use of Layer 3 switching, each STP instance might span only three to four switches, making the STP issues much more manageable—but more forgettable in terms of helping you remember things you need to know for the exam. This chapter reviews the details of IEEE 802.1D STP, and then goes on to related topics—802.1w RSTP, multiple spanning trees, STP optimizations, and STP security features.

STP uses messaging between switches to stabilize the network into a logical, loop-free topology. To do so, STP causes some interfaces (popularly called *ports* when discussing STP) to simply not forward or receive traffic—in other words, the ports are in a *blocking* state. The remaining ports, in an STP *forwarding* state, together provide a loop-free path to every Ethernet segment in the network.

Choosing Which Ports Forward: Choosing Root Ports and Designated Ports

To determine which ports forward and block, STP follows a three-step process, as listed in Table 3-2. Following the table, each of the three steps is explained in more detail.

Table 3-2 *Three Major 802.1D STP Process Steps*

KEY POINT

Major Step	Description
Elect the Root switch	The switch with the lowest bridge ID wins; the standard bridge ID is 2-byte priority followed by a MAC address unique to that switch.
Determine each switch's Root Port	The one port on each switch with the least cost path back to the root.
Determine the Designated Port for each segment	When multiple switches connect to the same segment, this is the switch that forwards the least cost Hello onto a segment.

Electing a Root Switch

Only one switch can be the *root* of the spanning tree; to select the root, the switches hold an *election*. Each switch begins its STP logic by creating and sending an STP Hello bridge protocol

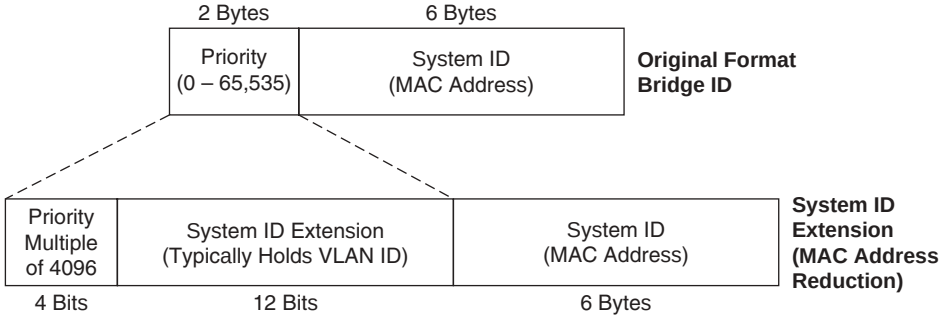
data unit (BPDU) message, claiming to be the root switch. If a switch hears a *superior Hello*—a Hello with a lower bridge ID—it stops claiming to be root by ceasing to originate and send Hellos. Instead, the switch starts forwarding the superior Hellos received from the superior candidate. Eventually, all switches except the switch with the best bridge ID cease to originate Hellos; that one switch wins the election and becomes the root switch.

The original IEEE 802.1D bridge ID held two fields:

- The 2-byte Priority field, which was designed to be configured on the various switches to affect the results of the STP election process.
- A 6-byte MAC Address field, which was included as a tiebreaker, because each switch's bridge ID includes a MAC address value that should be unique to each switch. As a result, some switch must win the root election.

The format of the original 802.1D bridge ID has been redefined. Figure 3-1 shows the original and new format of the bridge IDs.

Figure 3-1 *IEEE 802.1D STP Bridge ID Formats*



The format was changed mainly due to the advent of multiple spanning trees as supported by Per VLAN Spanning Tree Plus (PVST+) and IEEE 802.1s Multiple Spanning Trees (MST). With the old-style bridge ID format, a switch's bridge ID for each STP instance (possibly one per VLAN) was identical if the switch used a single MAC address when building the bridge ID. Having multiple STP instances with the same bridge ID was confusing, so vendors such as Cisco Systems used a different Ethernet BIA for each VLAN when creating the old-style bridge IDs. This provided a different bridge ID per VLAN, but it consumed a large number of reserved BIAs in each switch.

The System ID Extension allows a network to use multiple instances of STP, even one per VLAN, but without the need to consume a separate BIA on each switch for each STP instance. The System ID Extension field allows the VLAN ID to be placed into what was formerly the last 12 bits of the

Priority field. A switch can use a single MAC address to build bridge IDs, and with the VLAN number in the System ID Extension field still have a unique bridge ID in each VLAN. The use of the System ID Extension field is also called *MAC address reduction*, because of the need for many fewer reserved MAC addresses on each switch.

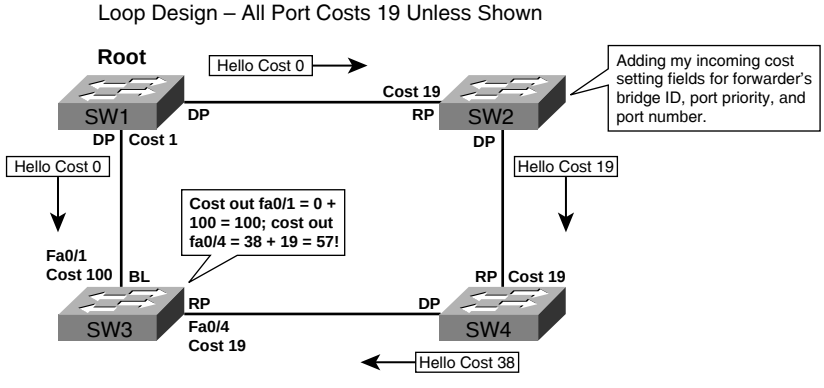
Determining the Root Port

Once the root is elected, the rest of the switches now need to determine their *Root Port (RP)*. The process proceeds as described in the following list:

1. The root creates and sends a Hello every Hello timer (2 seconds default).
2. Each switch that receives a Hello forwards the Hello after updating the following fields in the Hello: the cost, the forwarding switch's bridge ID, forwarder's port priority, and forwarder's port number.
3. Switches do not forward Hellos out ports that stabilize into a blocking state.
4. Of all the ports in which a switch receives Hellos, the port with the least calculated cost to the root is the RP.

A switch must examine the cost value in each Hello, plus the switch's STP port costs, in order to determine its least cost path to reach the root. To do so, the switch adds the cost listed in the Hello message to the switch's port cost of the port on which the Hello was received. For example, Figure 3-2 shows the loop network design and details several STP cost circulations.

Figure 3-2 *Calculating STP Costs to Determine RPs*



In Figure 3-2, SW1 happened to become root, and is originating Hellos of cost 0. SW3 receives two Hellos, one with cost 0 and one with cost 38. However, SW3 must then calculate its cost to reach the root, which is the advertised cost (0 and 38, respectively) plus SW3's port costs (100 and 19, respectively). As a result, although SW3 has a direct link to SW1, the calculated cost is lower

out interface fa0/4 (cost 57) than it is out interface fa0/1 (cost 100), so SW3 chooses its fa0/4 interface as its RP.

NOTE Many people think of STP costs as being associated with a segment; however, the cost is actually associated with interfaces. Good design practices dictate using the same STP cost on each end of a point-to-point Ethernet segment, but the values can be different.

While the costs shown in Figure 3-2 might seem a bit contrived, the same result would happen with default port costs if the link from SW1 to SW3 were Fast Ethernet (default cost 19), and the other links were Gigabit Ethernet (default cost 4). Table 3-3 lists the default port costs according to IEEE 802.1D. Note that the IEEE updated 802.1D in the late 1990s, changing the suggested default port costs.

Table 3-3 *Default Port Costs According to IEEE 802.1D*

Speed of Ethernet	Original IEEE Cost	Revised IEEE Cost
10 Mbps	100	100
100 Mbps	10	19
1 Gbps	1	4
10 Gbps	1	2

When a switch receives multiple Hellos with equal calculated cost, it uses the following tiebreakers:

1. Pick the lowest value of the forwarding switch's bridge ID.
2. Use the lowest port priority of the neighboring switch. The neighboring switch added its own port priority to the Hello before forwarding it.
3. Use the lowest internal port number (of the forwarding switch) as listed inside the received Hellos.

Note that if the first tiebreaker in this list fails to produce an RP, this switch must have multiple links to the same neighboring switch. The last two tiebreakers simply help decide which of the multiple parallel links to use.

Determining the Designated Port

A converged STP topology results in only one switch forwarding onto each LAN segment. The switch that forwards onto a LAN segment is called the *designated switch* for that segment, and

the port that it uses to forward frames onto that segment is called the *Designated Port (DP)*. By definition, only the DP on that segment should forward frames onto the segment.

To win the right to be the DP, a switch must send the Hello with the *lowest advertised cost* onto the segment. For instance, consider the segment between SW3 and SW4 in Figure 3-2 before the DP has been determined on that segment. SW3 would get Hellos directly from SW1, compute its cost to the root over that path, and then forward the Hello out its fa 0/4 interface to SW4, with cost 100. Similarly, SW4 will forward a Hello with cost 38, as shown in Figure 3-2. SW4's fa 0/3 port becomes the DP due to its lower advertised cost.

Only the DP forwards Hellos onto a LAN segment as well. In the same example, SW4 keeps sending the cost-38 Hellos out the port, but SW3 stops sending its inferior Hellos.

When the cost is a tie, STP uses the same tiebreakers to choose the DP as when choosing an RP: lowest forwarder's bridge ID, lowest forwarder's port priority, and lowest forwarder's port number.

Converging to a New STP Topology

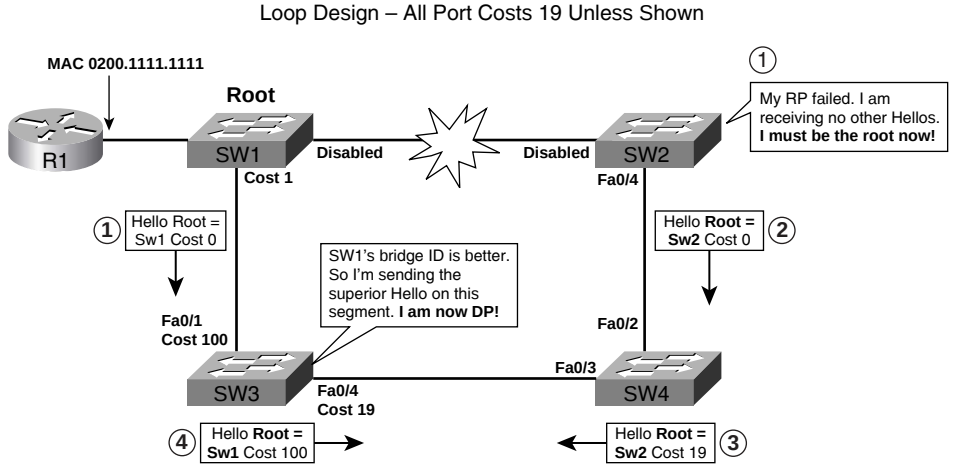
STP logic monitors the normal ongoing Hello process when the network topology is stable; when the Hello process changes, STP then needs to react and converge to a new STP topology. When STP has a stable topology, the following occurs:

1. The root switch generates a Hello regularly based on the Hello timer.
2. Each non-root switch regularly (based on the Hello timer) receives a copy of the root's Hello on its RP.
3. Each switch updates and forwards the Hello out its Designated Ports.
4. For each blocking port, the switch regularly receives a copy of the Hello from the DP on that segment. (The switches do not forward Hellos out blocking interfaces.)

When some deviation from these events occurs, STP knows that the topology has changed and that convergence needs to take place. For instance, one simple case might be that the root switch loses power; the rest of the switches will not hear any Hello messages, and after the maxage timer expires (default 10 times Hello, or 20 seconds), the switches elect a new root based on the logic described earlier in this chapter.

For a more subtle example, consider Figure 3-3, which shows the same loop network as in Figure 3-2. In this case, however, the link from SW1 to SW2 has just failed.

Figure 3-3 *Reacting to Loss of Link Between SW1 and SW2*



The following list describes some of the key steps from Figure 3-3:

1. SW2 ceases the receive Hellos on its RP.
2. Because SW2 is not receiving Hellos over any other path, it begins a new root election by claiming to be root and flooding Hellos out every port.
3. SW4 notices that the latest Hello implies a new root switch, but SW4 ends up with the same RP (for now). SW4 forwards the Hello out toward SW3 after updating the appropriate fields in the Hello.
4. SW3 receives the Hello from SW4, but it is inferior to the one SW3 receives from SW1. So, SW3 becomes the DP on the segment between itself and SW4, and starts forwarding the superior Hello on that port.

Remember, SW1 had won the earlier election; as of Steps 3 and 4, the Hellos from SW1 and SW2 are competing, and the one claiming SW1 as root will again win. The rest of the process results with SW3's fa0/4 as DP, SW4's fa 0/3 as RP, SW4's fa 0/2 as DP, and SW3's fa 0/4 as RP.

Topology Change Notification and Updating the CAM

When STP reconvergence occurs, some Content Addressable Memory (CAM) entries might be invalid (CAM is the Cisco term for what's more generically called the MAC address table, switching table, or bridging table on a switch). For instance, before the link failure shown in Figure 3-3, SW3's CAM might have had an entry for 0200.1111.1111 (Router1's MAC address) pointing out fa0/4 to SW4. (Remember, at the beginning of the scenario described in Figure 3-3,

SW3 was blocking on its fa0/1 interface back to SW1.) When the link between SW1 and SW2 failed, SW3 would need to change its CAM entry for 0200.1111.111 to point out port fa0/1.

To update the CAMs, two things need to occur:

- All switches need to be notified to time out their CAM entries.
- Each switch needs to use a short timer, equivalent to the forward delay timer (default 15 seconds), to time out the CAM entries.

Because some switches might not directly notice a change in the STP topology, any switch that detects a change in the STP topology has a responsibility to notify the rest of the switches. To do so, a switch simply notifies the root switch in the form of a *Topology Change Notification (TCN)* BPDU. The TCN goes up the tree to the root. After that, the root notifies all the rest of the switches. The process runs as follows:

1. A switch experiencing the STP port state change sends a TCN BPDU out its Root Port; it repeats this message every Hello time until it is acknowledged.
2. The next switch receiving that TCN BPDU sends back an acknowledgement via its next forwarded Hello BPDU by marking the *Topology Change Acknowledgement (TCA)* bit in the Hello.
3. The switch that was the DP on the segment in the first two steps repeats the first two steps, sending a TCN BPDU out its Root Port, and awaiting acknowledgement from the DP on that segment.

By each successive switch repeating Steps 1 and 2, eventually the root receives a TCN BPDU. Once received, the root sets the TCA flag on the next several Hellos, which are forwarded to all switches in the network, notifying them that a change has occurred. A switch receiving a Hello BPDU with the TCA flag set uses the short (forward delay time) timer to time out entries in the CAM.

Transitioning from Blocking to Forwarding

When STP reconverges to a new, stable topology, some ports that were blocking might have been designated as DP or RP, so these ports need to be in a forwarding state. However, the transition from blocking to forwarding state cannot be made immediately without the risk of causing loops.

To transition to forwarding state but also prevent temporary loops, a switch first puts a formerly blocking port into *listening* state, and then into *learning* state, with each state lasting for the length of time defined by the forward delay timer (by default, 15 seconds). Table 3-4 summarizes the key points about all of the 802.1D STP port states.

Table 3-4 *IEEE 802.1D Spanning Tree Interface States*

State	Forwards Data Frames?	Learn Source MACs of Received Frames?	Transitory or Stable State?
Blocking	No	No	Stable
Listening	No	No	Transitory
Learning	No	Yes	Transitory
Forwarding	Yes	Yes	Stable
Disabled	No	No	Stable

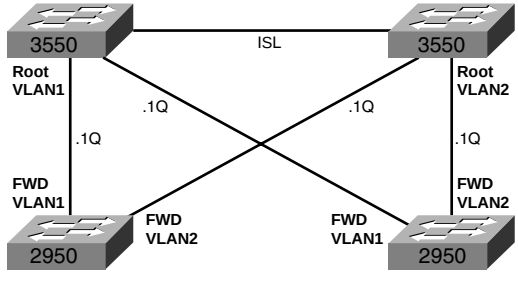
In summary, when STP logic senses a change in the topology, it converges, possibly picking different ports as RP, DP, or neither. Any switch changing its RPs or DPs sends a TCN BPDU to the root at this point. For the ports newly designated as RP or DP, 802.1D STP first uses the listening and learning states before reaching the forwarding state. (The transition from forwarding to blocking can be made immediately.)

Per-VLAN Spanning Tree and STP over Trunks

If only one instance of STP was used for a switched network with redundant links but with multiple VLANs, several ports would be in a blocking state, unused under stable conditions. The redundant links would essentially be used for backup purposes.

The Cisco Per VLAN Spanning Tree Plus (PVST+) feature creates an STP instance for each VLAN. By tuning STP configuration per VLAN, each STP instance can use a different root switch and have different interfaces block. As a result, the traffic load can be balanced across the available links. For instance, in the common building design with distribution and access links in Figure 3-4, focus on the left side of the figure. In this case, the access layer switches block on different ports on VLANs 1 and 2, with different root switches.

Figure 3-4 *Operation of PVST+ for Better Load Balancing*



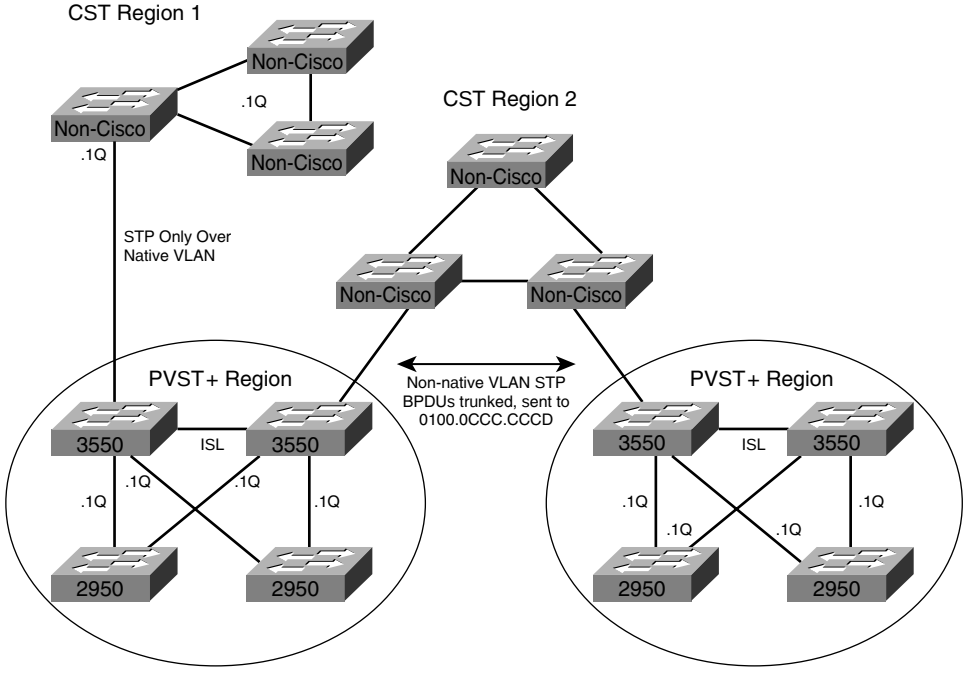
With different root switches and with default port costs, the access layer switches end up sending VLAN1 traffic over one uplink and VLAN2 traffic over another uplink.

Using 802.1Q with STP requires some extra thought as to how it works. 802.1Q does not support PVST+ natively; however, Cisco switches do support PVST+ over 802.1Q trunks. So, with all Cisco switches, and PVST+ (which is enabled by default), PVST+ works fine.

When using 802.1Q with non-Cisco switches, the switches must follow the IEEE standard completely, so the trunks support only a *Common Spanning Tree (CST)*. With standard 802.1Q, only one instance of STP runs only over the native VLAN, and that one STP topology is used for all VLANs. Although using only one STP instance reduces the STP messaging overhead, it does not allow load balancing by using multiple STP instances, as was shown with PVST+ in Figure 3-4.

When building networks using a mix of Cisco and non-Cisco switches, along with 802.1Q trunking, you can still take advantage of multiple STP instances in the Cisco portion of the network. Figure 3-5 shows two general options in which two CST regions of non-Cisco switches connect to two regions of Cisco PVST+ supporting switches.

Figure 3-5 *Combining Standard IEEE 802.1Q and CST with PVST+*



The left side of Figure 3-5 shows an example in which the CST region is not used for transit between multiple PVST+ regions. In this case, none of the PVST+ per-VLAN STP information needs to pass over the CST region. The PVST+ region maps the single CST instance to each of the PVST+ STP instances.

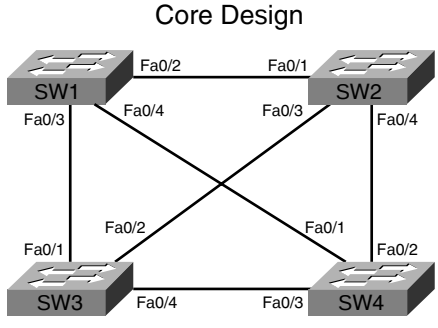
The rest of Figure 3-5 shows two PVST+ regions, separated by a single CST region (CST Region 2). In this case, the PVST+ per-VLAN STP information needs to pass through the CST region. To do so, PVST+ treats the CST region as a single link and tunnels the PVST+ BPDUs across the CST region. The tunnel is created by sending the BPDUs using a multicast destination MAC of 0100.0CCC.CCCD, with the BPDUs being VLAN tagged with the correct VLAN ID. As a result, the non-Cisco switches forward the BPDUs as a multicast, and do not interpret the frames as BPDUs. When a forwarded BPDU reaches the first Cisco PVST+ switch in the other PVST+ region, the switch, listening for multicasts to 0100.0CCC.CCCD, reads and interprets the BPDU.

NOTE 802.1Q, along with 802.1S Multiple-instance Spanning Tree (MST), allows 802.1Q trunks for support multiple STP instances. MST is covered later in this chapter.

STP Configuration and Analysis

Example 3-1, based on Figure 3-6, shows some of the basic STP configuration and **show** commands. Take care to note that many of the upcoming commands allow the parameters to be set for all VLANs by omitting the VLAN parameter, or set per VLAN by including a VLAN parameter. Example 3-1 begins with SW1 coincidentally becoming the Root switch. After that, SW2 is configured to become root, and SW3 changes its Root Port as a result of a configured port cost in VLAN 1.

Figure 3-6 *Network Used with Example 3-1*



Example 3-1 *STP Basic Configuration and* **show** *Commands*

```
! First, note the Root ID column lists the root's bridge ID as two parts,
! first the priority, followed by the MAC address of the root. The root cost of
! 0 implies that SW1 (where the command is executed) is the root.
SW1#sh spanning-tree root
                                        Root    Hello Max Fwd
Vlan                Root ID             Cost    Time  Age Dly  Root Port
---------------- ------------------- --------- ----- --- ---  ------------
VLAN0001         32769 000a.b7dc.b780         0    2   20  15
VLAN0011         32779 000a.b7dc.b780         0    2   20  15
VLAN0012         32780 000a.b7dc.b780         0    2   20  15
VLAN0021         32789 000a.b7dc.b780         0    2   20  15
VLAN0022         32790 000a.b7dc.b780         0    2   20  15
```

Example 3-1 *STP Basic Configuration and* **show** *Commands (Continued)*

```
! The next command confirms that SW1 believes that it is the root of VLAN 1.
SW1#sh spanning-tree vlan 1 root detail
  Root ID     Priority    32769
              Address     000a.b7dc.b780
              This bridge is the root
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
! Next, SW2 is configured with a lower (better) priority than SW1,
! so it becomes the root. Note that because SW2 is defaulting to use
! the system ID extension, the actual priority must be configured as a
! multiple of 4096.
SW2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
SW2(config)#spanning-tree vlan 1 priority ?
  <0-61440>  bridge priority in increments of 4096

SW2(config)#spanning-tree vlan 1 priority 28672
SW2(config)#^Z
SW2#sh spanning-tree vlan 1 root detail
VLAN0001
  Root ID     Priority    28673
              Address     0011.92b0.f500
              This bridge is the root
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
! The System ID Extension field of the bridge ID is implied next. The output
! does not separate the 4-bit Priority field from the System ID field. The output
! actually shows the first 2 bytes of the bridge ID, in decimal. For VLAN1,
! the priority is 28,763, which is the configured 28,672 plus the VLAN ID,
! because the VLAN ID value is used in the System ID field in order to implement
! the MAC address reduction feature. The other VLANs have a base priority
! of 32768, plus the VLAN ID - for example, VLAN11 has priority 32779,
! (priority 32,768 plus VLAN 11), VLAN12 has 32780, and so on.
SW2#sh spanning-tree root priority
VLAN0001         28673
VLAN0011         32779
VLAN0012         32780
VLAN0021         32789
VLAN0022         32790
! Below, SW3 shows a Root Port of Fa 0/2, with cost 19. SW3 gets Hellos
! directly from the root (SW2) with cost 0, and adds its default cost (19).
! This next command also details the breakdown of the priority and system ID.
SW3#sh spanning-tree vlan 1
VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    28673
              Address     0011.92b0.f500
              Cost        19
```

continues

Example 3-1 *STP Basic Configuration and* **show** *Commands (Continued)*

```
             Port        2 (FastEthernet0/2)
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32769  (priority 32768 sys-id-ext 1)
             Address     000e.837b.3100
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1            Altn BLK 19        128.1    P2p
Fa0/2            Root FWD 19        128.2    P2p
Fa0/4            Desg FWD 19        128.4    P2p
Fa0/13           Desg FWD 100       128.13   Shr
! Above, the port state of BLK and FWD for each port is shown, as well as the
! Root port and the Designated Ports.
! Below, Switch3's VLAN 1 port cost is changed on its Root Port (fa0/2),
! causing SW3 to reconverge, and pick a new RP.
SW3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
SW3(config)#int fa 0/2
SW3(config-if)#spanning-tree vlan 1 cost 100
SW3(config-if)#^Z
! The next command was done immediately after changing the port cost on
! SW3. Note the state listed as "LIS," meaning listen. STP has already
! chosen fa 0/1 as the new RP, but it must now transition through listening
! and learning states.
SW3#sh spanning-tree vlan 1
VLAN0001
  Spanning tree enabled protocol ieee
  Root ID    Priority    28673
             Address     0011.92b0.f500
             Cost        38
             Port        1 (FastEthernet0/1)
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32769  (priority 32768 sys-id-ext 1)
             Address     000e.837b.3100
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time 15
Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1            Root LIS 19        128.1    P2p
Fa0/2            Altn BLK 100       128.2    P2p
Fa0/4            Desg FWD 19        128.4    P2p
Fa0/13           Desg FWD 100       128.13   Shr
```

The preceding example shows one way to configure the priority to a lower value to become the root. Optionally, the **spanning-tree vlan** *vlan-id* **root** {**primary** | **secondary**} [**diameter** *diameter*] command could be used. This command causes the switch to set the priority lower. The optional **diameter** parameter causes this command to lower the Hello, forward delay, and maxage timers. (This command does not get placed into the configuration, but rather it acts as a macro, being expanded into the commands to set priority and the timers.)

NOTE When using the **primary** option, the **spanning-tree vlan** command sets the priority to 24,576 if the current root has a priority larger than 24,576. If the current root's priority is 24,576 or less, this command sets this switch's priority to 4096 less than the current root. With the **secondary** keyword, this switch's priority is set to 28,672. Also note that this logic applies to when the configuration command is executed; it does not dynamically change the priority if another switch later advertises a better priority.

Optimizing Spanning Tree

Left to default settings, IEEE 802.1D STP works, but convergence might take up to a minute or more for the entire network. For instance, when the root fails, a switch must wait on the 20-second maxage timer to expire. Then, newly forwarding ports spend 15 seconds each in listening and learning states, which makes convergence take 50 seconds for that one switch.

Over the years, Cisco added features to its STP code, and later the IEEE made improvements as well. This section covers the key optimizations to STP.

PortFast, UplinkFast, and BackboneFast

The Cisco-proprietary PortFast, UplinkFast, and BackboneFast features each solve specific STP problems. Table 3-5 summarizes when each is most useful, and the short version of how they improve convergence time.

Table 3-5 *PortFast, UplinkFast, and BackboneFast*

KEY POINT

Feature	Requirements for Use	How Convergence Is Optimized
PortFast	Used on access ports that are not connected to other switches or hubs	Immediately puts the port into forwarding state once the port is physically working
UplinkFast	Used on access layer switches that have multiple uplinks to distribution/core switches	Immediately replaces a lost RP with an alternate RP, immediately forwards on the RP, and triggers updates of all switches' CAMs
BackboneFast	Used to detect indirect link failures, typically in the network core	Avoids waiting for Maxage to expire when its RP ceases to receive Hellos; does so by querying the switch attached to its RP

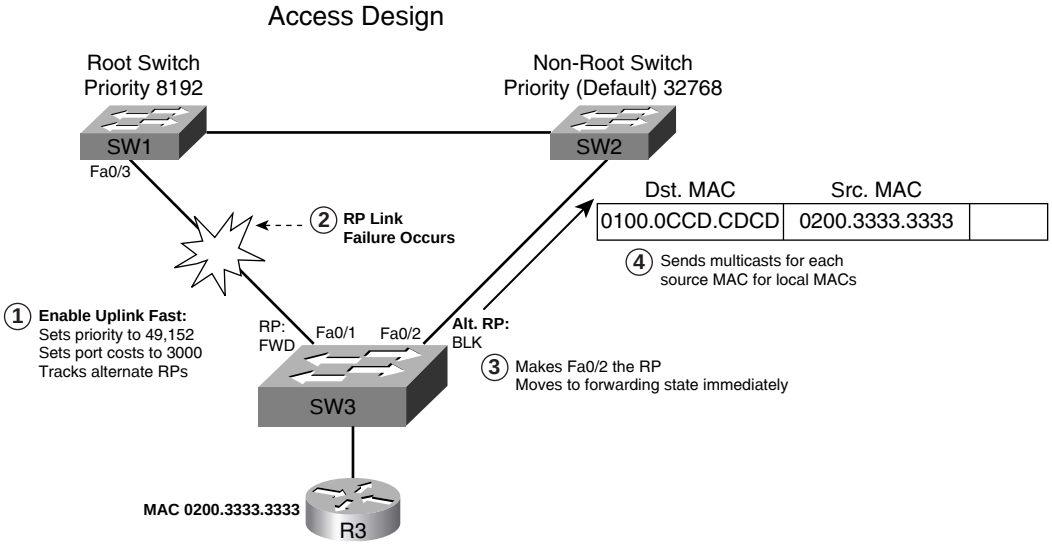
PortFast

PortFast optimizes convergence by simply ignoring listening and learning states on ports. In effect, convergence happens instantly on ports with PortFast enabled. Of course, if another switch is connected to a port on which PortFast is enabled, loops may occur. So, PortFast is intended for access links attached to single end-user devices. To be safe, you should also enable the BPDU Guard and Root Guard features when using PortFast, as covered later in this chapter.

UplinkFast

UplinkFast optimizes convergence when an uplink fails on an access layer switch. For good STP design, access layer switches should not become root or become transit switches. (A transit switch is a switch that forwards frames between other switches.) Figure 3-7 shows the actions taken when UplinkFast is enabled on a switch, and then when the Root Port fails.

Figure 3-7 *UplinkFast Operations*



Upon enabling UplinkFast globally in a switch, the switch takes three actions:

- Increases the root priority to 49,152
- Sets the post costs to 3000
- Tracks alternate RPs, which are ports in which root Hellos are being received

As a result of these steps, SW3 can become root if necessary, but it is unlikely to do so given the large root priority value. Also, the very large costs on each link make the switch unlikely to be used as a transit switch. When the RP port does fail, SW3 can fail over to an alternate uplink as the new RP and forward immediately.

The final step in Uplink Fast logic causes the switches to time-out the correct entries in their CAMs, but it does not use the TCN process. Instead, the access switch finds all the MAC addresses of local devices and sends one multicast frame with each local MAC address as the source MAC—causing all the other switches to update their CAMs. The access switch also clears out the rest of the entries in its own CAM.

BackboneFast

BackboneFast optimizes convergence for any generalized topological case, improving convergence when an *indirect failure* occurs. When some direct failures occur (for instance, a switch's RP interface fails), the switch does not have to wait for maxage to expire. However, when another switch's direct link fails, resulting in lost Hellos for other switches, the downstream switches indirectly learn of the failure because they cease to receive Hellos. Any time a switch learns of an STP failure indirectly, the switch must wait for maxage to expire before trying to change the STP topology.

BackboneFast simply causes switches that indirectly learn of a potential STP failure to ask their upstream neighbors if they know about the failure. To do so, when the first Hello goes missing, a BackboneFast switch sends a *Root Link Query (RLQ)* BPDU out the port in which the missing Hello should have arrived. The RLQ asks the neighboring switch if that neighboring switch is still receiving Hellos from the root. If that neighboring switch had a direct link failure, it can tell the original switch (via another RLQ) that this path to the root is lost. Once known, the switch experiencing the indirect link failure can go ahead and converge without waiting for maxage to expire.

NOTE All switches must have BackboneFast configured for it to work correctly.

PortFast, UplinkFast, and BackboneFast Configuration

Configuration of these three STP optimizing tools is relatively easy, as summarized in Table 3-6.

Table 3-6 *PortFast, UplinkFast, and BackboneFast Configuration*

Feature	Configuration Command
PortFast	**spanning-tree portfast** (interface subcommand) **spanning-tree portfast default** (global)
UplinkFast	**spanning-tree uplinkfast** [**max-update-rate** *rate*] (global)
BackboneFast	**spanning-tree backbonefast** (global)

PortChannels

When a network design includes multiple parallel segments between the same pair of switches, one switch ends up in a forwarding state on all the links, but the other switch blocks all but one of the ports of those parallel segments. As a result, only one of the links can be used at any point in time. Using *Fast EtherChannel (FEC)* (using FastE segments) and *Gigabit EtherChannel (GEC)* (using GigE segments) allows the combined links to be treated as one link from an STP perspective, so that all the parallel physical segments are used. (When configuring a Cisco switch, a group of segments comprising an FEC or GEC is called a *PortChannel*.) Most campus designs today use a minimum of two segments per trunk, in a PortChannel, for better availability. That way, as long as at least one of the links in the EtherChannel is up, the STP path cannot fail, and no STP convergence is required.

Load Balancing Across PortChannels

When a switch decides to forward a frame out a PortChannel, the switch must also decide which physical link to use to send each frame. To use the multiple links, Cisco switches load balance the traffic over the links in an EtherChannel based on the switch's global load-balancing configuration.

Load-balancing methods differ depending on the model of switch and software revision. Generally, load balancing is based on the contents of the Layer 2, 3, and/or 4 headers. If load balancing is based on only one header field in the frame, a bitmap of the low-order bits is used; if more than one header field is used, an XOR of the low-order bits is used.

For the best balancing effect, the header fields on which balancing is based need to vary among the mix of frames sent over the PortChannel. For instance, for a Layer 2 PortChannel connected to an access layer switch, most of the traffic going from the access layer switch to the distribution layer switch is probably going from clients to the default router. So most of the frames have different source MAC addresses, but the same destination MAC address. For packets coming back from a distribution switch toward the access layer switch, many of the frames might have a source address of that same router, with differing destination MAC addresses. So, you could balance based on source MAC at the access layer switch, and based on destination MAC at the distribution layer switch—or balance based on both fields on both switches. The goal is simply to use a balancing method for which the fields in the frames vary.

The **port-channel load-balance** *type* command sets the type of load balancing. The *type* options include using source and destination MAC, IP addresses, and TCP and UDP ports—either a single field or both the source and destination.

PortChannel Discovery and Configuration

You can explicitly configure interfaces to be in a PortChannel by using the **channel-group** *number* **mode on** interface subcommand. You would simply put the same command under each of the physical interfaces inside the PortChannel, using the same PortChannel number.

You can also use dynamic protocols to allow neighboring switches to figure out which ports should be part of the same PortChannel. Those protocols are the Cisco-proprietary *Port Aggregation Protocol (PAgP)* and the IEEE 802.1AD *Link Aggregation Control Protocol (LACP)*. To dynamically form a PortChannel using PAgP, you still use the **channel-group** command, with a mode of **auto** or **desirable**. To use LACP to dynamically create a PortChannel, use a mode of **active** or **passive**. Table 3-7 lists and describes the modes and their meanings.

Table 3-7 *PAgP and LACP Configuration Settings and Recommendations*

PAgP Setting	LACP 802.1AD Setting	Action
on	**on**	Disables PAgP or LACP, and forces the port into the PortChannel
off	**off**	Disables PAgP or LACP, and prevents the port from being part of a PortChannel
auto	**passive**	Uses PAgP or LACP, but waits on other side to send first PAgP or LACP message
desirable	**active**	Uses PAgP or LACP, and initiates the negotiation

NOTE Using **auto** (PAgP) or **passive** (LACP) on both switches prevents a PortChannel from forming dynamically. Cisco recommends the use of desirable mode (PAgP) or active mode (LACP) on ports that you intend to be part of a PortChannel.

When PAgP or LACP negotiate to form a PortChannel, the messages include the exchange of some key configuration information. As you might imagine, they exchange a system ID to determine which ports connect to the same two switches. The two switches then exchange other information about the candidate links for a PortChannel; several items must be identical on the links for them to be dynamically added to the PortChannel, as follows:

- Same speed and duplex settings.
- If not trunking, same access VLAN.
- If trunking, same trunk type, allowed VLANs, and native VLAN.
- On a single switch, each port in a PortChannel must have the same STP cost per VLAN on all links in the PortChannel.
- No ports can have SPAN configured.

When PAgP or LACP completes the process, a new PortChannel interface exists, and is used as if it were a single port for STP purposes, with balancing taking place based on the global load-balancing method configured on each switch.

Rapid Spanning Tree Protocol

IEEE 802.1w *Rapid Spanning Tree Protocol (RSTP)* enhances the 802.1D standard with one goal in mind: improving STP convergence. To do so, RSTP defines new variations on BPDUs between switches, new port states, and new port roles, all with the capability to operate backwardly compatible with 802.1D switches. The key components of speeding convergence with 802.1w are as follows:

- Waiting for only three missed Hellos on an RP before reacting (versus ten missed Hellos via the maxage timer with 802.1D)
- New processes that allow transition from the disabled state (replaces the blocking state in 802.1D) to learning state, bypassing the concept of an 802.1D listening state
- Standardization of features like Cisco PortFast, UplinkFast, and BackboneFast
- An additional feature to allow a backup DP when a switch has multiple ports connected to the same shared LAN segment

To support these new processes, RSTP uses the same familiar Hello BPDUs, using some previously undefined bits to create the new features. For instance, RSTP defines a Hello message option for the same purpose as the Cisco proprietary RLQ used by the Cisco BackboneFast feature.

RSTP takes advantage of a switched network topology by categorizing ports, using a different link type to describe each. RSTP takes advantage of the fact that STP logic can be simplified in some cases, based on what is attached to each port, thereby allowing faster convergence. Table 3-8 lists the three RSTP link types.

Table 3-8 *RSTP Link Types*

Link Type	Description
Point-to-point	Connects a switch to one other switch; Cisco switches treat FDX links in which Hellos are received as point-to-point links.
Shared	Connects a switch to a hub; the important factor is that switches are reachable off that port.
Edge	Connects a switch to a single end-user device.

In most modern LAN designs with no shared hubs, all links would be either the point-to-point (a link between two switches) or edge link type. RSTP knows that link-type edge means the port is cabled to one device, and the device is not a switch. So, RSTP treats edge links with the same logic as Cisco PortFast—in fact, the same **spanning-tree portfast** command defines a port as link-type edge to RSTP. In other words, RSTP puts edge links into forwarding state immediately.

RSTP takes advantage of point-to-point links (which by definition connect a switch to another switch) by asking the other switch about its status. For instance, if one switch fails to receive its periodic Hello on a point-to-point link, it will query the neighbor. The neighbor will reply, stating whether it also lost its path to the root. It is the same logic as BackboneFast, but using IEEE standard messages to achieve the same goal.

RSTP also redefines the port states used with 802.1D, in part because the listening state is no longer needed. Table 3-9 compares the port states defined by each protocol.

Table 3-9 *RSTP and STP Port States*

Administrative State	STP State (802.1D)	RSTP State (802.1w)
Disabled	Disabled	Discarding
Enabled	Blocking	Discarding
Enabled	Listening	Discarding
Enabled	Learning	Learning
Enabled	Forwarding	Forwarding

In RSTP, a discarding state means that the port does not forward frames, receive frames, or learn source MAC addresses, regardless of whether the port was shut down, failed, or simply does not have a reason to forward. Once RSTP decides to transition from discarding to forwarding state (for example, a newly selected RP), it goes immediately to the learning state. From that point on, the process continues just as it does with 802.1D. RSTP no longer needs the listening state because of its active querying to neighbors, which guarantees no loops during convergence.

RSTP uses the term *port role* to refer to whether a port acts as an RP or a DP. RSTP uses the RP and DP port roles just as 802.1D does; however, RSTP adds several other roles, as listed in Table 3-10.

Table 3-10 *RSTP and STP Port Roles*

RSTP Role	Definition
Root Port	Same as 802.1D Root Port.
Designated Port	Same as 802.1D Designated Port.
Alternate Port	Same as the Alternate Port concept in UplinkFast; an alternate Root Port.
Backup Port	A port that is attached to the same link-type shared link as another port on the same switch, but the other port is the DP for that segment. The Backup Port is ready to take over if the DP fails.

The Alternate Port concept is like the UplinkFast concept—it offers protection against the loss of a switch's RP by keeping track of the Alternate Ports with a path to the root. The concept and general operation is identical to UplinkFast, although RSTP might converge more quickly via its active messaging between switches.

The Backup Port role has no equivalent with Cisco-proprietary features; it simply provides protection against losing the DP attached to a shared link when the switch has another physical port attached to the same shared LAN.

You can enable RSTP in a Cisco switch by using the **spanning-tree mode rapid-pvst** global command. Alternatively, you can simply enable 802.1s MST, which by definition uses 802.1w RSTP.

Multiple Spanning Trees: IEEE 802.1s

IEEE 802.1s *Multiple Spanning Trees (MST)*, sometimes referred to as *Multiple Instance STP (MISTP)* or *Multiple STP (MSTP)*, defines a way to use multiple instances of STP in a network that uses 802.1Q trunking. The following are some of the main benefits of 802.1s:

- Like PVST+, it allows the tuning of STP parameters so that while some ports block for one VLAN, the same port can forward in another VLAN.
- Always uses 802.1w RSTP, for faster convergence.
- Does not require an STP instance for each VLAN; rather, the best designs use one STP instance per redundant path.

If the network consists of all MST-capable switches, MST is relatively simple to understand. A group of switches that together uses MST is called an *MST region*; to create an MST region, the switches need to be configured as follows:

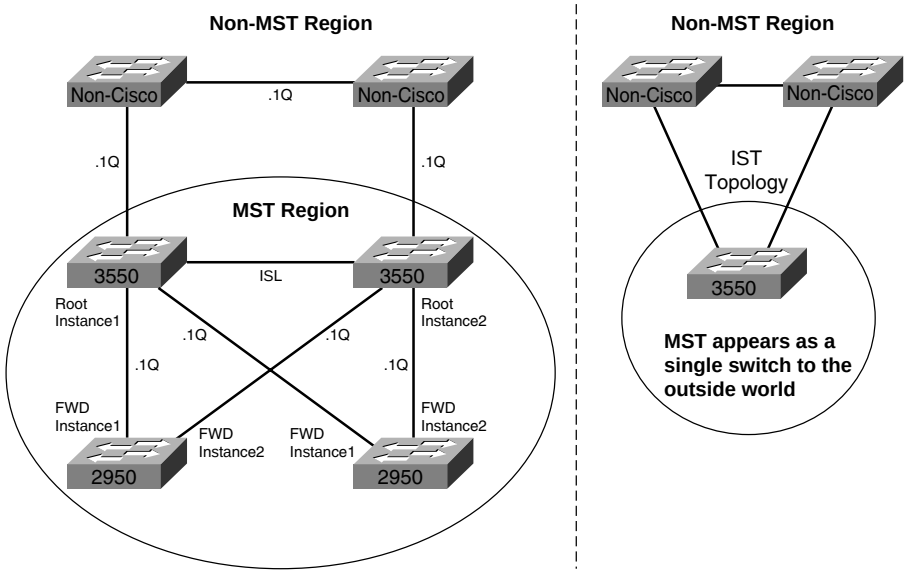
1. Globally enable MST, and enter MST configuration mode by using the **spanning-tree mode mst** command.

2. From MST configuration mode, create an MST region name (up to 32 characters) by using the **name** subcommand.
3. From MST configuration mode, create an MST revision number by using the **revision** command.
4. From MST configuration mode, map VLANs to an MST STP instance by using the **instance** command.

The key to MST configuration is to configure the same parameters on all the switches in the region. For instance, if you match VLANs 1–4 to MST instance 1 on one switch, and VLANs 5–8 to MST instance 1 on another switch, the two switches will not consider themselves to be in the same MST region, even though their region names and revision numbers are identical.

For example, in Figure 3-8, an MST region has been defined, along with connections to non-MST switches. Focusing on the left side of the figure, inside the MST region, you really need only two instances of STP—one each for roughly half of the VLANs. With two instances, the access layer switches will forward on their links to SW1 for one set of VLANs using one MST instance, and forward on their links to SW2 for the other set of VLANs using the second MST instance.

Figure 3-8 *MST Operations*



One of the key benefits of MST versus PVST+ is that it requires only one MST instance for a group of VLANs. If this MST region had hundreds of VLANs, and used PVST+, hundreds of sets of STP

messages would be used. With MST, only one set of STP messages is needed for each MST instance.

When connecting an MST region to a non-MST region or to a different MST region, MST makes the entire MST region appear to be a single switch, as shown on the right side of Figure 3-8. An MST region can guarantee loop-free behavior inside the MST region. To prevent loops over the CST links connecting the MST region to a non-MST region, MST participates in an STP instance with the switches outside the MST region. This additional STP instance is called the *Internal Spanning Tree (IST)*. When participating in STP with the external switches, the MST region is made to appear as if it is a single switch; the right side of Figure 3-8 depicts the STP view of the left side of the figure, as seen by the external switches.

Protecting STP

The final section in this chapter covers four switch configuration tools that protect STP from different types of problems or attacks, depending on whether a port is a trunk or an access port.

Root Guard and BPDU Guard: Protecting Access Ports

Network designers probably do not intend for end users to connect a switch to an access port that is intended for attaching end-user devices. However, it happens—for instance, someone just may need a few more ports in the meeting room down the hall, so they figure they could just plug a small, cheap switch into the wall socket.

The STP topology can be changed based on one of these unexpected and undesired switches being added to the network. For instance, this newly added and unexpected switch might have the lowest bridge ID and become the root. To prevent such problems, BPDU Guard and Root Guard can be enabled on these access ports to monitor for incoming BPDUs—BPDUs that should not enter those ports, because they are intended for single end-user devices. Both features can be used together. Their base operations are as follows:

- **BPDU Guard**—Enabled per port; error disables the port upon receipt of any BPDU.
- **Root Guard**—Enabled per port; ignores any received superior BPDUs to prevent a switch connected to this port from becoming root. Upon receipt of superior BPDUs, this switch puts the port in a loop-inconsistent state, ceasing forwarding and receiving frames until the superior BPDUs cease.

With BPDU Guard, the port does not recover from the *err-disabled* state unless additional configuration is added. You can tell the switch to change from err-disabled state back to an up state

after a certain amount of time. With Root Guard, the port recovers when the undesired superior BPDUs are no longer received.

UDLD and Loop Guard: Protecting Trunks

Both UniDirectional Link Detection (UDLD) and Loop Guard protect a switch trunk port from causing loops. Both features prevent switch ports from errantly moving from a blocking to a forwarding state when a unidirectional link exists in the network.

Unidirectional links are simply links for which one of the two transmission paths on the link has failed, but not both. This can happen as a result of miscabling, cutting one fiber cable, unplugging one fiber, GBIC problems, or other reasons. Because STP monitors incoming BPDUs to know when to reconverge the network, adjacent switches on a unidirectional link could both become forwarding, causing a loop, as shown in Figure 3-9.

Figure 3-9 *STP Problems with Unidirectional Links*

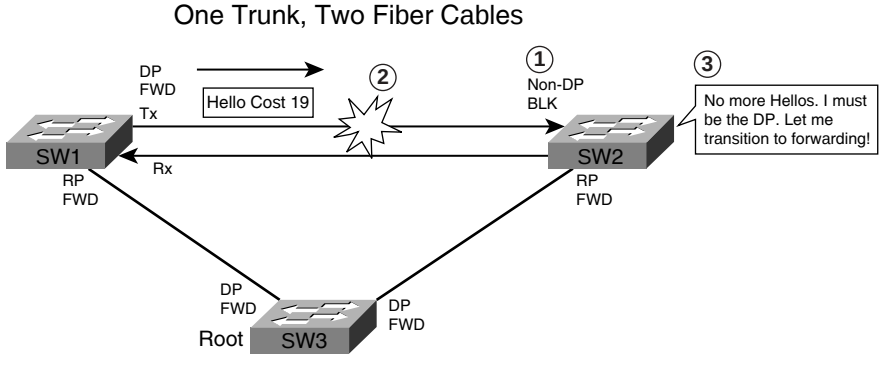


Figure 3-9 shows the fiber link between SW1 and SW2 with both cables. SW2 starts in a blocking state, but as a result of the failure on SW1's transmit path, SW2 ceases to hear Hellos from SW1. SW2 then transitions to forwarding state, and now all trunks on all switches are forwarding. Even with the failure of SW1's transmit cable, frames will now loop counter-clockwise in the network.

UDLD uses two modes to attack the unidirectional link problem. As described next, both modes, along with Loop Guard, solve the STP problem shown in Figure 3-9:

- **UDLD**—Uses Layer 2 messaging to decide when a switch can no longer receive frames from a neighbor. The switch whose transmit interface did not fail is placed into an err-disabled state.
- **UDLD aggressive mode**—Attempts to reconnect with the other switch (eight times) after realizing no messages have been received. If the other switch does not reply to the repeated additional messages, both sides become err-disabled.

- **Loop Guard**—When normal BPDUs are no longer received, the port does not go through normal STP convergence, but rather falls into an STP *loop-inconsistent* state.

In all cases, the formerly blocking port that would now cause a loop is prevented from migrating to a forwarding state. With both types of UDLD, the switch can be configured to automatically transition out of err-disabled state. With Loop Guard, the switch automatically puts the port back into its former STP state when the original Hellos are received again.

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most of the Cisco Press *Exam Certification Guides*, this book does not repeat information presented in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in the "Foundation Topics" section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Topic icon.

Table 3-11 lists the protocols mentioned in this chapter and their respective standards documents.

Table 3-11 *Protocols and Standards for Chapter 3*

Name	Standards Body
RSTP	IEEE 802.1W
MST	IEEE 802.1S
STP	IEEE 802.1D
LACP	IEEE 802.1AD
Dot1Q trunking	IEEE 802.1Q
PVST+	Cisco
PagP	Cisco

Table 3-12 lists the three key timers that impact STP convergence.

Table 3-12 *IEEE 802.1D STP Timers*

Timer	Default	Purpose
Hello	2 sec	Interval at which the root sends Hellos
Forward Delay	15 sec	Time that switch leaves a port in listening state and learning state; also used as the short CAM timeout timer
Maxage	20 sec	Time without hearing a Hello before believing that the root has failed

Table 3-13 lists some of the key IOS commands related to the topics in this chapter. (The command syntax for switch commands was taken from the *Catalyst 3550 Multilayer Switch Command Reference, 12.1(20)EA2*.) Also refer to Table 3-5 for several other commands.

Table 3-13 *Command Reference for Chapter 3*

Command	Description
spanning-tree mode {**mst** | **pvst** | **rapid-pvst**}	Global config command that sets the STP mode
[**no**] **spanning-tree vlan x**	Enables or disables STP inside a particular VLAN when using PVST+
spanning-tree vlan *vlan-id* {**forward-time** *seconds* | **hello-time** *seconds* | **max-age** *seconds* | **priority** *priority* | {**root** {**primary** | **secondary**} [**diameter** *net-diameter* [**hello-time** *seconds*]]}}	Global config command to set a variety of STP parameters
spanning-tree vlan *x* **cost** *y*	Interface subcommand used to set interface costs, per VLAN
spanning-tree vlan *x* **port-priority** *y*	Interface subcommand used to set port priority, per VLAN
channel-group *channel-group-number* **mode** {**auto** [**non-silent**] | **desirable** [**non-silent**] | **on** | **active** | **passive**}	Interface subcommand that places the interface into a port channel, and sets the negotiation parameters
channel-protocol {**lacp** | **pagp**}	Interface subcommand to define which protocol to use for EtherChannel negotiation
interface port-channel *port-channel-number*	Global command that allows configuration of parameters for the EtherChannel
spanning-tree portfast	Interface subcommand that enables PortFast on the interface
spanning-tree uplinkfast	Global command that enables UplinkFast
spanning-tree backbonefast	Global command that enables BackboneFast
spanning-tree mst *instance-id* **priority** *priority*	Global command used to set the priority of an MST instance
spanning-tree mst configuration	Global command that puts user in MST configuration mode
show spanning-tree root | **brief** | **summary**	EXEC command to show various details about STP operation
show spanning-tree uplinkfast | **backbonefast**	EXEC command to show various details about UplinkFast and BackboneFast

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

CST, STP, MST, RSTP, Hello timer, Maxage timer, Forward Delay timer, blocking state, forwarding state, listening state, learning state, disabled state, alternate state, discarding state, backup state, Root Port, Designated Port, superior BPDU, PVST+, UplinkFast, BackboneFast, PortFast, Root Guard, BPDU Guard, UDLD, Loop Guard, LACP, PAgP

Refer to the CD-based glossary to check your answers.

Further Reading

The topics in this chapter tend to be covered in slightly more detail in CCNP Switching exam preparation books. For more details on these topics, refer to *CCNP BCMSN Exam Certification Guide* and *CCNP Self-Study: CCNP BCMSN Exam Certification Guide*, Second Edition, listed in the introduction to this book.

Cisco LAN Switching, by Kennedy Clark and Kevin Hamilton, covers STP logic and operations in detail.

Part II: TCP/IP

Chapter 4 IP Addressing

Chapter 5 IP Services

Chapter 6 TCP/IP Transport and Application Services

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- IP
  - — Addressing
  - — IPv6

In addition, this chapter covers information related to the following specific CCIE Routing and Switching exam topics:

- CIDR
- NAT

CHAPTER 4

IP Addressing

Complete mastery of IP addressing and subnetting is required for any candidate to have a reasonable chance at passing both the CCIE written and lab exam. In fact, even the CCNA exam has fairly rigorous coverage of IP addressing and the related protocols. For the CCIE exam, understanding these topics is required to answer much deeper questions—for instance, a question might ask for the interpretation of the output of a **show ip bgp** command and a configuration snippet to decide what routes would be summarized into a new prefix. To answer such questions, the basic concepts and math behind subnetting need to be very familiar.

"Do I Know This Already?" Quiz

Table 4-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 4-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
IP Addressing and Subnetting	1–4	
CIDR, Private Addresses, and NAT	5–8	
IP Version 6	9	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. In what subnet does address 192.168.23.197/27 reside?

- **a.** 192.168.23.0
- **b.** 192.168.23.128
- **c.** 192.168.23.160
- **d.** 192.168.23.192
- **e.** 192.168.23.196

2. Router1 has four LAN interfaces, with IP addresses 10.1.1.1/24, 10.1.2.1/24, 10.1.3.1/24, and 10.1.4.1/24. What is the smallest summary route that could be advertised out a WAN link connecting Router1 to the rest of the network, if subnets not listed here were allowed to be included in the summary?

   a. 10.1.2.0/22
   b. 10.1.0.0/22
   c. 10.1.0.0/21
   d. 10.1.0.0/16

3. Router1 has four LAN interfaces, with IP addresses 10.22.14.1/23, 10.22.18.1/23, 10.22.12.1/23, and 10.22.16.1/23. Which one of the answers lists the smallest summary route(s) that could be advertised by R1 without also including subnets not listed in this question?

   a. 10.22.12.0/21
   b. 10.22.8.0/21
   c. 10.22.8.0/21 and 10.22.16.0/21
   d. 10.22.12.0/22 and 10.22.16.0/22

4. Which two of the following VLSM subnets, when taken as a pair, overlap?

   a. 10.22.21.128/26
   b. 10.22.22.128/26
   c. 10.22.22.0/27
   d. 10.22.20.0/23
   e. 10.22.16.0/22

5. Which of the following protocols or tools includes a feature like route summarization, plus administrative rules for global address assignment, with a goal of reducing the size of Internet routing tables?

   a. Classless interdomain routing
   b. Route summarization
   c. Supernetting
   d. Private IP addressing

6. Which of the following terms refers to a NAT feature that allows for significantly fewer IP addresses in the enterprise network as compared with the required public registered IP addresses?

   a. Static NAT
   b. Dynamic NAT
   c. Dynamic NAT with overloading
   d. PAT
   e. VAT

7. Consider an enterprise network using private class A network 10.0.0.0, and using NAT to translate to IP addresses in registered class C network 205.1.1.0. Host 10.1.1.1 has an open www session to Internet web server 198.133.219.25. Which of the following terms refers to the destination address of a packet, sent by the web server back to the client, when the packet has not yet made it back to the enterprise's NAT router?

   a. Inside Local
   b. Inside Global
   c. Outside Local
   d. Outside Global

8. Router1 has its fa0/0 interface, address 10.1.2.3/24, connected to an Enterprise network. Router1's S0/1 interface connects to an ISP, with the interface using a publicly-registered IP address of 171.1.1.1/30,. Which of the following commands could be part of a valid NAT overload configuration, with 171.1.1.1 used as the public IP address?

   a. **ip nat inside source list 1 int s0/1 overload**
   b. **ip nat inside source list 1 pool fred overload**
   c. **ip nat inside source list 1 171.1.1.1 overload**
   d. None of the answers is correct.

9. Which of the following show a legal equivalent representation of the IPv6 address 2000:0000:0000:0BEA:0000:0000:FE11:1111?

   a. 2000:0:0:BEA:0:0:FE11:1111
   b. 2000::BEA:0:0:FE11:1111
   c. 2000::BEA::FE11:1111
   d. 2:0:0:BEA::FE11:1111
   e. 2000:0:0:BEA::FE11:1111

Foundation Topics

IP Addressing and Subnetting

You need a postal address to receive letters; similarly, computers must use an IP address to be able to send and receive data using the TCP/IP protocols. Just as the postal service dictates the format and meaning of a postal address to aid the efficient delivery of mail, the TCP/IP protocol suite imposes some rules about IP address assignment so that routers can efficiently forward packets between IP hosts. This chapter begins with coverage of the format and meaning of IP addresses, with required consideration for how they are grouped to aid the routing process.

IP Addressing and Subnetting Review

First, here's a quick review of some of the core facts about IPv4 addresses that should be fairly familiar to you:

- 32-bit binary number.
- Written in "dotted decimal" notation (for example, 1.2.3.4), with each decimal octet representing 8 bits.
- Addresses are assigned to network interfaces, so computers or routers with multiple interfaces have multiple IP addresses.
- A computer with an IP address assigned to an interface is an *IP host*.
- A group of IP hosts that are not separated from each other by an IP router are in the same grouping.
- These groupings are called *networks*, *subnets*, or *prefixes*, depending on the context.
- IP hosts separated from another set of IP hosts by a router must be in separate groupings (network/subnet/prefix).

IP addresses may be analyzed using *classful* or *classless* logic, depending on the situation. Classful logic simply means that the main class A, B, and C rules from RFC 791 are considered. The next several pages present a classful view of IP addresses, as reviewed in Table 4-2.

With classful addressing, class A, B, and C networks can be identified as such by their first several bits (shown in the last column of Table 4-1) or by the range of decimal values for their first octets. Also, each class A, B, or C address has two parts (when not subnetted): a *network part* and a *host part*. The size of each is implied by the class, and can be stated explicitly using the default mask

for that class of network. For instance, mask 255.0.0.0, the default mask for class A networks, has 8 binary 1s and 24 binary 0s, representing the size of the network and host parts, respectively.

Table 4-2 *Classful Network Review*

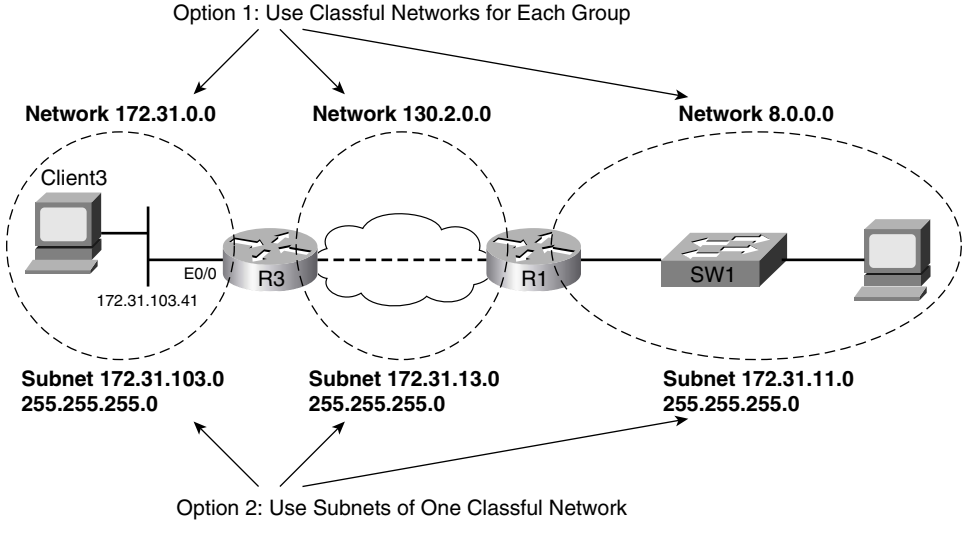
KEY POINT

Class of Address	Size of Network and Host Parts of the Addresses	Range of First Octet Values	Default Mask for Each Class of Network	Identifying Bits at Beginning of Address
A	8/24	1–126	255.0.0.0	0
B	16/16	128–191	255.255.0.0	10
C	24/8	192–223	255.255.255.0	110
D	—	224–239	—	1110
E	—	240–255	—	1111

Subnetting a Classful Network Number

With classful addressing, and no subnetting, an entire class A, B, or C network is needed on each individual instance of a data link. For example, Figure 4-1 shows a sample internetwork, with dashed-line circles representing the set of hosts that must be in the same IP network—in this case requiring three networks. Figure 4-1 shows two options for how IP addresses may be assigned and grouped together for this internetwork topology.

Figure 4-1 *Sample Internetwork with Two Alternatives for Address Assignment—Without and With Subnetting*



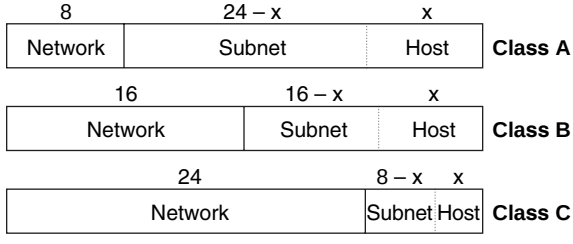
Option 1 uses three classful networks; however, it wastes a lot of IP addresses. For example, all hosts in class A network 8.0.0.0 must reside on the LAN on the right side of the figure.

Of course, the much more reasonable alternative is to reserve one classful IP network number, and use *subnetting* to subdivide that network into at least three subdivisions, called *subnets*. Option 2 (bottom of Figure 4-1) shows how to subdivide a class A, B, or C network into subnets.

To create subnets, the IP addresses must have three fields instead of just two—the network, *subnet*, and host. When using classful logic to interpret IP addresses, the size of the network part is still defined by classful rules—either 8, 16, or 24 bits based on class. To create the subnet field, the host field is shortened, as shown in Figure 4-2.

Figure 4-2 *Formats of IP Addresses when Subnetting*

KEY POINT



NOTE The term *internetwork* refers to a collection of computers and networking hardware; because TCP/IP discussions frequently use the term *network* to refer to a classful class A, B, or C IP network, this book uses the term internetwork to refer to an entire network topology, as shown in Figure 4-1.

To determine the size of each field in a subnetted IP address, you can follow the three easy steps shown in Table 4-3. Note that Figure 4-1 also showed alternative addressing for using subnets, with the last column in Table 4-3 showing the size of each field for that particular example, which used class B network 172.31.0.0, mask 255.255.255.0.

Table 4-3 *Finding the Size of the Network, Subnet, and Host Fields in an IP Address*

KEY POINT

Name of Part of the Address	Process to Find Its Size	Size per Figure 4-1 Example
Network	8, 16, or 24 bits based on class rules	16
Subnet	32 minus network and host bits	8
Host	Equal to the number of binary 0s in the mask	8

Comments on Classless Addressing

The terms *classless* and *classful* can be applied to three popular topics that are all related to IP. This chapter explains classful and classless IP addressing, which are relatively simple concepts. Two other chapters explain the other uses of the terms classless and classful: Chapter 7, "IP Forwarding (Routing)," describes classless/classful routing, and Chapter 8, "RIP Version 2," covers classless/classful routing protocols.

Classless IP addressing, simply put, means that class A, B, and C rules are ignored. Each address is viewed as a two-part address, formally called the *prefix* and the *host* parts of the address. The prefix simply states how many of the beginning bits of an IP address identify or define the group. It is the same idea as using the combined network and subnet parts of an address to identify a subnet. All the hosts with identical prefixes are in effect in the same group, which can be called a *subnet* or a *prefix*.

Just as a classful subnet must be listed with the subnet mask to know exactly which addresses are in the subnet, a prefix must be listed with its *prefix length*. The prefix itself is a dotted-decimal number. It is typically followed by a / symbol, after which the prefix length is listed. The prefix length is a decimal number that denotes the length (in bits) of the prefix. For example, 172.31.13.0/24 means a prefix of 172.31.13.0 and a prefix length of 24 bits. Also, the prefix can be implied by a subnet mask, with the number of 1s in the binary version of the mask implying the prefix length.

Classless and classful addressing are mainly just two ways to think about IP address formats. For the exam, make sure to understand both perspectives and the terminology used by each.

Subnetting Math

Knowing how to interpret the meaning of addresses and masks, routes and masks in the routing table, addresses and masks in ACLs, and configure route-filtering are all very important topics for the CCIE Routing and Switching written and lab exams. This section covers the binary math briefly, with coverage of some tricks to do the math quickly without binary math. Several subsequent chapters cover the configuration details of features that require this math.

Dissecting the Component Parts of an IP Address

First, deducing the size of the three parts (classful view) or two parts (classless view) of an IP address is important, because it allows you to analyze information about that subnet and other subnets. Every internetwork requires some number of subnets, and some number of hosts per subnet. Analyzing the format of an existing address, based on the mask or prefix length, allows

you to determine whether enough hosts per subnet exist, or whether enough subnets exist to support the number of hosts. The following list summarizes some of the common math facts about subnetting related to the format of IP addresses:

KEY POINT

- If a subnet has been defined with *y* host bits, there are $2^y - 2$ valid usable IP addresses in the subnet, because two numeric values are reserved.
- One reserved IP address in each subnet is the subnet number itself. This number, by definition, has binary 0s for all host bits. This number represents the subnet, and is typically seen in routing tables.
- The other reserved IP address in the subnet is the subnet broadcast address, which by definition has binary 1s for all host bits. This number can be used as a destination IP address to send a packet to all hosts in the subnet.
- When you are thinking classfully, if the mask implies *x* subnet bits, then 2^x possible subnets exist for that classful network, assuming the same mask is used throughout the network.
- Although there are no truly reserved values for the subnet numbers, two (lowest and highest values) may be discouraged from use in some cases:
  - **Zero subnet**—The subnet field is all binary 0s; in decimal, each zero subnet is the exact same dotted-decimal number as the classful network number, potentially causing confusion.
  - **Broadcast subnet**—The subnet field is all binary 1s; in decimal, this subnet's broadcast address is the same as the network-wide broadcast address, potentially causing confusion.

In Cisco routers, by default, zero subnets and broadcast subnets work fine. You can disable the use of the zero subnet with the **no ip subnet-zero** global command. The only time that using the zero subnet typically causes problems is when classful routing protocols are used.

Finding Subnet Numbers and Valid Range of IP Addresses—Binary

When examining an IP address and mask, the process of finding the subnet number, the broadcast address, and the range of valid IP addresses is as fundamental to networking as is addition and subtraction for advanced math. Possibly more so for the CCIE Routing and Switching lab exam, mastery of the math behind subnetting, which is the same basic math behind route summarization and filtering, will improve your speed completing complex configurations on the exam.

The range of valid IP addresses in a subnet begins with the number that is one larger than the subnet number, and ends with the address that is one smaller than the broadcast address for the

subnet. So, to determine the range of valid addresses, just calculate the subnet number and broadcast address, which can be done as follows:

KEY POINT

- **To derive the subnet number**—Perform a bit-wise Boolean AND between the IP address and mask
- **To derive the broadcast address**—Change all host bits in the subnet number from 0s to 1s

A bitwise Boolean AND means that you place two long binary numbers on top of each other, and then AND the two bits that line up vertically. (A Boolean AND results in a binary 1 only if both bits are 1; otherwise, the result is 0.) Table 4-4 shows an easy example based on subnet 172.31.103.0/24 from Figure 4-1.

Table 4-4 *Binary Math to Calculate the Subnet Number and Broadcast Address*

Address	172.31.103.41	1010 1100 0001 1111 0110 0111 **0010 1001**
Mask	255.255.255.0	1111 1111 1111 1111 1111 1111 **0000 0000**
Subnet Number (Result of AND)	172.31.103.0	1010 1100 0001 1111 0110 0111 **0000 0000**
Broadcast	172.31.103.255	1010 1100 0001 1111 0110 0111 **1111 1111**

Probably most everyone reading this already knew that the decimal subnet number and broadcast addresses shown in Table 4-4 were correct, even without looking at the binary math. The important part is to recall the binary process, and practice until you can confidently and consistently find the answer without using any binary math at all. The only parts of the math that typically trip people up are the binary to decimal and decimal to binary conversions. When working in binary, keep in mind that you will not have a calculator for the written exam, and that when converting to decimal, you always convert 8 bits at a time—even if an octet contains some prefix bits and some host bits. (Appendix D, "Decimal to Binary Conversion Table," contains a conversion table for your reference.)

Decimal Shortcuts to Find the Subnet Number and Valid Range of IP Addresses

Many of the IP addressing and routing related problems on the exam come back to the ability to solve a couple of classic basic problems. One of those problems runs as follows:

> Given an IP address and mask (or prefix length), determine the subnet number/prefix, broadcast address, and range of valid IP addresses.

If you personally can already solve such problems with only a few seconds' thought, even with tricky masks, then you can skip this section of the chapter. If you cannot solve such questions

easily and quickly, this section can help you learn some math shortcuts that allow you to find the answers without needing to use any Boolean math.

NOTE The next several pages of this chapter describe some algorithms you can use to find many important details related to IP addressing, without needing to convert to and from binary. In my experience, some people simply work better performing the math in binary until the answers simply start popping into their heads. Others find that the decimal shortcuts are more effective.

If you use the decimal shortcuts, it is best to practice them until you no longer really use the exact steps listed in this book; rather, the processes should become second nature. To that end, CD-only Appendix E, "IP Addressing Practice," lists several practice problems for each of the algorithms presented in this chapter.

To solve the "find the subnet/broadcast/range of addresses" type of problem, at least three of the four octets should have pretty simple math. For example, with a nice, easy mask like 255.255.255.0, the logic used to find the subnet number and broadcast address is intuitive to most people. The more challenging cases occur when the mask or prefix does not divide the host field at a byte boundary. For instance, the same IP address 172.31.103.41, with mask 255.255.252.0 (prefix /22), is actually in subnet 172.31.100.0. Working with the third octet in this example is the hard part, because the mask value for that octet is not 0 or 255; for the upcoming process, this octet is called the *interesting octet*. The following process finds the subnet number, using decimal math, even with a challenging mask:

Step 1 Find the mask octets of value 255; copy down the same octets from the IP address.

Step 2 Find the mask octets of value 0; write down 0s for the same octets.

Step 3 If one octet has not yet been filled in, that octet is the interesting octet. Find the subnet mask's value in the interesting octet, and subtract it from 256. Call this number the "magic number."

Step 4 Find the integer multiple of the magic number that is closest to, but not larger than, the interesting octet's value.

An example certainly helps, as shown in Table 4-5, with 172.31.103.41, mask 255.255.252.0. The table separates the address into its four component octets. In this example, the first, second, and fourth octets of the subnet number are easily found from Steps 1 and 2 in the process. Because the interesting octet is the third octet, the magic number is 256 – 252, or 4. The integer multiple of 4, closest to 103 but not exceeding 103, is 100—making 100 the subnet number's value in the third octet. (Note that you can use this same process even with an easy mask, and Steps 1 and 2 will give you the complete subnet number.)

Table 4-5 *Quick Math to Find the Subnet Number—172.31.103.41, 255.255.252.0*

	Octet				Comments
	1	**2**	**3**	**4**	
Address	172	31	103	41	
Mask	255	255	252	0	Equivalent to /22.
Subnet number results after Steps 1 and 2	172	31		0	Magic number will be 256 – 252 = 4.
Subnet number after completing the interesting octet	172	31	**100**	0	100 is the multiple of 4 closest to, but not exceeding, 103.

A similar process can be used to determine the subnet broadcast address. This process assumes that the mask is tricky. The detailed steps are as follows:

Step 1 Start with the subnet number.

Step 2 Decide which octet is interesting, based on which octet of the mask does not have a 0 or 255.

Step 3 For octets to the left of the interesting octet, copy down the subnet number's values into the place where you are writing down the broadcast address.

Step 4 For any octets to the right of the interesting octet, write down 255 for the broadcast address.

Step 5 Calculate the magic number: find the subnet mask's value in the interesting octet and subtract it from 256.

Step 6 Take the subnet number's interesting octet value, add the magic number to it, and subtract 1. Fill in the broadcast address's interesting octet with this number.

Table 4-6 shows the 172.31.103.41/22 example again, using this process to find the subnet broadcast address.

Table 4-6 *Quick Math to Find the Broadcast Address—172.31.103.41, 255.255.252.0*

	Octet				Comments
	1	**2**	**3**	**4**	
Subnet number (per Step 1)	172	31	100	0	
Mask (for reference)	255	255	252	0	Equivalent to /22
Results after Steps 1 to 4	172	31		255	Magic number will be 256 – 252 = 4
Subnet number after completing the empty octet	172	31	**103**	255	Subnet's third octet (100), plus magic number (4), minus 1 is 103

NOTE If you have read the last few pages to improve your speed at dissecting a subnet without requiring binary math, it is probably a good time to pull out the CD in the back of the book. CD-only Appendix E, "IP Addressing Practice," contains several practice problems for finding the subnet and broadcast address, as well as for many other math related to IP addressing.

Determining All Subnets of a Network—Binary

Another common question, typically simply a portion of a more challenging question on the CCIE written exam, relates to finding all subnets of a network. The base underlying question might be as follows:

> Given a particular class A, B, or C network, and a mask/prefix length used on all subnets of that network, what are the actual subnet numbers?

The answers can be found using binary or using a simple decimal algorithm. This section first shows how to answer the question using binary, using the following steps. Note that the steps include details that are not really necessary for the math part of the problem; these steps are mainly helpful for practicing the process.

KEY POINT

Step 1 Write down the binary version of the classful network number; that value is actually the zero subnet as well.

Step 2 Draw two vertical lines through the number, one separating the network and subnet parts of the number, the other separating the subnet and host part.

Step 3 Calculate the number of subnets, including the zero and broadcast subnet, based on 2^y, where y is the number of subnet bits.

Step 4 Write down y-1 copies of the binary network number below the first one, but leave the subnet field blank.

Step 5 Using the subnet field as a binary counter, write down values, top to bottom, in which the next value is 1 greater than the previous.

Step 6 Convert the binary numbers, 8 bits at a time, back to decimal.

This process takes advantage of a couple of facts about the binary form of IP subnet numbers:

- All subnets of a classful network have the same value in the network portion of the subnet number.
- All subnets of any classful network have binary 0s in the host portion of the subnet number.

Step 4 in the process simply makes you write down the network and host parts of each subnet number, because those values are easily predicted. To find the different subnet numbers, you then

just need to discover all possible different combinations of binary digits in the subnet field, because that is the only part of the subnet numbers that differs from subnet to subnet.

For example, consider the same class B network 172.31.0.0, with Static Length Subnet Masking (SLSM) assumed, and a mask of 255.255.224.0. Note that this example uses 3 subnet bits, so there will be 2^3 subnets. Table 4-7 lists the example.

Table 4-7 *Binary Method to Find All Subnets—Steps 1 Through 4*

	Octet				
Subnet	**1**	**2**	**3**		**4**
Network number/zero subnet	10101100	000 11111	000	00000	00000000
2nd subnet	10101100	000 11111		00000	00000000
3rd subnet	10101100	000 11111		00000	00000000
4th subnet	10101100	000 11111		00000	00000000
5th subnet	10101100	000 11111		00000	00000000
6th subnet	10101100	000 11111		00000	00000000
7th subnet	10101100	000 11111		00000	00000000
8th subnet (2^y = 8); broadcast subnet	10101100	000 11111		00000	00000000

At this point, you have the zero subnet recorded at the top, and you are ready to use the subnet field (the missing bits in the table) as a counter to find all possible values. Table 4-8 completes the process.

Table 4-8 *Binary Method to Find All Subnets—Step 5*

	Octet				
Subnet	**1**	**2**	**3**		**4**
Network number/zero subnet	10101100	00011111	**000**	00000	00000000
2nd subnet	10101100	00011111	**001**	00000	00000000
3rd subnet	10101100	00011111	**010**	00000	00000000
4th subnet	10101100	00011111	**011**	00000	00000000
5th subnet	10101100	00011111	**100**	00000	00000000
6th subnet	10101100	00011111	**101**	00000	00000000
7th subnet	10101100	00011111	**110**	00000	00000000
8th subnet (2^y = 8); broadcast subnet	10101100	00011111	**111**	00000	00000000

The final step to determine all subnets is simply to convert the values back to decimal. Take care to always convert 8 bits at a time. In this case, you end up with the following subnets: 172.31.0.0, 172.31.32.0, 172.31.64.0, 172.31.96.0, 172.31.128.0, 172.31.160.0, 172.31.192.0, and 172.31.224.0.

Determining All Subnets of a Network—Decimal

You may have noticed the trend in the third octet values in the subnets listed in the previous paragraph. When assuming SLSM, the subnet numbers in decimal do have a regular increment value, which turns out to be the value of the magic number. For example, instead of the binary math in the previous section, you could have thought the following:

- The interesting octet is the third octet.
- The magic number is 256 – 224 = 32.
- 172.31.0.0 is the zero subnet, because it is the same number as the network number.
- The other subnet numbers are increments of the magic number inside the interesting octet.

If that logic already clicks in your head, you can skip to the next section in this chapter. If not, the rest of this section outlines an decimal algorithm that takes a little longer pass at the same general logic. First, the question and the algorithm assume that the same subnet mask is used on all subnets of this one classful network—a feature sometimes called *Static Length Subnet Masking (SLSM)*. In contrast, *Variable Length Subnet Masking (VLSM)* means that different masks are used in the same classful network. The algorithm assumes a subnet field of 8 bits or less just to keep the steps uncluttered; for longer subnet fields, the algorithm can be easily extrapolated.

KEY POINT

Step 1 Write down the classful network number in decimal.

Step 2 For the first (lowest numeric) subnet number, copy the entire network number. That is the first subnet number, and is also the zero subnet.

Step 3 Decide which octet contains the entire subnet field; call this octet the interesting octet. (Remember, this algorithm assumes 8 subnet bits or less, so the entire subnet field will be in a single interesting octet.)

Step 4 Calculate the magic number by subtracting the mask's interesting octet value from 256.

Step 5 Copy down the previous subnet number's noninteresting octets onto the next line as the next subnet number; only one octet is missing at this point.

Step 6 Add the magic number to the previous subnet's interesting octet, and write that down as the next subnet number's interesting octet, completing the next subnet number.

Step 7 Repeat Steps 5 and 6 until the new interesting octet is 256. That subnet is not valid. The previously calculated subnet is the last valid subnet, and also the broadcast subnet.

For example, consider the same class B network 172.31.0.0, with SLSM assumed, and a mask of 255.255.224.0. Table 4-9 lists the example.

Table 4-9 *Subnet List Chart—172.31.0.0/255.255.224.0*

	Octet				Comments
	1	**2**	**3**	**4**	
Network number	172	31	0	0	Step 1 from the process.
Mask	255	255	224	0	Magic number is 256 – 224 = 32.
Subnet zero	172	31	0	0	Step 2 from the process.
First subnet	172	31	32	0	Steps 5 and 6; previous interesting octet 0, plus magic number (32).
Next subnet	172	31	64	0	32 plus magic number is 64.
Next subnet	172	31	96	0	64 plus magic number is 96.
Next subnet	172	31	128	0	96 plus magic number is 128.
Next subnet	172	31	160	0	128 plus magic number is 160.
Next subnet	172	31	192	0	160 plus magic number is 192.
Last subnet (broadcast)	172	31	224	0	The broadcast subnet in this case.
Invalid; easy-to-recognize stopping point	172	31	256	0	256 is out of range; when writing this one down, note that it is invalid, and that the previous one is the last valid subnet.

You can use this process repeatedly as needed until the answers start jumping out at you without the table and step-wise algorithm. For more practice, refer to CD-only Appendix E.

VLSM Subnet Allocation

So far in this chapter, most of the discussion has been about examining existing addresses and subnets. Before deploying new networks, or new parts of a network, you must give some thought to the ranges of IP addresses to be allocated. Also, when assigning subnets for different locations, you should assign the subnets with thought for how routes could then be summarized. This section covers some of the key concepts related to subnet allocation and summarization. (This section focuses on the concepts behind summarization; the configuration of route summarization is routing protocol–specific and thus is covered in the individual chapters covering routing protocols.)

Many organizations purposefully use SLSM to simplify operations. Additionally, many internetworks also use private IP network 10.0.0.0, with an SLSM prefix length of /24, and use NAT for connecting to the Internet. By using SLSM, particularly with a nice, easy prefix like /24, operations and troubleshooting can be a lot easier.

In some cases, VLSM is required or preferred when allocating addresses. VLSM is typically chosen when the address space is constrained to some degree. The VLSM subnet assignment strategy covered here complies with the strategy you may remember from the Cisco BSCI course or from reading the Cisco Press CCNP Routing certification books.

Similar to when assigning subnets with SLSM, you should use an easily summarized block of addresses for a new part of the network. Because VLSM network addresses are likely constrained to some degree, you should choose the specific subnets wisely. The general rules for choosing wisely are as follows:

KEY POINT

Step 1 Determine the shortest prefix length (in other words, the largest block) required.

Step 2 Divide the available address block into equal-sized prefixes based on the shortest prefix from Step 1.

Step 3 Allocate the largest required subnets/prefixes from the beginning of the IP address block, leaving some equal-sized unallocated address blocks at the end of the original large address block.

Step 4 Choose an unallocated block that you will further subdivide by repeating the first three steps, using the shortest required prefix length (largest address block) for the remaining subnets.

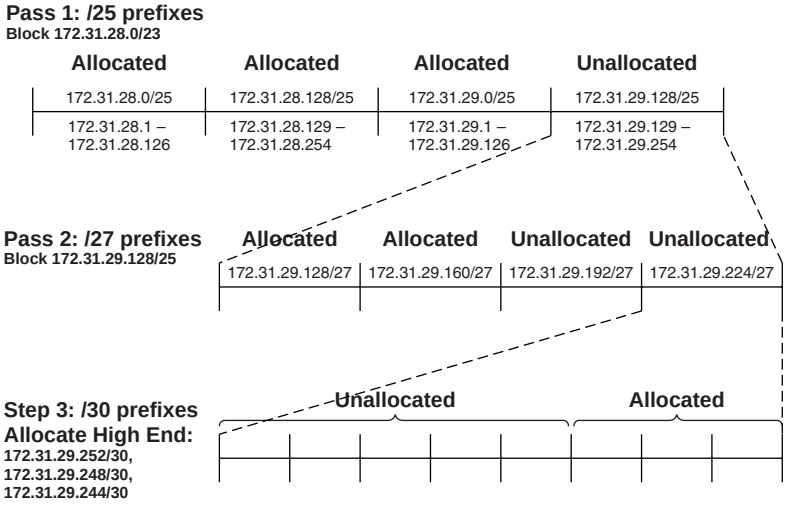
Step 5 When allocating very small address blocks for use on links between routers, consider using subnets at the end of the address range. This leaves the largest consecutive blocks available in case future requirements change.

For instance, imagine that a network engineer plans a new site installation. He allocates the 172.31.28.0/23 address block for the new site, expecting to use the block as a single summarized route. When planning, the engineer then subdivides 172.31.28.0/23 per the subnet requirements for the new installation, as shown in Figure 4-3. The figure shows three iterations through the VLSM subnet assignment process, because the requirements call for three different subnet sizes. Each iteration divides a remaining block into equal sizes, based on the prefix requirements of the subnets allocated at that step. Note that the small /30 prefixes were allocated from the end of the address range, leaving the largest possible consecutive address range for future growth.

Figure 4-3 *Example of VLSM Subnet Allocation Process*

KEY POINT

172.31.28.0.0/23 (172.31.28.0 Through 172.31.29.255)
Requirements:
3 /25's
2 /27's
3 /30's



Route Summarization Concepts

The ability to recognize and define how to most efficiently summarize existing address ranges is an important skill on both the written and lab exams. For the written exam, the question may not be as straightforward as, "What is the most efficient summarization of the following subnets?" Rather, the math required for such a question might simply be part of a larger question. Certainly, such math is required for the lab exam. This section looks at the math behind finding the best summarization; other chapters cover specific configuration commands.

Good IP address assignment practices should always consider the capabilities for route summarization. For instance, if a division of a company needs 15 subnets, an engineer needs to allocate those 15 subnets from the unused portions of the address block available to that internetwork. However, assigning subnets 10.1.101.0/24 through 10.1.115.0/24 would be a poor choice, because those do not easily summarize. Rather, allocate a range of addresses that can be easily summarized into a single route. For instance, subnets 10.1.96.0/24 through 10.1.110.0/24 can be summarized as a single 10.1.96.0/20 route, making those routes a better choice.

There are two main ways to think of the word "best" when you are looking for the "best summarization":

- **Inclusive summary routes**—A single summarized route that is as small a range of addresses as possible, while including all routes/subnets shown, and *possibly including subnets that do not currently exist.*

- **Exclusive summary routes**—As few as possible summarized routes that include all to-be-summarized address ranges, but *excluding all other routes/subnets.*

NOTE The terms *inclusive summary, exclusive summary,* and *candidate summary* are simply terms I invented for this book.

For instance, with the VLSM example in Figure 4-3, the network engineer purposefully planned so that an inclusive summary of 172.31.28.0/23 could be used. Even though not all subnets are yet allocated from that address range, the engineer is likely saving the rest of that address range for future subnets at that site, so summarizing using an inclusive summary is reasonable. In other cases, typically when trying to summarize routes in an internetwork for which summarization was not planned, the summarization must exclude routes that are not explicitly listed, because those address ranges may actually be used in another part of the internetwork.

Finding Inclusive Summary Routes—Binary

Finding the best inclusive summary lends itself to a formal binary process, as well as to a formal decimal process. The binary process runs as follows:

Step 1 Write down the binary version of each component subnet, one on top of the other.

Step 2 Inspect the binary values to find how many consecutive bits have the exact same value in all component subnets. That number of bits is the prefix length.

Step 3 Write a new 32-bit number at the bottom of the list by copying *y* bits from the prior number, *y* being the prefix length. Write binary 0s for the remaining bits. This is the inclusive summary.

Step 4 Convert the new number to decimal, 8 bits at a time.

Table 4-10 shows an example of this process, using four routes, 172.31.20.0, .21.0, .22.0, and .23.0, all with prefix /24. The second example adds 172.31.24.0 to that same list.

Table 4-10 *Example of Finding the Best Inclusive Summary—Binary*

	Octet 1	Octet 2	Octet 3		Octet 4
172.31.20.0/24	10101100	00011111	000101	00	00000000
172.31.21.0/24	10101100	00011111	000101	01	00000000
172.31.22.0/24	10101100	00011111	000101	10	00000000
172.31.23.0/24	10101100	00011111	000101	11	00000000
Prefix length: 22					
Inclusive summary	10101100	00011111	000101	**00**	**00000000**

The trickiest part is Step 2, in which you have to simply look at the binary values and find the point at which the bits are no longer equal. You can shorten the process by, in this case, noticing that all component subnets begin with 172.31, meaning that the first 16 bits will certainly have the same values.

Finding Inclusive Summary Routes—Decimal

To find the same inclusive summary using only decimal math, use the following process. The process works just fine with variable prefix lengths and nonconsecutive subnets.

Step 1 Count the number of subnets; then, find the smallest value of *y*, such that 2^y => that number of subnets.

Step 2 For the next step, use a prefix length based on the longest prefix length of the component subnets, minus *y*.

Step 3 Pretend that the lowest numeric subnet number in the list of component subnets is an IP address. Using the new, smaller prefix from Step 2, calculate the subnet number in which this pretend address resides.

Step 4 Repeat Step 3 for the largest numeric component subnet number and the same prefix. If it is the same subnet derived as in Step 3, the resulting subnet is the best summarized route, using the new prefix.

Step 5 If Steps 3 and 4 do not yield the same resulting subnet, repeat Steps 3 and 4 with another new prefix length of 1 less than the last prefix length.

Table 4-11 shows two examples of the process. The first example has four routes, 172.31.20.0, .21.0, .22.0, and .23.0, all with prefix /24. The second example adds 172.31.24.0 to that same list.

Table 4-11 *Example of Finding the Best Summarizations*

Step	Range of .20.0, .21.0, .22.0, and .23.0, /24	Same Range, Plus 172.31.24.0
Step 1	$2^2 = 4$, $y = 2$	$2^3 = 8$, $y = 3$
Step 2	24 – 2 = 22	24 – 3 = 21
Step 3	Smallest subnet 172.31.20.0, with /22, yields **172.31.20.0/22**	Smallest subnet 172.31.20.0, with /21, yields **172.31.16.0/21**
Step 4	Largest subnet 172.31.23.0, with /22, yields **172.31.20.0/22**	Largest subnet 172.31.24.0, with /22, yields **172.31.24.0/21**
Step 5	—	21 – 1 = 20; new prefix
Step 3, 2nd time	—	172.31.16.0/20
Step 4, 2nd time	—	172.31.16.0/20; the same as prior step, so that is the answer

With the first example, Steps 3 and 4 yielded the same answer, which means that the best inclusive summary had been found. With the second example, a second pass through the process was required. CD-only Appendix E contains several practice problems to help you develop speed and make this process second nature.

Finding Exclusive Summary Routes—Binary

A similar process, listed next, can be used to find the exclusive summary. Keep in mind that the best exclusive summary can be comprised of multiple summary routes. Once again, to keep it simple, the process assumes SLSM.

Step 1 Find the best *inclusive* summary route; call it a *candidate exclusive* summary route.

Step 2 Determine if the candidate summary includes any address ranges it should not. To do so, compare the summary's implied address range with the implied address ranges of the component subnets.

Step 3 If the candidate summary only includes addresses in the ranges implied by the component subnets, the candidate summary is part of the best exclusive summarization of the original component subnets.

Step 4 If instead the candidate summary includes some addresses that match the candidate summary routes and some addresses that do not, split the current candidate summary in half, into two new candidate summary routes, each with a prefix 1 *longer* than before.

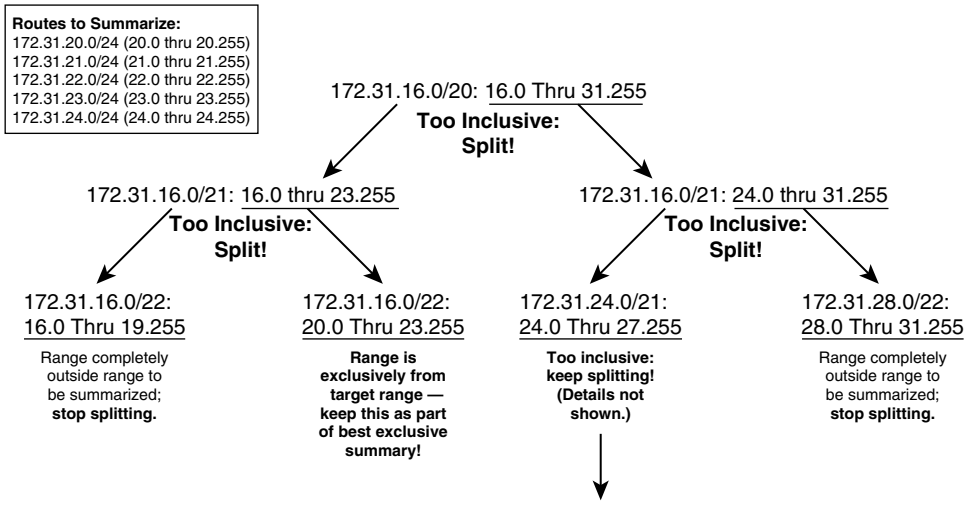
Step 5 If the candidate summary only includes addresses outside the ranges implied by the component subnets, the candidate summary is not part of the best exclusive summarization, and it should not be split further.

Step 6 Repeat Steps 2 through 4 for each of the two possible candidate summary routes created at Step 4.

For example, take the same five subnets used with the inclusive example—172.31.20.0/24, .21.0, .22.0, .23.0, and .24.0. The best inclusive summary is 172.31.16.0/20, which implies an address range of 172.31.16.0 to 172.31.31.255—clearly, it includes more addresses than the original five subnets. So, repeat the process of splitting the summary in half, and repeating, until summaries are found that do not include any unnecessary address ranges. Figure 4-4 shows the idea behind the logic.

The process starts with one candidate summary. If it includes some addresses that need to be summarized and some addresses it should not summarize, split it in half, and try again with each half. Eventually, the best exclusive summary routes are found, or the splitting keeps happening until you get back to the original routes. In fact, in this case, after a few more splits (not shown), the process ends up splitting to 172.31.24.0/24, which is one of the original routes—meaning that 172.31.24.0/24 cannot be summarized any further in this example.

Figure 4-4 *Example of Process to Find Exclusive Summary Routes*



CIDR, Private Addresses, and NAT

The sky was falling in the early 1990s in that the commercialization of the Internet was rapidly depleting the IP Version 4 address space. Also, Internet routers' routing tables were doubling annually (at least). Without some changes, the incredible growth of the Internet in the 1990s would have been stifled.

To solve the problems associated with this rapid growth, several short-term solutions were created, as well as an ultimate long-term solution. The short-term solutions included classless interdomain routing (CIDR), which helps reduce the size of routing tables by aggregating routes, and Network Address Translation (NAT), which reduces the number of required public IP addresses used by each organization or company. This section covers the details of CIDR and NAT, plus a few related features. The final major section of this chapter looks at the long-term solution to this problem—namely, IP Version 6 (IPv6).

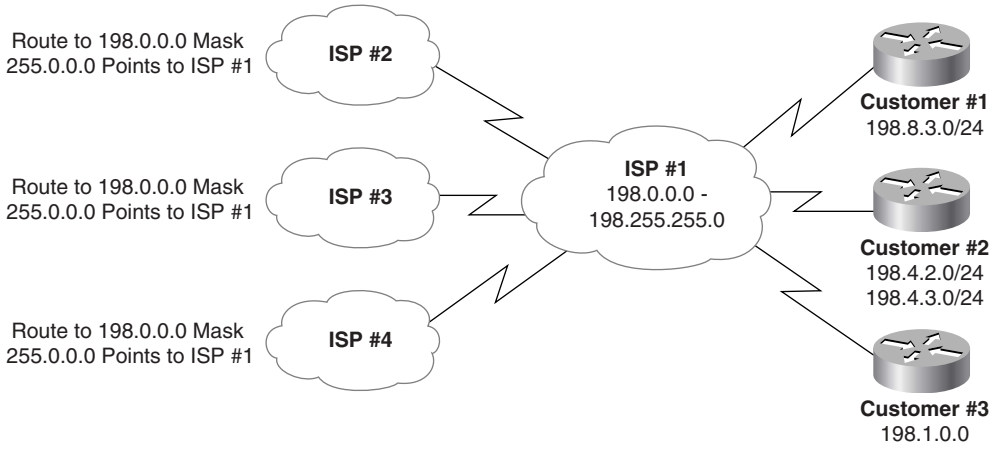
Classless Interdomain Routing

CIDR is a convention defined in RFCs 1517 through 1520 that calls for aggregating routes for multiple classful network numbers into a single routing table entry. The primary goal of CIDR is to improve the scalability of Internet routers' routing tables. Imagine the implications of an Internet router being burdened by carrying a route to every class A, B, and C network on the planet!

CIDR uses both technical tools and administrative strategies to reduce the size of the Internet routing tables. Technically, CIDR uses route summarization, but with Internet scale in mind.

For instance, CIDR might be used to allow a large ISP to control a range of IP addresses from 198.0.0.0 to 198.255.255.255, with the improvements to routing shown in Figure 4-5.

Figure 4-5 *Typical Use of CIDR*



ISPs 2, 3, and 4 need only one route (198.0.0.0/8) in their routing tables to be able to forward packets to all destinations that begin with 198. Note that this summary actually summarizes multiple class C networks—a typical feature of CIDR. ISP 1's routers contain more detailed routing entries for addresses beginning with 198, based on where they allocate IP addresses for their customers. ISP 1 would reduce its routing tables similarly with large ranges used by the other ISPs.

KEY POINT CIDR attacks the problem of large routing tables via administrative means as well. As shown in Figure 4-5, ISPs are assigned contiguous blocks of addresses to use when assigning addresses for their customers. Likewise, regional authorities are assigned large address blocks, so when individual companies ask for registered public IP addresses, they ask their regional registry to assign them an address block. As a result, addresses assigned by the regional agency will at least be aggregatable into one large geographic region of the world. For instance, the Latin American and Caribbean Internet Addresses Registry (LACNIC, http://www.lacnic.net) administers the IP address space of the Latin American and Caribbean region (LAC) on behalf of the Internet community.

In some cases, the term CIDR is used a little more generally than the original intent of the RFCs. Some texts use the term CIDR synonymously with the term route summarization. Others use the term CIDR to refer to the process of summarizing multiple classful networks together. In other cases, when an ISP assigns subsets of a classful network to a customer who does not need an entire class C network, the ISP is essentially performing subnetting; once again, this idea sometimes gets categorized as CIDR. But CIDR itself refers to the administrative assignment of large address blocks, and the related summarized routes, for the purpose of reducing the size of the Internet routing tables.

NOTE Because CIDR defines how to combine routes for multiple classful networks into a single route, some people think of this process as being the opposite of subnetting. As a result, many people refer to CIDR's summarization results as *supernetting*.

Private Addressing

One of the issues with Internet growth was the assignment of all possible network numbers to a small number of companies or organizations. Private IP addressing helps to mitigate this problem by allowing computers that will never be directly connected to the Internet to not use public, Internet-routable addresses. For IP hosts that will purposefully have no direct Internet connectivity, you can use several reserved network numbers, as defined in RFC 1918 and listed in Table 4-12.

Table 4-12 *RFC 1918 Private Address Space*

KEY POINT

Range of IP Addresses	Class of Networks	Number of Networks
10.0.0.0 to 10.255.255.255	A	1
172.16.0.0 to 172.31.255.255	B	16
192.168.0.0 to 192.168.255.255	C	256

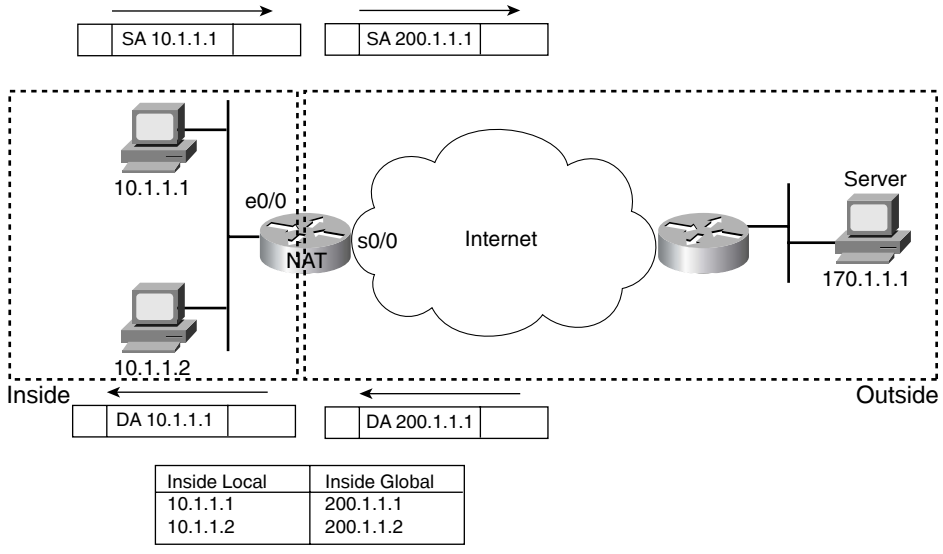
In other words, any organization can use these network numbers. However, no organization is allowed to advertise these networks using a routing protocol on the Internet. Furthermore, all Internet routers should be configured to reject these routes.

Network Address Translation

NAT, defined in RFC 1631, allows a host that does not have a valid registered IP address to communicate with other hosts on the Internet. NAT has gained such wide-spread acceptance that the majority of enterprise IP networks today use private IP addresses for most hosts on the network and use a small block of public IP addresses, with NAT translating between the two.

NAT translates, or changes, one or both IP addresses inside a packet as it passes through a router. (Many firewalls also perform NAT; for the CCIE Routing and Switching exam, you do not need to know NAT implementation details on firewalls.) In most cases, NAT changes the (typically private range) addresses used inside an enterprise network into address from the public IP address space. For instance, Figure 4-6 shows static NAT in operation; the enterprise has registered class C network 200.1.1.0/24, and uses private class A network 10.0.0.0/8 for the hosts inside its network.

Figure 4-6 *Basic NAT Concept*



Beginning with the packets sent from a PC on the left to the server on the right, the private IP source address 10.1.1.1 is translated to a public IP address of 200.1.1.1. The client sends a packet with source address 10.1.1.1, but the NAT router changes the source to 200.1.1.1—a registered public IP address. Once the server receives a packet with source IP address 200.1.1.1, the server thinks it is talking to host 200.1.1.1, so it replies with a packet sent to destination 200.1.1.1. The NAT router then translates the destination address (200.1.1.1) back to 10.1.1.1.

Figure 4-6 provides a good backdrop for the introduction of a couple of key terms, *Inside Local* and *Inside Global*. Both terms take the perspective of the owner of the enterprise network. In Figure 4-6, address 10.1.1.1 is the Inside Local address, and 200.1.1.1 is the Inside Global address. Both addresses represent the client PC on the left, which is *inside the enterprise network*. Address 10.1.1.1 is from the enterprise's IP address space, which is only *locally* routable inside the enterprise—hence the term Inside Local. Address 200.1.1.1 represents the local host, but the address is from the globally routable public IP address space—hence the name Inside Global. Table 4-13 lists and describes the four main NAT address terms.

Table 4-13 *NAT Terminology*

KEY POINT

Name	Location of Host Represented by Address	IP Address Space in Which Address Exists
Inside Local address	Inside the enterprise network	Part of the enterprise IP address space; typically a private IP address
Inside Global address	Inside the enterprise network	Part of the public IP address space

Table 4-13 *NAT Terminology (Continued)*

Name	Location of Host Represented by Address	IP Address Space in Which Address Exists
Outside Local address	In the public Internet; or, outside the enterprise network	Part of the enterprise IP address space; typically a private IP address
Outside Global address	In the public Internet; or, outside the enterprise network	Part of the public IP address space

Static NAT

Static NAT works just like the example in Figure 4-6, but with the IP addresses statically mapped to each other via configuration commands. With static NAT:

- A particular Inside Local address always maps to the same Inside Global (public) IP address.
- If used, each Outside Local address always maps to the same Outside Global (public) IP address.
- Static NAT does not conserve public IP addresses.

Although static NAT does not help with IP address conservation, static NAT does allow an engineer to make an inside server host available to clients on the Internet, because the inside server will always use the same public IP address.

Example 4-1 shows a basic static NAT configuration based on Figure 4-6. Conceptually, the NAT router has to identify which interfaces are inside (attach to the enterprise's IP address space) or outside (attach to the public IP address space). Also, the mapping between each Inside Local and Inside Global IP address must be made. (Although not needed for this example, outside addresses can also be statically mapped.)

Example 4-1 *Static NAT Configuration*

KEY POINT

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! E0/0 attaches to the internal Private IP space, so it is configured as an inside
! interface.
interface Ethernet0/0
 ip address 10.1.1.3 255.255.255.0
 ip nat inside
! S0/0 is attached to the public Internet, so it is defined as an outside
! interface.
interface Serial0/0
 ip address 200.1.1.251 255.255.255.0
 ip nat outside
```

continues

Example 4-1 *Static NAT Configuration (Continued)*

```
! Next, two inside addresses are mapped, with the first address stating the
! Inside Local address, and the next stating the Inside Global address.
ip nat inside source static 10.1.1.2 200.1.1.2
ip nat inside source static 10.1.1.1 200.1.1.1
! Below, the NAT table lists the permanent static entries from the configuration.
NAT# show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
--- 200.1.1.1          10.1.1.1           ---                --
--- 200.1.1.2          10.1.1.2           ---                ---
```

The router is performing NAT only for inside addresses. As a result, the router processes packets entering E0/0—packets that could be sent by inside hosts—by examining the source IP address. Any packets with a source IP address listed in the Inside Local column of the **show ip nat translations** command output (10.1.1.1 or 10.1.1.2) will be translated to source address 200.1.1.1 or 200.1.1.2, respectively, per the NAT table. Likewise, the router examines the destination IP address of packets entering S0/0, because those packets would be destined for inside hosts. Any such packets with a destination of 200.1.1.1 or .2 will be translated to 10.1.1.1 or .2, respectively.

In cases with static outside addresses being configured, the router also looks at the destination IP address of packets sent from the inside to the outside interfaces, and the source IP address of packets sent from outside interfaces to inside interfaces.

Dynamic NAT Without PAT

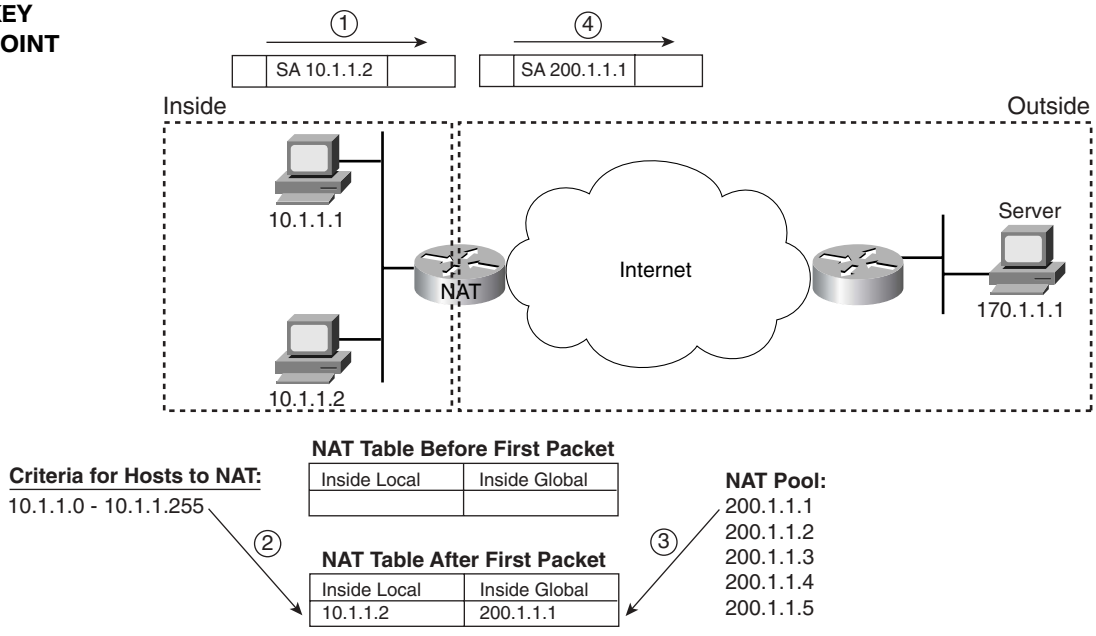
Dynamic NAT (without PAT), like static NAT, creates a one-to-one mapping between an Inside Local and Inside Global address. However, unlike static NAT, it does so by defining a set or pool of Inside Local and Inside Global addresses, and dynamically mapping pairs of addresses as needed. For example, Figure 4-7 shows a pool of five Inside Global IP addresses—200.1.1.1 through 200.1.1.5. NAT has also been configured to translate any Inside Local addresses whose address starts with 10.1.1.

The numbers 1, 2, and 3 in Figure 4-7 refer to the following sequence of events:

1. Host 10.1.1.2 starts by sending its first packet to the server at 170.1.1.1.
2. As the packet enters the NAT router, the router applies some matching logic to decide if the packet should have NAT applied. Because the logic has been configured to mean “translate Inside Local addresses that start with 10.1.1,” the router dynamically adds an entry in the NAT table for 10.1.1.2 as an Inside Local address.
3. The NAT router needs to allocate a corresponding IP address from the pool of valid Inside Global addresses. It picks the first one available (200.1.1.1 in this case) and adds it to the NAT table to complete the entry.

Figure 4-7 *Dynamic NAT*

KEY POINT



With the completion of step 3, the NAT router can actually translate the source IP address, and forward the packet. Note that as long as the dynamic NAT entry exists in the NAT table, only host 10.1.1.2 can use Inside Global IP address 200.1.1.1.

Overloading NAT with Port Address Translation

As mentioned earlier, NAT is one of the key features that helped to reduce the speed at which the IPv4 address space was being depleted. *NAT overloading*, also known as *Port Address Translation* (*PAT)*, is the NAT feature that actually provides the significant savings of IP addresses. The key to understanding how PAT works is to consider the following: From a server's perspective, there is no significant difference between 100 different TCP connections, each from a different host, and 100 different TCP connections all from the same host.

PAT works by making large numbers of TCP or UDP flows from many Inside Local hosts appear to be the same number of large flows from one (or a few) host's Inside Global addresses. With PAT, instead of just translating the IP address, NAT also translates the port numbers as necessary. And because the port number fields are 16 bits in length, each Inside Global IP address can support over 65,000 concurrent TCP and UDP flows. For instance, in a network with 1000 hosts, a single public IP address used as the only Inside Global address could handle an average of six concurrent flows from each host to and from hosts on the Internet.

Dynamic NAT and PAT Configuration

Like static NAT, dynamic NAT configuration begins with identifying the inside and outside interfaces. Additionally, the set of Inside Local addresses is configured with the **ip nat inside** global command. If you are using a pool of public Inside Global addresses, the set of addresses is defined by the **ip nat pool** command. Example 4-2 shows a dynamic NAT configuration based on the internetwork shown in Figure 4-7. The example defines 256 Inside Local addresses and two Inside Global addresses.

Example 4-2 *Dynamic NAT Configuration*

KEY POINT

```
! First, the ip nat pool fred command lists a range of IP addresses. The ip nat
! inside source list 1 pool fred command points to ACL 1 as the list of Inside
! Local addresses, with a cross-reference to the pool name.
interface Ethernet0/0
 ip address 10.1.1.3 255.255.255.0
 ip nat inside
!
interface Serial0/0
 ip address 200.1.1.251 255.255.255.0
 ip nat outside
!
ip nat pool fred 200.1.1.1 200.1.1.2 netmask 255.255.255.252
ip nat inside source list 1 pool fred
!
access-list 1 permit 10.1.1.0 0.0.0.255
! Next, the NAT table begins as an empty table, because no dynamic entries had
! been created at that point.
NAT# show ip nat translations

! The NAT statistics show that no hits or misses have occurred. Hits occur when
! NAT looks for a mapping, and finds one. Misses occur when NAT looks for a NAT
! table entry, does not find one, and then needs to dynamically add one.
NAT# show ip nat statistics
Total active translations: 0 (0 static, 0 dynamic; 0 extended)
Outside interfaces:
  Serial0/0
Inside interfaces:
  Ethernet0/0
Hits: 0  Misses: 0
Expired translations: 0
Dynamic mappings:
-- Inside Source
access-list 1 pool fred refcount 0
 pool fred: netmask 255.255.255.252
    start 200.1.1.1 end 200.1.1.2
    type generic, total addresses 2, allocated 0 (0%), misses 0
! At this point, a Telnet session from 10.1.1.1 to 170.1.1.1 started.
```

Example 4-2 *Dynamic NAT Configuration (Continued)*

```
! Below, the 1 "miss" means that the first packet from 10.1.1.2 did not have a
! matching entry in the table, but that packet triggered NAT to add an entry to the
! NAT table. Host 10.1.1.2 has then sent 69 more packets, noted as "hits" because
! there was an entry in the table.
NAT# show ip nat statistics
Total active translations: 1 (0 static, 1 dynamic; 0 extended)
Outside interfaces:
  Serial0/0
Inside interfaces:
  Ethernet0/0
Hits: 69  Misses: 1
Expired translations: 0
Dynamic mappings:
-- Inside Source
access-list 1 pool fred refcount 1
 pool fred: netmask 255.255.255.252
    start 200.1.1.1 end 200.1.1.2
    type generic, total addresses 2, allocated 1 (50%), misses 0
! The dynamic NAT entry is now displayed in the table.
NAT# show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
--- 200.1.1.1          10.1.1.2           ---                ---
! Below, the configuration uses PAT via the overload parameter. Could have used the
! ip nat inside source list 1 int s0/0 overload command instead, using a single
! IP Inside Global IP address.
NAT(config)# no ip nat inside source list 1 pool fred
NAT(config)# ip nat inside source list 1 pool fred overload
! To test, the dynamic NAT entries were cleared after changing the NAT
! configuration. Before the next command was issued, host 10.1.1.1 had created two
! Telnet connections, and host 10.1.1.2 created 1 more TCP connection.
NAT# clear ip nat translations *
! Before the next command was issued, host 10.1.1.1 had created two
! Telnet connections, and host 10.1.1.2 created 1 more TCP connection. Note that
! all three dynamically mapped flows use common Inside Global 200.1.1.1.
NAT# show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
tcp 200.1.1.1:3212     10.1.1.1:3212      170.1.1.1:23       170.1.1.1:23
tcp 200.1.1.1:3213     10.1.1.1:3213      170.1.1.1:23       170.1.1.1:23
tcp 200.1.1.1:38913    10.1.1.2:38913     170.1.1.1:23       170.1.1.1:23
```

IP Version 6

The ultimate solution to rapidly growing Internet routing tables and IPv4 address depletion was the development of IPv6, which defines 128-bit source and destination addresses. At the risk of being derided 20 years from now, I'll venture a guess that IPv6 has more addresses than we'll ever need. IPv6 can support over a trillion, trillion IP addresses per person on the planet—with plenty

of publicly routable addresses for everyone. Plus, the structure is well established for CIDR-like allocation of address blocks, keeping Internet routing tables small.

So, why hasn't the world moved to IPv6? Well, CIDR, NAT, and other tools have made the IPv4 address space much more manageable. The killer app that drives IPv6 may well be the growth of mobile wireless phones and handhelds, or it may be the sensor revolution. Of more direct interest is that the most basic features of IPv6 now appear on the written and lab CCIE Routing and Switching exams.

IPv6 Address Formats

IPv6 addresses have eight *quartets* of hex digits, separated by colons. Each quartet consists of four hex digits, which together represent 16 bits. The rules for encoding the actual hex values are as follows:

KEY POINT

- Each quartet is separated by a colon (:).
- In a quartet, leading hex 0s can optionally be omitted.
- If one or more consecutive quartets are hex 0000, then the set of consecutive all-0 quartets can be represented as a null quartet (::), no matter how many consecutive all-0 quartets are in this range.
- Only one use of :: is allowed in a single IPv6 address.

For example, the following IPv6 address is shown in three different valid formats. The first one uses no shortcuts; the next removes leading 0s in each quartet; and the last abbreviates multiple, consecutive, all-0 quartets with ::.

```
0123:0078:0000:0000:9ABC:0000:0000:DEF0
123:78:0:0:9ABC:0:0:DEF0
123:78::9ABC:0:0:DEF0
```

All three versions are legal and common, but router command output generally shows the briefest form so that command output best fits on the visible screen. Also, notice that there were two places in the address with two consecutive all-0 quartets. The :: shortcut implies one or more consecutive all-0 quartets, so using this abbreviation in more than one place would be ambiguous.

Aggregatable Global Unicast Addresses

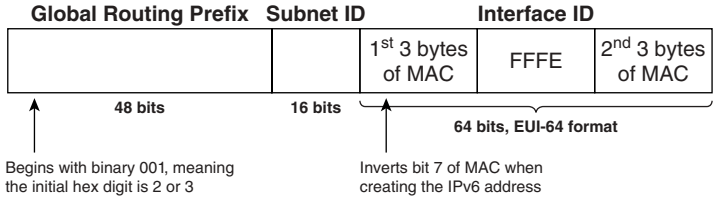
IPv6 defines several different types of unicast addresses. The format used for publicly registered addresses is called *aggregatable global unicast*. The term *aggregatable* refers to fact that these addresses can be easily aggregated to reduce the problem of large Internet routing tables. The term *global* refers to the fact that the address is a registered public IP address, routable through the global Internet. In short, when a company obtains a registered IPv6 prefix from a Regional Internet

Authority, the addresses follow the aggregatable global unicast format. For short, most people call them *global addresses*.

The original global address format was a little more complicated. The current version of the format contains three fixed-length major parts. Conceptually, the format is like the network, subnet, and host organization of an IPv4 address, except the fixed-length fields do not require a subnet mask equivalent. Figure 4-8 shows the format and details.

Figure 4-8 *Aggregatable Global Address Format*

KEY POINT



With IPv4, to use public addresses, you obtain a prefix from one of the numbering authorities—maybe a classful network, or maybe just a small block defined with a prefix. With IPv6, you would get an assigned block with a 48-bit prefix. The first 3 bits are always binary 001, leaving 45 bits in the prefix that can be assigned by IANA and its member local, national, or regional registries. As a result, there are 2^{45} possible prefixes to assign, or roughly 32 trillion possible prefixes. And, as is the case with CIDR, large numbers of consecutive prefixes are assigned to ISPs and local, national, or regional registries to aid in managing the growth of Internet routing tables.

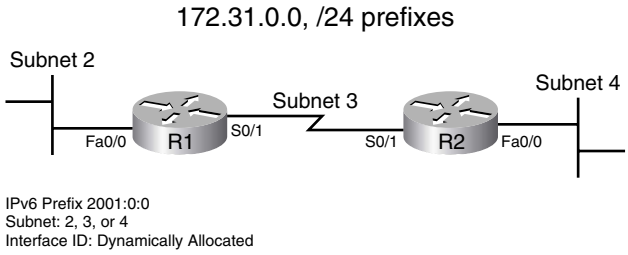
Once you have a registered prefix, you can subdivide it using the *subnet ID field*. For each link, you could pick a different value for this field, just like you pick different subnet numbers for IPv4. With 16 bits, each IPv6 prefix holder can assign more than 65,000 subnets within that address space.

Finally, IPv6 global addresses allow for the dynamic assignment of the interface ID portion of the addresses, as well as for static assignment. And, because IEEE MAC addresses already have a proven method to ensure global uniqueness, IPv6 uses the MAC to create the final 64 bits of the address, called the *interface ID*. To create the interface ID dynamically, IPv6 splits the MAC in half, and puts hex FFFE in the middle. (The format of the interface ID portion is called *EUI-64*.) For non-LAN interfaces on a router, a MAC is borrowed from another interface, or created from the router serial number, much like routers create the node part of Novell IPX addresses on a router.

Simple IPv6 Configuration

Figure 4-9 shows a small, sample network that is used in this section to show basic IPv6 connectivity using global addresses.

Figure 4-9 *Sample Dual-Stack IPv4/IPv6 Network*



The routers use *dual stacks* (both IPv4 and IPv6 addresses are assigned to the interfaces) for this network. Hosts with dual stacks (running both IPv4 and IPv6) would be able to send IPv4 or IPv6 packets, and the routers could accommodate either. This relatively simple example of IPv6 uses global addresses and a minimal OSPF configuration. Example 4-3 shows the configuration and a working **ping**.

Example 4-3 *Configuring Dual-Stack IPv4/IPv6 with OSPF*

KEY POINT

```
! The ipv6 unicast-routing command enables IPv6 forwarding on this router.
! The ipv6 cef command is optional, but it does cause CEF switching of IPv6.
Router1# show run
!lines omitted for brevity
!
ip cef
ipv6 unicast-routing
ipv6 cef
! The ipv6 address command lists the global routing prefix and IPV6 subnet 2.
! The /64 means that only the first 64 bits defines the prefix, and the
! eui-64 parameter means that the router should dynamically create the rest of the
! address for the interface. Note that the :: at the end of the address implies
! 3 quartets of 0s, which will be replaced with the EUI-64 interface id.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
interface FastEthernet0/0
 mac-address 0200.1111.1111
 ip address 172.31.2.1 255.255.255.0
 ipv6 address 2001:0:0:2::/64 eui-64
 ipv6 ospf 1 area 0
! Above, the ipv6 address command does the same thing it did for fa0/0, but this
! time with a subnet value of 3.
interface Serial0/1
 ip address 172.31.3.1 255.255.255.0
 ipv6 address 2001:0:0:3::/64 eui-64
 ipv6 ospf 1 area 0
! The ipv6 router ospf 1 command creates an IPv6 OSPF process; the command
! was automatically created when the ipv6 ospf 1 area 0 commands were added
! under the fa0/0 and s0/1 interfaces. These two commands enable OSPF on the
! respective interfaces, and identify the OSPF process id and area number.
```

Example 4-3 *Configuring Dual-Stack IPv4/IPv6 with OSPF (Continued)*

```
ipv6 router ospf 1
 log-adjacency-changes
! Next, the full IPv6 addresses are shown. Note that the interfaces have 2
! addresses; one is the automatically generated link-local address, beginning with
! FE80, used for some protocols on the local link. Note the subnets can be seen
! in the third quartet.
Router1# sh ipv6 int brief
FastEthernet0/0            [up/up]
    FE80::11FF:FE11:1111
    2001::2:0:11FF:FE11:1111
Serial0/1                  [up/up]
    FE80::11FF:FE11:1111
    2001::3:0:11FF:FE11:1111
Virtual-Access1            [up/up]
    unassigned
! Below, the ping from R1 to R2's fa0/0 works.
Router1# ping 2001::4:0:22FF:FE22:2222

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001::3:0:22FF:FE22:2222, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

IPv6 Addressing Summary

Besides global addresses, other styles of IPv6 addresses exist. Table 4-14 lists and briefly describes the different types of addresses.

Table 4-14 *IPv6 Address Type Summary*

KEY POINT

Type of Address	Definition and Purpose
Aggregatable global unicast	Unicast IPV6 address; must be globally unique. Registered unique globally routable address.
Link-local unicast	Required for each IPv6 interface. Used for processes occurring only on the local link; not routable.
Site-local unicast	Intended for use only within a site. Included in the IPv6 definitions in RFC 3513, but deprecated in RFC 3879.
Centrally assigned unique local	Similar to IPv4 private addresses; a range of IPv6 addresses that will not be assigned as public globally routable addresses.
IPv4-compatible unicast	Used mainly for migration to IPv6 when the core is still IPv4. IPv6 addresses are 96 binary 0s, followed by the interface's assigned IPv4 address.

continues

Table 4-14 *IPv6 Address Type Summary (Continued)*

Type of Address	Definition and Purpose
Anycast	The configuration of the same IPv6 unicast address on multiple hosts, with IPv6 routing packets to the closest host. Used with redundant identical services for load balancing.
Multicast	Includes some nonrouted and some routable ranges. Large range (all addresses beginning with FF). Used instead of broadcast addresses.
Broadcast	Nonexistent in IPv6. For instance, ARP is replaced by Neighbor Discovery (ND) messages, which use a reserved IPv6 multicast address.

This short section on IPv6 just scratches the surface of IPv6 features in general, and Cisco routers in particular. The “Further Reading” section near the end of the chapter lists a couple of good references for additional details about IPv6.

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most of the Cisco Press *Exam Certification Guides*, this book does not repeat information presented in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in the Foundation Topics section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 4-15 lists and briefly explains several variations on NAT.

Table 4-15 *Variations on NAT*

Name	Function
Static NAT	Statically correlates the same public IP address for use by the same local host every time. Does not conserve IP addresses.
Dynamic NAT	Pools the available public IP addresses, shared among a group of local hosts, but with only one local host at a time using a public IP address. Does not conserve IP addresses.
Dynamic NAT with overload (PAT)	Like dynamic NAT, but multiple local hosts share a single public IP address by multiplexing using TCP and UDP port numbers. Conserves IP addresses.
NAT for overlapping address	Can be done with any of the first three types. Translates both source and destination addresses, instead of just the source (for packets going from enterprise to the Internet).

Table 4-16 lists the protocols mentioned in this chapter and their respective standards documents.

Table 4-16 *Protocols and Standards for Chapter 4*

Name	Standardized In
IP	RFC 791
Subnetting	RFC 950
NAT	RFC 1631
Private addressing	RFC 1918
CIDR	RFCs 1517–1520
IPv6	RFC 3587, RFC 3513, many others

Table 4-17 lists and describes some of the most commonly used IOS commands related to the topics in this chapter.

Table 4-17 *Command Reference for Chapter 4*

Command	Description
ip address *ip-address mask* [**secondary**]	Interface subcommand to assign an IPv4 address
ipv6 address *ipv6-prefix/prefix-length* **eui-64**	Interface subcommand to assign an IPv6 global unicast address
ipv6 unicast-routing	Globally enables IPv6 in a router
ip nat {**inside** | **outside**}	Interface subcommand; identifies inside or outside part of network
ip nat inside source {**list** {*access-list-number* | *access-list-name*} | **route-map** *name*} {**interface** ***type number*** | **pool** *pool-name*} [**overload**]	Global command that defines the set of inside addresses for which NAT will be performed, and corresponding outside addresses
ip nat inside destination list {*access-list-number* | *name*} **pool** *name*	Global command used with destination NAT
ip nat outside source {**list** {*access-list-number* | *access-list-name*} | **route-map** *name*} **pool** *pool-name* [**add-route**]	Global command used with both destination and dynamic NAT
ip nat pool *name start-ip end-ip* {**netmask** ***netmask*** | **prefix-length** *prefix-length*}[**type rotary**]	Global command to create a pool of addresses for dynamic NAT
show ip nat statistics	Lists counters for packets and for NAT table entries, as well as basic configuration information
show ip nat translations [**verbose**]	Displays the NAT table
clear ip nat translation {***** | [**inside** *global-ip local-ip*] [**outside** *local-ip global-ip*]}	Clears all or some of the dynamic entries in the NAT table, depending on which parameters are used
debug ip nat	Issues log messages describing each packet whose IP address is translated with NAT
show ip interface [*type number*] [**brief**]	Lists information about IPv4 on interfaces
show ipv6 interface [**brief**] [[*interface-type* ***interface-number***] [**prefix**]]	Lists information about IPv6 on interfaces

Figure 4-10 shows the IP header format.

Figure 4-10 *IP Header*

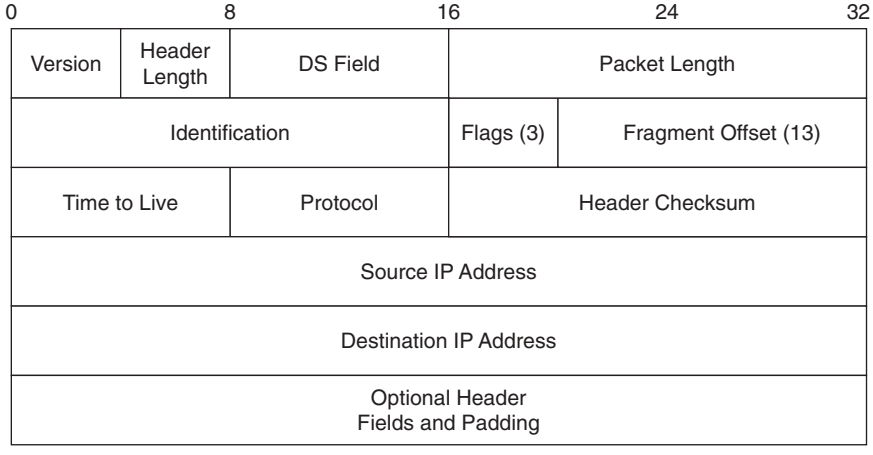


Table 4-18 lists the terms and meanings of the fields inside the IP header.

Table 4-18 *IP Header Fields*

Field	Meaning
Version	Version of the IP protocol. Most networks use IPv4 today, with IPv6 becoming more popular. The header format reflects IPv4.
Header Length	Defines the length of the IP header, including optional fields. Because the length of the IP header must always be a multiple of 4, the IP header length (IHL) is multiplied by 4 to give the actual number of bytes.
DS Field	Differentiated Services Field. This byte was originally called the Type of Service (ToS) byte, but was redefined by RFC 2474 as the DS Field. It is used for marking packets for the purpose of applying different quality of service (QoS) levels to different packets.
Packet Length	Identifies the entire length of the IP packet, including the data.
Identification	Used by the IP packet fragmentation process. If a single packet is fragmented into multiple packets, all fragments of the original packet contain the same identifier, so that the original packet can be reassembled.
Flags	3 bits used by the IP packet fragmentation process.
Fragment Offset	A number set in a fragment of a larger packet that identifies the fragment's location in the larger original packet.
Time to Live (TTL)	A value used to prevent routing loops. Routers decrement this field by 1 each time the packet is forwarded; once it decrements to 0, the packet is discarded.
Protocol	A field that identifies the contents of the data portion of the IP packet. For example, protocol 6 implies a TCP header is the first thing in the IP packet data field.

continues

Table 4-18 *IP Header Fields (Continued)*

Field	Meaning
Header Checksum	A value used to store a frame check sequence (FCS) value, whose purpose is to determine if any bit errors occurred in the IP header (not the data) during transmission.
Source IP Address	The 32-bit IP address of the sender of the packet.
Destination IP Address	The 32-bit IP address of the intended recipient of the packet.
Optional header fields and padding	IP supports additional header fields for future expansion via optional headers. Also, if these optional headers do not use a multiple of 4 bytes, padding bytes are added, comprised of all binary 0s, so that the header is a multiple of 4 bytes in length.

Table 4-19 lists some of the more common IP protocol field values.

Table 4-19 *IP Protocol Field Values*

KEY POINT

Protocol Name	Protocol Number
ICMP	1
TCP	6
UDP	17
EIGRP	88
OSPF	89
PIM	103

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

subnet, prefix, classless IP addressing, classful IP addressing, CIDR, NAT, IPv6, IPv4, subnet broadcast address, subnet number, subnet zero, broadcast subnet, subnet mask, private addresses, aggregatable global unicast address, SLSM, VLSM, Inside Local address, Inside Global address, Outside Local address, Outside Global address, PAT, overloading, quartet, EUI-64, dual stack, global routing prefix, subnet ID, interface ID

Further Reading

All topics in this chapter are covered to varying depth for the CCNP Routing exam. For more details on these topics, refer to *CCNP BSCI Exam Certification Guide*, Third Edition, and *CCNP Self-Study: CCNP BSCI Exam Certification Guide*, Third Edition, listed in the introduction to this book.

For more information on IPv6, refer to http://www.cisco.com/en/US/partner/tech/tk872/tsd_technology_support_protocol_home.html. This website requires a CCO username and password.

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- General Networking Theory
  - — Standards
- IP
  - — Services

In addition, this chapter covers information related to the following specific CCIE Routing and Switching exam topics:

- ICMP
- DHCP
- NTP
- HSRP

CHAPTER 5

IP Services

IP relies on several protocols to perform a variety of tasks related to the process of routing packets. This chapter provides a reference for the most popular of these protocols.

"Do I Know This Already?" Quiz

Table 5-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 5-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
ICMP	1–3	
ARP, Proxy ARP, Reverse ARP, BOOTP, and DHCP	4–6	
HSRP, VRRP, and GLBP	7	
Network Time Protocol	8	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. What ICMP message is used by a router when the router tries to forward a packet to a host on an attached subnet, but the host does not respond to ARP requests?

- **a.** Echo Request
- **b.** Time Exceeded
- **c.** Source Quench
- **d.** Destination Unreachable
- **e.** Host Unreachable
- **f.** Port Unreachable

2. What ICMP message does the Cisco IOS **traceroute** command rely upon?
   a. Echo Request
   b. Source Quench
   c. Time Exceeded
   d. Destination Unreachable
   e. Host Unreachable

3. What ICMP message implies that a packet arrived at a host, but there was no application listening on the TCP or UDP port number listed as the destination port in the packet?
   a. Echo Request
   b. Source Quench
   c. Destination Unreachable
   d. Host Unreachable
   e. Port Unreachable

4. Two hosts, named PC1 and PC2, sit on subnet 172.16.1.0/24, along with router R1. A web server sits on subnet 172.16.2.0/24, which is connected to another interface of R1. At some point, both PC1 and PC2 send an ARP request before they successfully send packets to the web server. With PC1, R1 makes a normal ARP reply, but for PC2, R1 uses a proxy ARP reply. Which two of the following answers could be true given the stated behavior in this network?
   a. PC2 set the proxy flag in the ARP request.
   b. PC2 encapsulated the ARP request inside an IP packet.
   c. PC2's ARP broadcast implied that PC2 was looking for the web server's MAC address.
   d. PC2 has a subnet mask of 255.255.0.0.
   e. R1's proxy ARP reply contains the web server's MAC address.

5. Host PC3 is using DHCP to discover its IP address. Only one router attaches to PC3's subnet, using its fa 0/0 interface, with an **ip helper-address 10.5.5.5** command on that same interface. That same router interface has an **ip address 10.4.5.6 255.255.252.0** command configured as well. Which of the following are true about PC3's DHCP request?
   a. The destination IP address of the DHCP request packet is set to 10.5.5.5 by the router.
   b. The DHCP request packet's source IP address is unchanged by the router.

c. The DHCP request is encapsulated inside a new IP packet, with source IP address 10.4.5.6, and destination 10.5.5.5.

d. The DHCP request's source IP address is changed to 10.4.5.255.

e. The DHCP request's source IP address is changed to 10.4.7.255.

6. Which of the following statements are true about BOOTP, but not true about RARP?

a. The client can be assigned a different IP address on different occasions, because the server can allocate a pool of IP addresses for allocation to a set of clients.

b. The server can be on a different subnet from the client.

c. The client's MAC address must be configured on the server, with a one-to-one mapping to the IP address to be assigned to the client with that MAC address.

d. The client can discover its IP address, subnet mask, and default gateway IP address.

7. R1 is HSRP active for virtual IP address 172.16.1.1, with HSRP priority set to 115. R1 is tracking three separate interfaces. An engineer configures the same HSRP group on R2, also connected to the same subnet, only using the **standby 1 ip 172.16.1.1** command, and no other HSRP-related commands. Which of the following would cause R2 to take over as HSRP active?

a. R1 experiences failures on tracked interfaces, totaling 16 or more lost points.

b. R1 experiences failures on tracked interfaces, totaling 15 or more lost points.

c. R2 could configure a priority of 116 or greater.

d. R1's fa0/0 interface fails.

e. R2 would take over immediately.

8. Which of the following NTP modes in a Cisco router requires a predefinition of the IP address of an NTP server?

a. Server mode

b. Static client mode

c. Broadcast client mode

d. Symmetric active mode

Foundation Topics

ICMP

The Internetwork Control Message Protocol (ICMP) allows for testing and troubleshooting of the TCP/IP internetwork layer by defining messages that can be used to determine whether the network can currently deliver packets. In fact, ICMP is a required component of every IP implementation, as described in the following brief excerpt from RFC 792:

> Occasionally a gateway or destination host will communicate with a source host, for example, to report an error in datagram processing. For such purposes this protocol, the Internet Control Message Protocol (ICMP), is used. ICMP uses the basic support of IP as if it were a higher level protocol; however, ICMP is actually an integral part of IP, and must be implemented by every IP module.

While "ping packets," also known as *ICMP echo requests* and *ICMP echo replies*, are well known, ICMP includes a variety of different message types for varying purposes. Table 5-2 lists the most important of these message types.

Table 5-2 *ICMP Message Types*

KEY POINT

Message	Purpose
Destination Unreachable	Tells the source host that there is a problem delivering a packet.
Time Exceeded	The time it takes a packet to be delivered has become too long; the packet has been discarded.
Source Quench	The source is sending data faster than it can be forwarded; this message requests that the sender slow down.
Redirect	Used by a router to tell a host to use a different, better router when sending packets to that same IP address in the future.
Echo	Used by the **ping** command to verify connectivity.
Address Mask Request/Reply	Used to inquire about and learn the correct subnet mask to be used.
Router Advertisement and Selection	Used to allow hosts to dynamically learn the IP addresses of the routers attached to the subnet.

Each ICMP message contains a Type field and a Code field. The Type field implies the message types from Table 5-2, with the Code field implying a subtype. For instance, many people refer to the messages generated by the **ping** command as ICMP Echo Request and ICMP Echo Reply, but in reality, the two messages have the same message type (Echo) and different codes (Request and Reply). The remaining coverage of ICMP explains some details behind several of the types and codes for ICMP messages.

ICMP Unreachable

When a device realizes that a packet cannot be delivered to its destination, the device sends an ICMP Unreachable message. To help determine the root cause of why the packet cannot be delivered, the ICMP Unreachable message includes one of five code field values to convey the reason for the failure. For instance, in Figure 5-1, assume that Fred is trying to connect to the web server, called Web. Table 5-3, following the figure, lists the key ICMP Unreachable message codes, along with an example set of circumstances from Figure 5-1 that would result in each Unreachable code.

Figure 5-1 *Sample Network for ICMP Unreachable Examples*

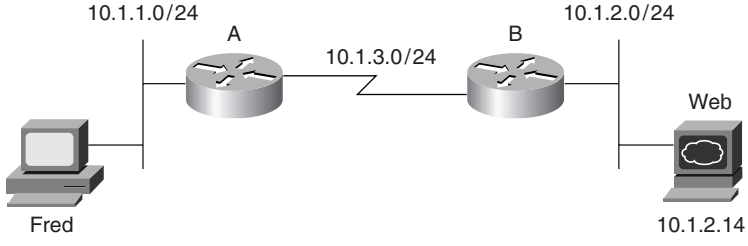


Table 5-3 *ICMP Unreachable Codes*

KEY POINT

Unreachable Code	Meaning	Example
Network Unreachable	A router has no route matching the destination IP address of the packet.	Router A does not have a route to 10.1.2.0/24; sends an Unreachable to Fred.
Host Unreachable	The packet arrives at the last router, but the host does not reply to ARP requests.	Host Web is turned off. Router B sends Fred a Host Unreachable message.
Can't Fragment	A packet is longer than the outgoing link's MTU, but the packet has the Don't Fragment (DF) bit set.	Router A defaults to a 1500-byte MTU, and the packet is 1600 bytes with the DF bit set.

continues

Table 5-3 *ICMP Unreachable Codes (Continued)*

Unreachable Code	Meaning	Example
Protocol Unreachable	The packet reached the destination host, but the host is not running that transport layer protocol.	Somehow, host Web does not have its TCP software running. Host Web sends the Unreachable message to Fred.
Port Unreachable	The packet is delivered to the destination host, but there is no process listening on the destination port.	Host Web is working, but the web server software is not currently running. Host Web sends the Unreachable message to Fred.

Time Exceeded ICMP Message

The ICMP Time Exceeded message notifies a host when a packet it sent has been discarded because it was "out of time." Packets are not actually timed, but to prevent packets from being forwarded forever when there is a routing loop, each IP packet header includes a Time to Live (TTL) field. Routers decrement TTL by 1 every time they forward a packet; if a router happens to decrement TTL of a packet to 0, the router discards the packet and sends an ICMP Time Exceeded message to the sender of the original packet.

The Cisco IOS **trace** command uses the Time Exceeded message and the IP TTL field to its advantage. The **trace** command sends three packets, each with TTL set to 1, resulting in an ICMP Time Exceeded message being returned by the first router in the route (because that router decrements TTL to 0, discards the packet, and returns a Time Exceeded message). The **trace** command then sends another three packets with a TTL of 2, then another set with a TTL of 3, and so on, until it gets a response from the host. Example 5-1 shows this **trace** command on RouterA from Figure 5-1, with **debug** messages from RouterB listing the Time Exceeded messages sent from RouterB back to RouterA.

Example 5-1 *ICMP* **debug** *on RouterB, When Running* **trace** *Command on RouterA*

```
RouterA# trace 10.1.2.14

Type escape sequence to abort.
Tracing the route to 10.1.2.14

  1 10.1.3.253 8 msec 4 msec 4 msec
  2 10.1.2.14 12 msec 8 msec 4 msec
RouterA#
```

```
RouterB#
ICMP: time exceeded (time to live) sent to 10.1.3.251 (dest was 10.1.2.14)
ICMP: time exceeded (time to live) sent to 10.1.3.251 (dest was 10.1.2.14)
ICMP: time exceeded (time to live) sent to 10.1.3.251 (dest was 10.1.2.14)
```

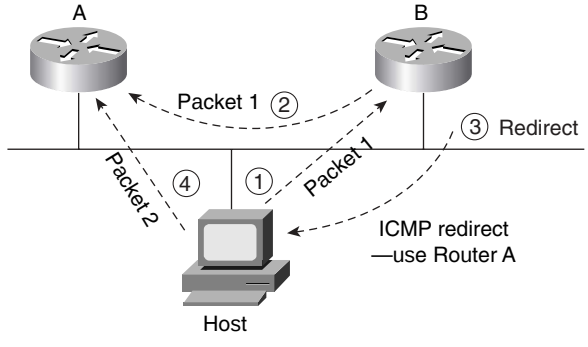
ICMP Redirect

ICMP Redirect messages allow a host's default gateway router to inform local hosts of a better router to use to reach certain destinations. To do so, a router sends an ICMP Redirect to the host to tell it the IP address of the better alternative router. For example, in Figure 5-2, the PC uses RouterB as its default router, but RouterA's route to subnet 10.1.4.0/24 is a better route. Following the steps in Figure 5-2:

1. The PC sends a packet, destined for subnet 10.1.4.0/24, to RouterB.
2. RouterB forwards the packet based on its own routing table.
3. RouterB sends the ICMP Redirect message to the PC, telling it to forward future packets destined for 10.1.4.0/24 to RouterA instead.
4. The PC sends future packets to that host directly to RouterA.

Figure 5-2 *Example of an ICMP Redirect*

KEY POINT



Ironically, the host can ignore the redirect and keep sending the packets to RouterB.

ARP, Proxy ARP, Reverse ARP, BOOTP, and DHCP

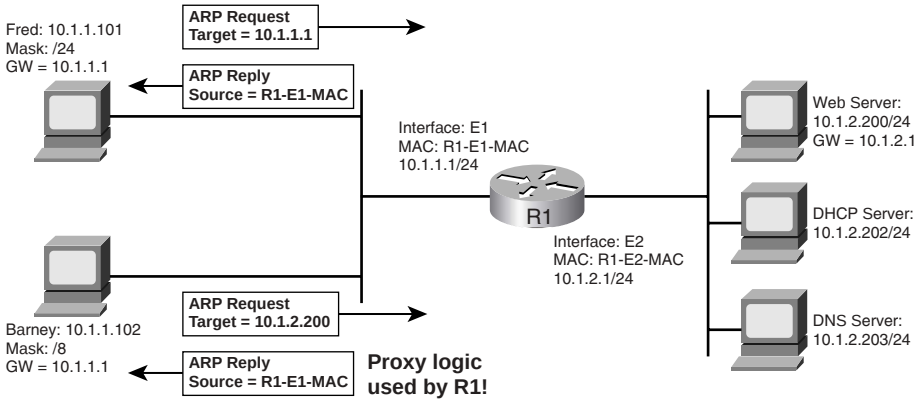
The heading for this section may seem like a laundry list of a lot of different protocols. However, these five protocols do have one central theme, namely that they help a host learn information so that it can successfully send and receive IP packets. Specifically, ARP and proxy ARP define methods for a host to learn another host's MAC address, whereas the core functions of RARP, BOOTP, and DHCP define how a host can discover its own IP address, plus additional related information.

ARP and Proxy ARP

You would imagine that anyone getting this far in their CCIE study would already have a solid understanding of the Address Resolution Protocol (ARP, RFC 826). However, proxy ARP

(RFC 1027) is often ignored, in part because of its lack of use today. To see how they both work, Figure 5-3 shows an example of each, with Fred and Barney both trying to reach the web server at IP address 10.1.2.200.

Figure 5-3 *Comparing ARP and Proxy ARP*



Fred follows a normal ARP process, broadcasting an ARP request, with R1's E1 IP address as the target. The ARP message has a *Target* field of all 0s for the MAC address that needs to be learned, and a target IP address of the IP address whose MAC address it is searching, namely 10.1.1.1 in this case. The ARP reply lists the MAC address associated with the IP address, in this case, the MAC address of R1's E1 interface.

NOTE The ARP message itself does not include an IP header, although it does have destination and source IP addresses in the same relative position as an IP header. The ARP request lists an IP destination of 255.255.255.255. The ARP Ethernet protocol type is 0x0806, whereas IP packets have an Ethernet protocol type of 0x0800.

Proxy ARP uses the exact same ARP message as ARP, but the ARP request is actually requesting a MAC address that is not on the local subnet. Because the ARP request is broadcast on the local subnet, it will not be heard by the target host—so if a router can route packets to that target host, the router issues a proxy ARP reply on behalf of that target.

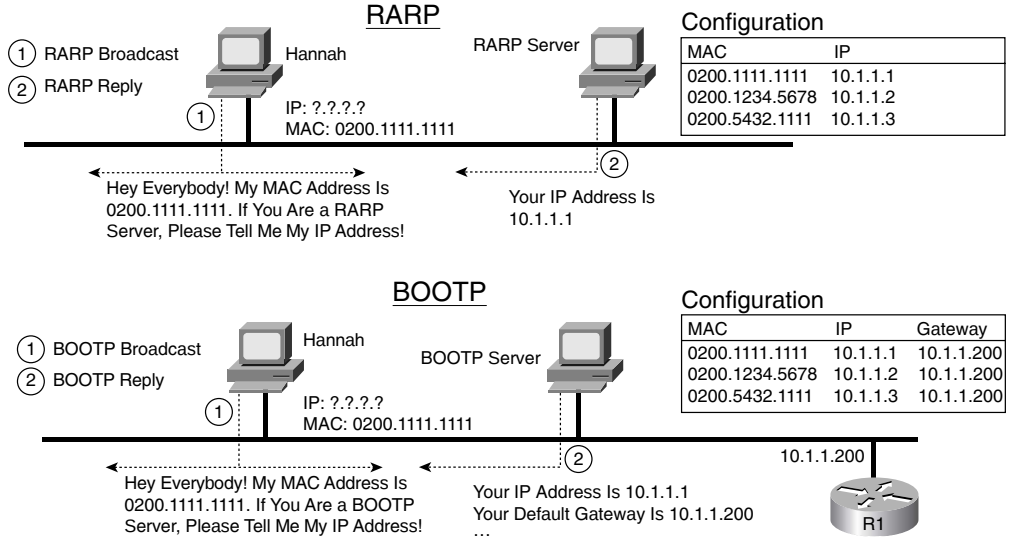
For instance, Barney places the web server's IP address (10.1.2.200) in the target field, because Barney thinks that he is on the same subnet as the web server due to Barney's mask of 255.0.0.0. The ARP request is a LAN broadcast, so R1, being a well-behaved router, does not forward the ARP broadcast. However, knowing that the ARP request will never get to the subnet where 10.1.2.200 resides, R1 saves the day by replying to the ARP on behalf of the web server. R1 takes the web server's place in the ARP process, hence the name *proxy* ARP. Also, note that R1's ARP reply contains R1's E1 MAC address, so that Barney will forward frames to R1 when Barney wants to send a packet to the web server.

Before the advent of DHCP, many networks relied on proxy ARP, configuring hosts to use the default masks in their respective networks. Regardless of whether the proxy version is used, the end result is that the host learns a router's MAC address to forward packets to another subnet.

RARP, BOOTP, and DHCP

The ARP and proxy ARP processes both occur after a host knows its IP address and subnet mask. RARP, BOOTP, and DHCP represent the evolution of protocols defined to help a host dynamically learn its IP address. All three protocols require the client host to send a broadcast to begin discovery, and all three rely on a server to hear the request and supply an IP address to the client. Figure 5-4 shows the basic processes with RARP and BOOTP.

Figure 5-4 *RARP and BOOTP—Basic Processes*



KEY POINT

A RARP request is a host's attempt to find its own IP address. So RARP uses the same old ARP message, but the ARP request lists a MAC address target of its own MAC address and a target IP address of 0.0.0.0. A preconfigured RARP server, which must be on the same subnet as the client, receives the request and performs a table lookup in its configuration. If that target MAC address listed in the ARP request is configured on the RARP server, the RARP server sends an ARP reply, after entering the configured IP address in the Source IP address field.

KEY POINT

BOOTP was defined in part to improve IP address assignment features of RARP. BOOTP uses a completely different set of messages, defined by RFC 951, with the commands encapsulated inside an IP and UDP header. With the correct router configuration, a router can forward the BOOTP packets to other subnets—allowing the deployment of a centrally located BOOTP server. Also, BOOTP supports the assignment of many other tidbits of information, including the subnet mask, default gateway, DNS addresses, and its namesake, the IP address of a boot (or image)

server. However, BOOTP does not solve the configuration burden of RARP, still requiring that the server be preconfigured with the MAC addresses and IP addresses of each client.

DHCP represents the next step in the evolution of dynamic IP address assignment. Building on the format of BOOTP protocols, DHCP focuses on dynamically assigning a variety of information and provides flexible messaging to allow for future changes, without requiring predefinition of MAC addresses for each client. DHCP also includes temporary leasing of IP addresses, enabling address reclamation, pooling of IP addresses, and, recently, dynamic registration of client DNS fully qualified domain names (FQDNs). (See http://www.ietf.org/internet-drafts/draft-ietf-dhc-fqdn-option-11.txt for more information on FQDN registration.)

DHCP servers typically reside in a centralized location, with remote routers forwarding the LAN-broadcast DHCP requests to the DHCP server by changing the request's destination address to match the DHCP server. This feature is called DHCP *relay agent*. For instance, in Figure 5-3, if Fred and Barney were to use DHCP, with the DHCP server at 10.1.2.202, R1 would change Fred's DHCP request from a source and destination of 255.255.255.255, to a source of 10.1.1.255 (directed broadcast of Fred's subnet) and destination of 10.1.2.202. The DHCP request would then be routed to the DHCP server, and the DHCP response would be forwarded to destination 10.1.1.255. The router would then broadcast the DHCP response back onto that subnet, as the destination address is Fred's subnet's broadcast address. The only configuration requirement on the router is an **ip helper-address 10.1.2.202** interface subcommand on its E1 interface.

Alternatively, R1 could be configured as a DHCP server—a feature that is not popular in production networks, but is certainly fair game for the CCIE written and lab exams. Example 5-2 shows R1's configuration for a DHCP relay agent, as well as an alternative for R1 to provide DNS services for subnet 10.1.1.0/24.

Example 5-2 *DHCP Configuration Options—R1, Figure 5-3*

KEY POINT

```
! UDP broadcasts coming in E0 will be forwarded as unicasts to 10.1.2.202.
! The source IP will be changed to 10.1.1.255, so that the reply packets will be
! broadcast back out E0.
interface Ethernet1
 ip address 10.1.1.1 255.255.255.0
 ip helper-address 10.1.2.202
! Below, an alternative configuration, with R1 as the DHCP server. R1 assigns IP
! addresses other than the excluded first 20 IP addresses in the subnet, and informs the
! clients of their IP addresses, mask, DNS, and default router. Leases are for 0 days,
! 0 hours, and 20 minutes.
ip dhcp excluded-address 10.1.1.0 10.1.1.20
!
ip dhcp pool subnet1
   network 10.1.1.0 255.255.255.0
   dns-server 10.1.2.203
   default-router 10.1.1.1
   lease 0 0 20
```

Table 5-4 summarizes some of the key comparison points with RARP, BOOTP, and DHCP.

Table 5-4 *Comparing RARP, BOOTP, and DHCP*

KEY POINT

Feature	RARP	BOOTP	DHCP
Relies on server to allocate IP addresses	Yes	Yes	Yes
Encapsulates messages inside IP and UDP, so they can be forwarded to a remote server	No	Yes	Yes
Client can discover its own mask, gateway, DNS, and download server	No	Yes	Yes
Dynamic address assignment from a pool of IP addresses, without requiring knowledge of client MACs	No	No	Yes
Allows temporary lease of IP address	No	No	Yes
Includes extensions for registering client's FQDN with a DNS	No	No	Yes

HSRP, VRRP, and GLBP

IP hosts can use several methods of deciding which default router or default gateway to use—DHCP, BOOTP, ICMP Router Discovery Protocol (IRDP), manual configuration, or even by running a routing protocol (although having hosts run a routing protocol is not common today). The most typical methods—using DHCP or manual configuration—result in the host knowing a single IP address of its default gateway. Hot Standby Router Protocol (HSRP), Virtual Router Redundancy Protocol (VRRP), and Gateway Load Balancing Protocol (GLBP) represent a chronological list of some of the best tools for overcoming the issues related to a host knowing a single IP address as its path to get outside the subnet.

HSRP allows multiple routers to share a virtual IP and MAC address so that the end-user hosts do not realize when a failure occurs. Some of the key HSRP features are as follows:

KEY POINT

- Virtual IP address and virtual MAC active on the Master router
- Standby routers listen for Hellos from the Active router, defaulting to a 3-second hello interval and 10-second dead interval
- Highest priority (IOS default 100, range 1–255) determines the active router, with pre-emption disabled by default
- Supports tracking, whereby a router's priority is decreased when a tracked interface fails
- Up to 255 HSRP groups per interface, enabling an administrative form of load balancing

- Virtual MAC of 0000.0C07.AC*xx*, where *xx* is the hex HSRP group
- Virtual IP address must be in the same subnet as the routers' interfaces on the same LAN
- Virtual IP address must be different from any of routers' individual interface IP addresses

Example 5-3 shows a typical HSRP configuration, with two groups configured. Routers R1 and R2 are attached to the same subnet, 10.1.1.0/24, both with WAN links (S0/0.1) connecting them to the rest of an enterprise network. The example contains the details and explanation of the configuration.

Example 5-3 *HSRP Configuration*

KEY POINT

```
! First, on Router R1, two HSRP groups are configured. R1 has a higher priority
! in group 21, with R2 having a higher priority in group 22. R1 is set to preempt
! in group 21, as well as to track interface s0/0.1 for both groups.
interface FastEthernet0/0
 ip address 10.1.1.2 255.255.255.0
 standby 21 ip 10.1.1.21
 standby 21 priority 105
 standby 21 preempt
 standby 21 track Serial0/0.1
 standby 22 ip 10.1.1.22
 standby 22 track Serial0/0.1
```

```
! Next, R2 is configured with a higher priority for HSRP group 22, and with
! HSRP tracking enabled in both groups. The tracking "decrement" used by R2,
! when S0/0.1 fails, is set to 9 (instead of the default of 10).
interface FastEthernet0/0
 ip address 10.1.1.1 255.255.255.0
 standby 21 ip 10.1.1.21
 standby 21 track Serial0/0.1
 standby 22 ip 10.1.1.22
 standby 22 priority 105
 standby 22 track Serial0/0.1 9
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! On R1 below, for group 21, the output shows that R1 is active, with R2
! (10.1.1.2) as standby.
! R1 is tracking s0/0.1, with a default "decrement" of 10, meaning that the
! configured priority of 105 will be decremented by 10 if s0/0.1 fails.
Router1# sh standby group 21
FastEthernet0/0 - Group 21
  State is Active
    2 state changes, last state change 00:00:45
  Virtual IP address is 10.1.1.21
  Active virtual MAC address is 0000.0c07.ac15
    Local virtual MAC address is 0000.0c07.ac15 (default)
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 2.900 secs
  Preemption enabled
```

Example 5-3 *HSRP Configuration (Continued)*

```
  Active router is local
  Standby router is 10.1.1.2, priority 100 (expires in 7.897 sec)
  Priority 105 (configured 105)
    Track interface Serial0/0.1 state Up decrement 10
  IP redundancy name is "hsrp-Fa0/0-21" (default)
! NOT SHOWN—R1 shuts down S0.0.1, lowering its priority in group 21 by 10.
! The debug below shows the reduced priority value. However, R2 does not become
! active, because R2's configuration did not include a standby 21 preempt command.
Router1# debug standby
*Mar  1 00:24:04.122: HSRP: Fa0/0 Grp 21 Hello  out 10.1.1.1 Active  pri 95 vIP 10.1.1.21
```

HSRP is Cisco proprietary, has been out a long time, and is widely popular. VRRP (RFC 3768) provides a standardized protocol to perform almost the exact same function. The Cisco VRRP implementation has the same goals in mind as HSRP, but with these differences:

KEY POINT

- VRRP uses a multicast virtual MAC address (0000.5E00.01*xx*, where *xx* is the hex VRRP group number).
- VRRP does not support interface tracking (at least in the 12.2T/12.3 mainline releases).
- In Cisco IOS Software 12.2T/12.3 mainline releases, VRRP defaults to use pre-emption, but HSRP defaults to not use pre-emption.
- The VRRP term *Master* means the same thing as the HSRP term *Active*.

GLBP is a newer Cisco-proprietary tool that adds load-balancing features in addition to gateway-redundancy features. Hosts still point to a default gateway IP address, but GLBP causes different hosts to send their traffic to one of up to four routers in a GLBP group. To do so, the GLBP Active Virtual Gateway (AVG) assigns each router in the group a unique virtual MAC address, following the format 0007.B400.*xxyy*, where *xx* is the GLBP group number, and *yy* is a different number for each router (01, 02, 03, or 04). When a client ARPs for the (virtual) IP address of its default gateway, the GLBP AVG replies with one of the four possible virtual MACs. By replying to ARP requests with different virtual MACs, the hosts in that subnet will in effect balance the traffic across the routers, rather than send all traffic to the one active router.

Network Time Protocol

NTP Version 3 (RFC 1305) allows IP hosts to synchronize their time-of-day clocks with a common source clock. For instance, routers and switches can synchronize their clocks to make event correlation from an SNMP management station more meaningful, by ensuring that any events and traps have accurate time stamps.

By design, most routers and switches use NTP *client mode*, adjusting their clocks based on the time as known by an NTP server. NTP defines the messages that flow between client and server,

and the algorithms a client uses to adjust its clock. Routers and switches can also be configured as NTP servers, as well as using NTP *symmetric active mode*—a mode in which the router or switch mutually synchronizes with another NTP host.

NTP servers may reference other NTP servers to obtain a more accurate clock source as defined by the *stratum level* of the ultimate source clock. For instance, atomic clocks and Global Positioning System (GPS) satellite transmissions provide a source of stratum 1 (lowest/best possible stratum level). For an enterprise network, the routers and switches can refer to a low-stratum NTP source on the Internet, or purpose-built rack-mounted NTP servers, with built-in GPS capabilities, can be deployed.

Example 5-4 shows a sample NTP configuration on four routers, all sharing the same 10.1.1.0/24 Ethernet subnet. Router R1 will be configured as an NTP server. R2 acts as an *NTP static client* by virtue of the static configuration referencing R1's IP address. R3 acts as an *NTP broadcast client* by listening for R1's NTP broadcasts on the Ethernet. Finally, R4 acts in NTP symmetric active mode, configured with the **ntp peer** command.

Example 5-4 *NTP Configuration*

KEY POINT

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! First, R1's configuration, the ntp broadcast command under interface fa0/0
! causes NTP to broadcast NTP updates on that interface. The first three of the
! four global NTP commands configure authentication; these commands are identical
! on all the routers.
R1# show running-config
interface FastEthernet0/0
 ntp broadcast
!
ntp authentication-key 1 md5 1514190900 7
ntp authenticate
ntp trusted-key 1
ntp master 7
! Below, the "127.127.7.1" notation implies that this router is the NTP clock
! source. The clock is synchronized, with stratum level 7, as configured on the
! ntp master 7 command above.
R1# show ntp associations

      address         ref clock     st  when  poll reach  delay  offset    disp
*~127.127.7.1      127.127.7.1       6    22    64  377     0.0    0.00     0.0
 * master (synced), # master (unsynced), + selected, - candidate, ~ configured
R1# show ntp status
Clock is synchronized, stratum 7, 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 C54483CC.E26EE853 (13:49:00.884 UTC Tue Nov 16 2004)
clock offset is 0.0000 msec, root delay is 0.00 msec
root dispersion is 0.02 msec, peer dispersion is 0.02 msec
```

Example 5-4 *NTP Configuration (Continued)*

```
! R2 is configured below as an NTP static client. Note that the ntp clock-period
! command is automatically generated as part of the synchronization process, and
! should not be added to the configuration manually.
R2# show run | begin ntp
ntp authentication-key 1 md5 1514190900 7
ntp authenticate
ntp trusted-key 1
ntp clock-period 17208144
ntp server 10.1.1.1
end
```

```
! Next, R3 has been configured as an NTP broadcast client. The ntp broadcast client
! command on R3 tells it to listen for the broadcasts from R1. This configuration
! relies on the ntp broadcast subcommand on R1's Fa0/0 interface, as shown at the
! beginning of this example.
R3# show run
interface Ethernet0/0
 ntp broadcast client
```

```
! R4's configuration is listed, with the ntp peer
! command implying the use of symmetric active mode.
R4# show run | beg ntp
ntp authentication-key 1 md5 0002010300 7
ntp authenticate
ntp trusted-key 1
ntp clock-period 17208233
ntp peer 10.1.1.1
```

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most of the Cisco Press *Exam Certification Guides*, this book does not repeat information presented in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in the "Foundation Topics" section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 5-5 lists the protocols mentioned in this chapter and their respective standards documents.

Table 5-5 *Protocols and Standards for Chapter 5*

Name	Standardized In
ICMP	RFCs 792 and 950
ARP	RFC 826
Proxy ARP	RFC 1027
RARP	RFC 903
BOOTP	RFC 951
DHCP	RFC 2131
DHCP FQDN option	Internet-Draft
HSRP	Cisco proprietary
VRRP	RFC 3768
GLBP	Cisco proprietary
CDP	Cisco proprietary
NTP	RFC 1305

Table 5-6 lists some of the most popular Cisco IOS commands related to the topics in this chapter.

Table 5-6 *Command Reference for Chapter 5*

Command	Description
ip dhcp pool *name*	Creates DHCP pool
default-router *address* [*address2...address8*]	DHCP pool subcommand to list the gateways

Table 5-6 *Command Reference for Chapter 5 (Continued)*

Command	Description
dns-server *address* [*address2...address8*]	DHCP pool subcommand to list DNS servers
lease {*days* [*hours*][*minutes*] | **infinite**}	DHCP pool subcommand to define the lease length
network *network-number* [*mask* | *prefix-length*]	DHCP pool subcommand to define IP addresses that can be assigned
ip dhcp excluded-address *low-address high-address*]	DHCP pool subcommand to disallow these addresses from being assigned
host *address* [*mask* | *prefix-length*]	DHCP pool subcommand, used with *hardware-address* or *client-identifier*, to predefine a single host's IP address
hardware-address *hardware-address type*	DHCP pool subcommand to define MAC address; works with **host** command
show ip dhcp binding [*ip-address*]	Lists addresses allocated by DHCP
show ip dhcp server statistics	Lists stats for DHCP server operations
standby [*group-number*] **ip** [*ip-address* [**secondary**]]	Interface subcommand to enable an HSRP group and define the virtual IP address
standby [*group-number*] **preempt** [**delay** {**minimum** *delay* | **reload** *delay* | **sync** *delay*}]	Interface subcommand to enable pre-emption and set delay timers
standby [*group-number*] **priority** *priority*	Interface subcommand to set the HSRP group priority for this router
standby [*group-number*] **timers** [**msec**] *hellotime* [**msec**] *holdtime*	Interface subcommand to set HSRP group timers
standby [*group-number*] **track** *interface-type interface-number* [*interface-priority*]	Interface subcommand to list HSRP tracked interfaces and define priority values deducted when the interface fails
show standby [*type number* [*group*]] [**active** | **init** | **listen** | **standby**] [**brief** | **all**]	Lists HSRP statistics
ntp peer *ip-address* [**version** *number*] [**key** *keyid*] [**source** *interface*] [**prefer**]	Global command to enable symmetric active mode NTP
ntp server *ip-address* [**version** *number*] [**key** *keyid*] [**source** *interface*] [**prefer**]	Global command to enable static client mode NTP
ntp broadcast [**version** *number*]	Interface subcommand on an NTP server to cause NTP broadcasts on the interface

continues

Table 5-6 *Command Reference for Chapter 5 (Continued)*

Command	Description
ntp broadcast client	Interface subcommand on an NTP client to cause it to listen for NTP broadcasts
ntp master [*stratum*]	Global command to enable NTP server
show ntp associations	Lists associations with other NTP servers and clients
show ntp status	Displays synchronization status, stratum level, and other basic information

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

HSRP, VRRP, GLBP, ARP, RARP, proxy ARP, BOOTP, DHCP, NTP symmetric active mode, NTP server mode, NTP client mode, NTP

Refer to the CD-based glossary to check your answers.

Further Reading

More information about several of the topics in this chapter can be easily found in a large number of books and online documentation. The RFCs listed in Table 5-5 of the "Foundation Summary" section also provide a great deal of background information for this chapter. Here are a few references for more information about some of the less popular topics covered in this chapter:

- **Proxy ARP**—http://www.cisco.com/en/US/tech/tk648/tk361/technologies_tech_note09186a0080094adb.shtml
- **GLBP**—http://www.cisco.com/en/US/partner/products/sw/iosswrel/ps1839/products_white_paper09186a00801541c8.shtml
- **VRRP**—http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120limit/120st/120st18/st_vrrpx.htm

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- General Networking Theory
  - — Standards
  - — Protocol Mechanics
- IP
  - — Applications
  - — Transport
  - — Network Management

In addition, this chapter covers information related to the following specific CCIE Routing and Switching exam topics:

- TCP
- UDP
- TCP/IP applications
- SNMP management

CHAPTER 6

TCP/IP Transport and Application Services

This last of the three chapters in this part of the book covers a wide variety of topics in TCP/IP's transport and application layers. Because most of the basics related to these topics should be familiar to the typical CCIE Routing and Switching candidate, the chapter briefly hits the highlights, summarizes key facts about the protocols, and expands upon particular features in some cases.

"Do I Know This Already?" Quiz

Table 6-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 6-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
TCP and UDP	1–3	
TCP/IP Applications	4–5	
Network Management and SNMP	6–7	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. Which of the following items are features of both TCP and UDP?

- **a.** Identification of application processes by using Port numbers
- **b.** Error recovery
- **c.** Flow control
- **d.** Supplying a header checksum

2. Which of the following TCP code bits are not used for connection initiation or termination?
   a. PSH
   b. RST
   c. SYN
   d. FIN
   e. ACK
   f. URG
3. Which of the following statements is always true about TCP flow control and error recovery?
   a. A TCP sender considers a connection's dynamic window size to be based on the Window Size field in segments received on that connection.
   b. A TCP sender can use a window size based on the average between two settings: the advertised window and the congestion window.
   c. After packet loss, a TCP sender both resends the lost TCP segment and sets the URG code bit.
   d. The TCP Acknowledgement header field contains the number associated with the last byte of data that was acknowledged.
4. Which of the following statements are true regarding a protocol, its use of TCP or UDP, and its well-known port number(s)?
   a. HTTP uses TCP well-known port 80.
   b. SMTP uses UDP well-known port 53.
   c. LDAP uses TCP well-known port 443.
   d. SNMP uses TCP well-known port 162.
   e. POP3 uses TCP well-known port 110.
5. Which of the following are true regarding active and passive mode FTP?
   a. Active mode causes the FTP server to initiate the TCP connection to the FTP client.
   b. Active mode causes the FTP server to allocate and begin listening on a dynamic unused port, and notify the FTP client of that port number so the client can connect to that port.
   c. Passive mode causes the FTP server to allocate and begin listening on a dynamic unused port, and notify the FTP client of that port number so that the client can connect to that port.
   d. Passive mode does not require any port numbers to be passed in either direction between the client and server; the client simply creates a data connection to the server's well-known port 20.

6. Which of the following are true about SNMP security?

   a. SNMP Version 1 calls for the use of community strings that are passed as clear text.

   b. SNMP Version 2c calls for the use of community strings that are passed as MD5 message digests generated with private keys.

   c. SNMP Version 3 allows for authentication using MD5 message digests generated with private keys.

   d. SNMP Version 3 authentication also requires concurrent use of encryption, typically done with DES.

7. Which of the following statements are true regarding features of SNMP based on the SNMP version?

   a. SNMP Version 2 added the GetNext protocol message to SNMP.

   b. SNMP Version 3 added the Inform protocol message to SNMP.

   c. SNMP Version 2 added the Inform protocol message to SNMP.

   d. SNMP Version 3 expanded the SNMP Response protocol message so that it must be used by managers in response to Traps sent by agents.

   e. SNMP Version 3 enhanced SNMP Version 2 security features, but not other features.

Foundation Topics

TCP and UDP

The Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) are the two most popular transport layer protocols in the TCP/IP suite. UDP consumes fewer bytes of overhead and requires less processing. TCP requires more overhead and more processing, but offers many useful functions, including error recovery. Table 6-2 summarizes the key comparisons between the two protocols.

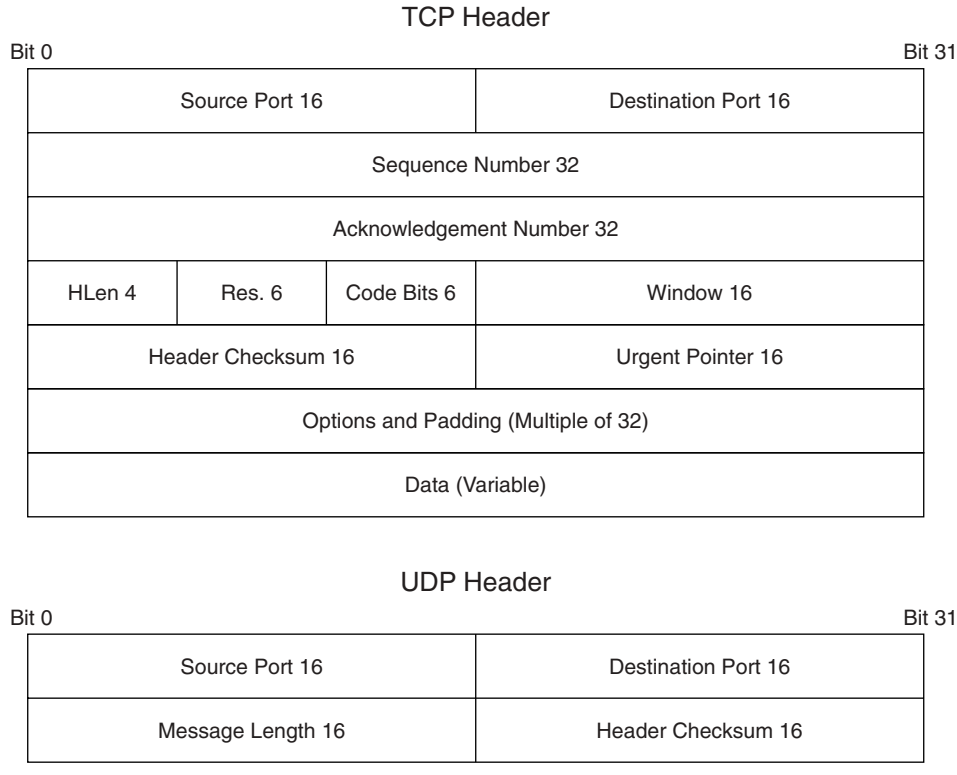
Table 6-2 *TCP and UDP Functional Comparison*

KEY POINT

Function	TCP	UDP
Multiplexing using ports	Yes; port numbers uniquely identify the processes (socket programs) on the computers sending and receiving the data.	Same as TCP.
Ordered data transfer	Yes; TCP reorders any data that is received out of order.	No; UDP has no concept of ordering of data.
Reliable transfer	Yes; TCP acknowledges data, resending lost segments, using the Sequence and Acknowledgment fields in the TCP header.	Not supported by UDP.
Flow control	Yes; TCP uses a granted (advertised) window, as well as a congestion window, to control each TCP sender's dynamic window.	Not supported by UDP.
Connections	Yes; the three-message TCP connection setup process initializes the port numbers.	UDP is connectionless.
IP protocol type	6	17
Base RFC	793	768

The TCP and UDP headers differ in length (20 and 8 bytes, respectively), mainly because TCP has more functions to support. Figure 6-1 shows the TCP and UDP header formats. Note that the TCP header shows the Options field, which extends the length of the TCP header past 20 bytes, but always to some multiple of 4 bytes.

Figure 6-1 *TCP and UDP Headers*



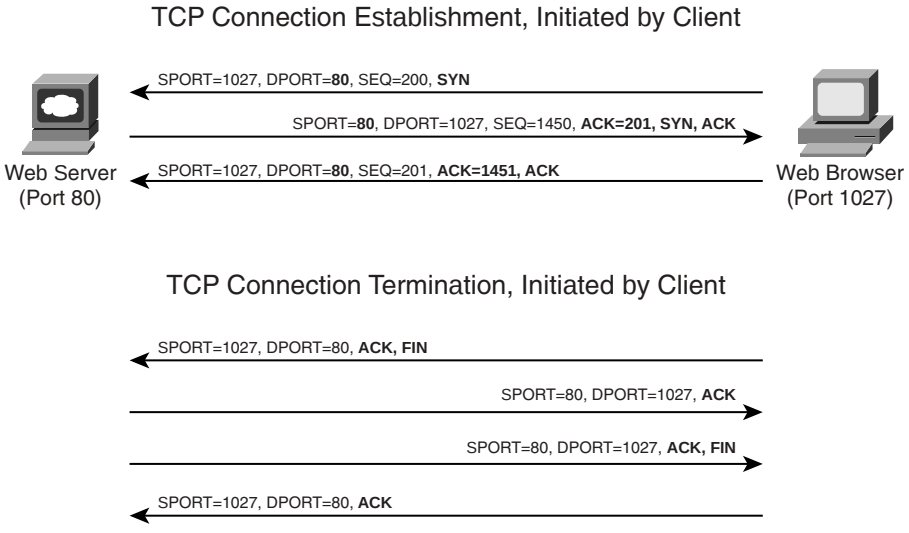
KEY POINT

Note that UDP operates identically to TCP in regard to the Port Number fields and the Header Checksum field. The UDP Length field identifies the length of the UDP message, including the UDP header and data. The fields inside the TCP header are explained in the next few sections of this chapter.

TCP Connections and Port Numbers

Two host applications using TCP must establish a TCP connection before data can flow. Each connection exists between a pair of TCP sockets, with a *socket* being defined as an IP address of the host, the port number used, and the transport layer protocol (TCP in this case). The connection establishment process essentially initializes the sockets, including the source and destination ports, as well as the Sequence and Acknowledgement fields. Figure 6-2 depicts a typical three-way TCP connection establishment, as well as the typical four-way connection termination process.

Figure 6-2 *TCP Connection Establishment and Termination*



In the connection establishment phase, the two hosts select port numbers, select the Sequence and Acknowledgement fields, and use TCP code bits to identify the messages in the three-way handshake for connection establishment. First, for port numbers, the server must already be listening for connection requests from clients, with those requests being to a particular well-known port—in this case, HTTP port 80. (Well-known ports are listed at http://www.iana.org.) The client picks a currently unused port number to use as the source port, typically a value of 1024 or greater. Note in Figure 6-2 that when comparing the segments going in opposite directions, the Port Number fields are reversed.

The TCP header includes several 1-bit fields, called *code bits* or *flags*, that are used for a variety of purposes. The SYN and ACK flags identify segments as either the first or second in a new TCP connection: a segment with just the SYN flag set is the first segment in a new connection, and a segment with both SYN and ACK set is the second segment in a new connection. These flags allow hosts to easily recognize new connection requests.

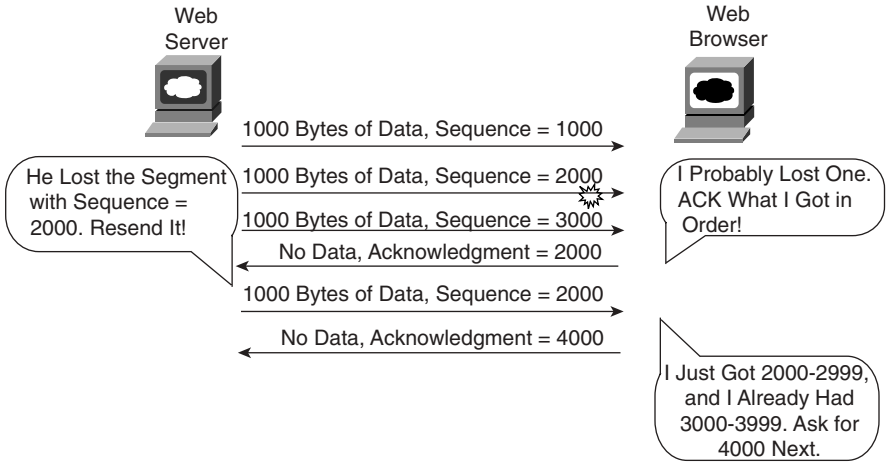
The initial sequence numbers can be set to any valid value, and often are not set to 0. Remember that error recovery, using these fields, happens independently in both directions. So, the first segment in the three-way handshake of Figure 6-2 sets the left-to-right sequence number; the second segment sets the right-to-left sequence number and also acknowledges the first segment; and the third segment acknowledges the right-to-left sequence number.

Connection termination can be accomplished in one of several ways. The most benign case uses a four-segment flow, as shown in the bottom half of Figure 6-2, with the ACK and FIN flags being used.

TCP Error Recovery

To perform error recovery, TCP acknowledges the receipt of data; when data is not acknowledged, the TCP sender can resend the data. Figure 6-3 shows an example in which the web server sends three 1000-byte segments, with the second segment being lost, and the data being recovered.

Figure 6-3 *TCP Error Recovery*



The example shows a case of error recovery in which the sender (the web server) got an acknowledgement that implied that at least one segment was lost. Note that the Acknowledgement field states the next expected byte—not the last acknowledged byte. Also note that the Acknowledgement (and Sequence) fields number the bytes, not the TCP segments. Also, the sender keeps a timer, based on TCP's *Measured Round-Trip Time (MRTT)*, so that if an expected acknowledgement is not received, the sender can resend all the unacknowledged data without waiting for the receiver to request a repeat.

TCP Dynamic Windowing

Like many other protocols that perform error recovery, TCP uses a *sliding window* mechanism to perform flow control. The mechanics are probably familiar to most readers—the receiver states a window size, in bytes, using the Window field of its TCP segments sent over the TCP connection. This window is sometimes called the *receiver's window*, the *receiver's advertised window*, or the *granted window*. The sender can then send only one window's worth of data to the receiver without receiving an acknowledgement. The end goal with this basic dynamic windowing is to allow the receiver to dictate how fast the sender can send data, thereby protecting the receiver from running out of memory. The receiver can increase or decrease the sender's window size for the connection by changing the window size stated in subsequent TCP segments. For instance, in Figure 6-3, the web server would have needed a window size of at least 3000 to send the first three segments shown in the figure.

Interestingly, TCP senders limit their dynamic window based on two different factors—the popularly known advertised window, described in the previous paragraph, and another mechanism that allows the sender to react to packet loss by lowering the size of the window. This additional mechanism, defined in RFC 2581 as "TCP Congestion Control," defines the details of how TCP should react to segment loss by slowing down. In effect, a TCP sender uses a sliding window, with the sliding window being the smaller of the two possible values—either the window granted to the sender by the receiver, or the calculated window defined in RFC 2581. This calculated window is called the *congestion window (CWND)*, and is the most important part of TCP Congestion Avoidance logic. This logic is summarized in the following list, with details to follow:

KEY POINT

1. At connection establishment, CWND is set to a low value, often equal to one maximum segment size (MSS).
2. If no segments are lost, CWND grows using *Slow Start* logic, which increases CWND at an exponential rate.
3. For each lost segment, CWND is halved.
4. After the lost segments have been successfully re-sent, CWND grows again, beginning with Slow Start.
5. After loss of segment(s), while CWND grows, it grows using Slow Start logic until CWND reaches a value of half the original CWND. Then, CWND slows down its growth, using an algorithm called *Congestion Avoidance*, which increases CWND at a linear rate.

More generally, the algorithm starts with a small window. The window grows rapidly, but when congestion occurs, the window is limited by lowering the CWND variable. Once the data is successfully re-sent, CWND can grow rapidly—but not too rapidly, because this TCP sender might have been causing the congestion that resulted in the original packet loss. The next few paragraphs take a closer look at the component steps in the list.

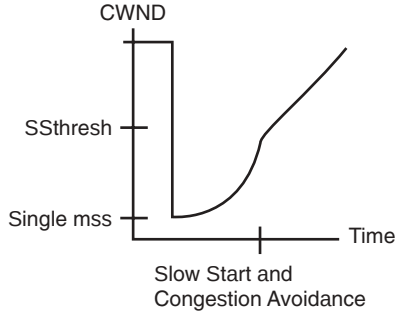
In Step 1, the TCP sender can send only one maximum size segment before requiring an acknowledgement—a relatively severe, small window. The TCP *maximum segment size* (MSS) defines the largest allowed size of the TCP Data field, not counting the TCP header itself. With the typical IP MTU default on most interfaces being 1500 bytes (MTU includes the IP header, by the way), the typical MSS is 1460.

In Step 2, CWND grows exponentially, using an algorithm called TCP Slow Start—a seemingly contradictory name. (Slow Start can be thought of as "slower than if there were no congestion window concept.") CWND is likely to grow to some number larger than the granted window over the course of a few seconds.

Most loss occurs because of congestion, so in Step 3, CWND is lowered quickly. The idea is simple: slow down TCP senders when packets are lost, which in turn should decrease network congestion. The reaction, halving CWND for each lost segment, means that the reaction is swift.

In Steps 4 and 5, the segments have been recovered, and the sender could be limiting its window based on the recently-lowered CWND. CWND can grow again at this point, but rather than rush immediately to the previous-high CWND value, the RFC calls for a less-aggressive growth rate as CWND approaches its value prior to the segment loss. Figure 6-4 depicts a graph of what happens with CWND upon segment loss, and how the sender grows CWND after the packet loss.

Figure 6-4 *CWND Growth with Slow Start and Congestion Avoidance After Multiple Segment Loss*



The graph shows the changes to CWND from connection establishment through multiple segment loss. Because multiple segments were lost, the TCP sender halved CWND multiple times, and in this case, CWND was lowered to its minimum value—a single MSS. Before lowering CWND, the sender calculated a variable called *Slow Start Threshold* (SSThresh), which is half of the original CWND before segments were lost.

Now back to Steps 4 and 5. In Step 4, the sender grows CWND using the exponential Slow Start logic, until CWND reaches SSThresh, at which point it grows CWND more slowly, using TCP Congestion Avoidance logic. Essentially, the SSThresh variable identifies the threshold at which the sender stops using Slow Start logic, and transitions to using Congestion Avoidance logic, to grow CWND.

TCP Header Miscellany

Thus far, this chapter has reviewed many of the features of TCP, including many of the TCP Header fields. Table 6-3 lists the Header fields not mentioned elsewhere in this chapter, with a brief explanation of each one.

Table 6-3 *Remaining TCP Header Fields*

Field Name	Function
PSH	Meaning "Push," this code bit (represented by *Code bits* in the TCP header diagram in Figure 6-3) is set by a TCP sender to cause the TCP receiver to immediately pass that segment's data to the receiver's application socket, along with all other in-order data that the receiver has yet to give to the application.
URG	Meaning "Urgent," this code bit (represented by *Code bits* in the TCP header diagram in Figure 6-3) is set by the TCP sender to cause the TCP receiver to immediately pass this segment's data to the receiving application process. Any other data that had been received earlier on this connection, but not yet passed to the application, does not have to be passed to the receiving application at this time.
Urgent Pointer	Used only when the URG bit is set, this 16-bit field defines the offset into this one segment's Data field at which the urgent data ends.
Offset (HLEN)	Defines the length of the header (HLEN), alternately called the offset to reach the end of the header. The value in the field, multiplied by 4, is the actual header length in bytes.
Checksum	Checksum computed against the TCP header, not including the TCP data.

TCP/IP Applications

Although most CCIE candidates use the TCP/IP application protocols on a regular basis, some of the underlying details may be easily ignored while using the protocols. Therefore, Table 6-4 summarizes the base RFC, transport layer protocol, and TCP or UDP port number used by many of the more popular TCP/IP application protocols.

Table 6-4 *TCP/IP Application Layer Protocols*

KEY POINT

Protocol	Transport Protocol	Well-Known Port	RFC
Telnet	TCP	23	318
FTP	TCP	20, 21	959
TFTP	UDP	69	1350
SMTP	TCP	25	2821
POP3	TCP	110	1939
IMAP	TCP	143	3501
TLS/SSL*	TCP	443	2246
LDAP	TCP	389	2251

Table 6-4 *TCP/IP Application Layer Protocols (Continued)*

KEY POINT

Protocol	Transport Protocol	Well-Known Port	RFC
HTTP	TCP	80	2616
DNS	TCP/UDP	53	1034, 1035
NetBIOS Name Service	UDP	137	1002
NetBIOS Datagram	UDP	138	1002
NetBIOS Session Service	TCP	139	1002
SNMP	UDP	161, 162	Various
BGP	TCP	179	Various
RIP	UDP	520	Various

* SSL is defined by Netscape, with TLS being a very similar protocol based on SSL and standardized by the IETF.

The next few sections cover a handful of topics related to some of the application layer protocols.

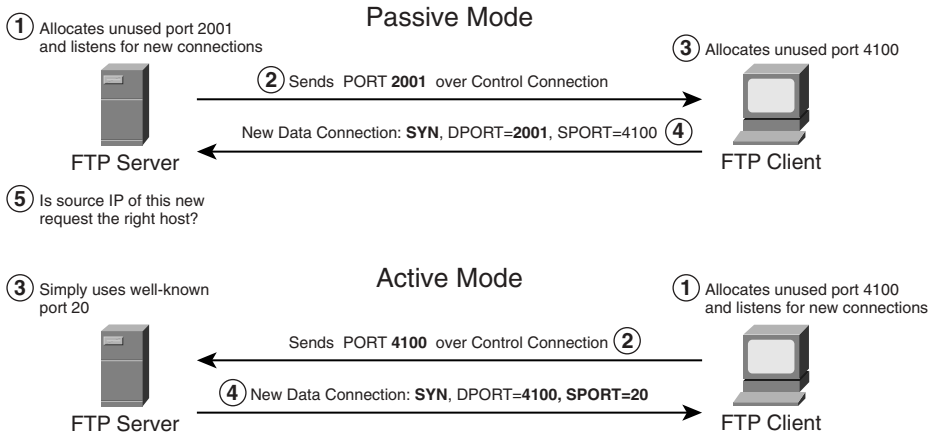
Passive and Active Mode FTP

FTP clients and servers use the typical TCP/IP client/server model for the FTP control connection, with the client initiating a TCP connection to well-known FTP port 21. FTP transfers commands and command acknowledgements over the this TCP control connection. However, at some point, data needs to be transferred—and FTP uses a separate but correlated TCP connection for the actual data transfer.

FTP clients use one of two modes, passive or active, to define the details of how an FTP data connection is established. With FTP *active* mode, the server initiates the TCP connection to the client, but with FTP *passive* mode, the client initiates the connection to the server. But passive mode has a few twists as well, as shown in Figure 6-5. The top part of the figure shows a passive FTP data connection establishment:

1. The server allocates a dynamic unused port number and starts listening on that port.
2. The server uses the FTP PORT command to tell the client on which port the server is now listening.
3. The client allocates a local unused port number.
4. The client initiates a new data connection to the server.
5. The server confirms that this one data connection request came from the same client to which the server sent the message in Step 2.

Figure 6-5 *Passive and Active FTP Behavior*



The bottom half of Figure 6-5 shows an active FTP data connection establishment:

1. The client allocates a dynamic unused port number and starts listening on that port.
2. The client uses the FTP PORT command to tell the server on which port the client is now listening.
3. The server uses well-known port 20.
4. The server initiates a new data connection to the client.

Table 6-5 compares the two FTP modes with regard to the key points shown in Figure 6-5.

Table 6-5 *Comparison of FTP Active and Passive Modes*

KEY POINT

	Active Mode	Passive Mode
Port number on client	Dynamic port	Dynamic port
Port number on server	20	Dynamic port
Who first listens on a local port, in preparation for an incoming TCP connection?	Client	Server
Who uses the FTP PORT command to tell the other host on which port it is listening?	Client	Server
Who initiates the connection?	Server	Client

Application Authentication and Privacy

Many of the application layer protocols mentioned in this chapter have built-in authentication through the use of a basic password mechanism. However, many of these protocols—for instance, Telnet, FTP, and TFTP—send the usernames and passwords in clear text.

Some of the more recently defined or updated protocols provide stronger security. For example, Secure Shell (SSH) provides an alternative to Telnet but with strong authentication and privacy. SSH was originally intended as a secure replacement for the Unix r-commands. SSH uses IPsec encryption for privacy and authentication. POP3 is another example of stronger authentication, with a hash algorithm used to create a one-time digest of the password, which allows authentication without sending the password in clear text.

Secure Sockets Layer (SSL) is a particularly important security protocol related to TCP/IP applications. SSL gets its name in part from the fact that it sits just below the TCP/IP application protocol, with those applications using sockets. SSL provides security to any application that uses SSL by using a message digest for strong authentication, and by using encryption for privacy. SSL was originally created by Netscape to be used with HTTP, creating a secure HTTP protocol that is often called HTTPS for "HTTP Secure." Netscape has developed specifications up through SSL Version 3.

SSL has been standardized by the IETF in RFC 2246 and renamed *Transport Layer Security (TLS)*. (Note that TLS Version 1 and SSL Version 3 are not completely compatible, but they perform the same type of functions.) Because SSL/TLS provides strong authentication and privacy for any application that uses it, the IETF has defined several RFCs that detail how unsecure, older applications can use TLS for better security. For example, HTTP, SMTP, POP3, and IMAP all have RFCs that define how they could use TLS.

Network Management and SNMP

This final section of the chapter summarizes some of the core SNMP concepts and details, particularly with regard to features of different SNMP versions. The Simple Network Management Protocol (SNMP), or more formally, the *Internet Standard Management Framework*, uses a structure in which the device being managed (the SNMP agent) has information that the management software (the SNMP manager) wants to display to someone operating the network. Each SNMP agent keeps a database, called a *Management Information Base (MIB)*, that holds a large variety of data about the operation of the device on which the agent resides. The manager collects the data by using SNMP.

SNMP has been defined with four major functional areas to support the core function of allowing managers to manage agents:

- **Data Definition**—The conventions for syntax for how to define the data to an agent or manager. These specifications are called the *Structure of Management Information (SMI)*.
- **MIBs**—Over 100 Internet standards define different MIBs, each for a different technology area, with countless vendor-proprietary MIBs as well. The MIB definitions conform to the appropriate SMI version.
- **Protocols**—The messages used by agents and managers to exchange management data.
- **Security and Administration**—Definitions for how to secure the exchange of data between agents and managers.

Interestingly, by separating SNMP into these major functional areas, each part has been improved and expanded independently over the years. However, it is important to know a few of the main features added for each official SNMP version, as well as for a pseudo-version called SNMPv2c, as summarized in Table 6-6.

Table 6-6 *SNMP Version Summaries*

KEY POINT

SNMP Version	Description
1	Uses SMIv1, simple authentication with communities, but used MIB-I originally.
2	Uses SMIv2, removed requirement for communities, added GetBulk and Inform messages, but began with MIB-II originally.
2c	Pseudo-release (RFC 1905) that allowed SNMPv1-style communities with SNMPv2; otherwise equivalent to SNMPv2.
3	Mostly identical to SNMPv2, but adds significantly better security, although it supports communities for backward compatibility. Uses MIB-II.

Table 6-6 hits the highlights of the comparison points between the various SNMP versions. As you might expect, each release builds on the previous one. For example, SNMPv1 defined *community strings* for use as simple clear-text passwords. SNMPv2 removed the requirement for community strings—however, backward compatibility for SNMP communities was defined via an optional RFC (1901). Even SNMPv3, with much better security, supports communities to allow backward compatibility.

NOTE The use of SNMPv1 communities with SNMPv2, based on RFC 1901, has popularly been called *SNMP Version 2c*, with *c* referring to "communities," although it is arguably not a legitimate full version of SNMP.

The next few sections provide a bit more depth about the SNMP protocol, with additional details about some of the version differences.

SNMP Protocol Messages

The SNMPv1 and SNMPv2 protocol messages (RFC 3416) define how a manager and agent, or even two managers, can communicate information. For instance, a manager can use three different messages to get MIB variable data from agents, with an SNMP *Response* message returned by the agent to the manager supplying the MIB data. SNMP uses UDP exclusively for transport, using the SNMP Response message to both acknowledge receipt of other protocol messages and supply SNMP information.

Table 6-7 summarizes the key information about each of the SNMP protocol messages, including the SNMP version in which the message first appeared.

Table 6-7 *SNMP Protocol Messages (RFCs 1157 and 1905)*

KEY POINT

Message	Initial Version	Response Message	Typically Sent By	Main Purpose
Get	1	Response	Manager	A request for a single variable's value.
GetNext	1	Response	Manager	A request for the next single MIB leaf variable in the MIB tree.
GetBulk	2	Response	Manager	A request for multiple consecutive MIB variables with one request. Useful for getting complex structures, for example, an IP routing table.
Response	1	None	Agent	Used to respond with the information in Get and Set requests.
Set	1	Response	Manager	Sent by a manager to an agent to tell the agent to set a variable to a particular value. The agent replies with a Response message.
Trap	1	None	Agent	Allows agents to send unsolicited information to an SNMP manager. The manager does not reply with any SNMP message.
Inform	2	Response	Manager	A message used between SNMP managers to allow MIB data to be exchanged.

The three variations of the SNMP Get message, and the SNMP Response message, are typically used when someone is actively using an SNMP manager. When a user of the SNMP manager asks for information, the manager sends one of the three types of Get commands to the agent. The agent replies with an SNMP Response message. The different variations of the Get command are useful, particularly when the manager wants to view large portions of the MIB. An agent's entire MIB—whose structure can vary from agent to agent—can be discovered with successive GetNext requests, or with GetBulk requests, using a process called a *MIB walk*.

The SNMP Set command allows the manager to change something on the agent. For example, the user of the management software can specify that a router interface should be shut down; the management station can then issue a Set command for a MIB variable on the agent. The agent sets the variable, which tells Cisco IOS Software to shut down the interface.

SNMP Traps are unsolicited messages sent by the agent to the management station. For example, when an interface fails, a router's SNMP agent could send a Trap to the SNMP manager. The management software could then highlight the failure information on a screen, e-mail first-level support personnel, page support, and so on. Also of note, there is no specific message in response to the receipt of a Trap; technically, of the messages in Table 6-7, only the Trap and Response messages do not expect to receive any kind of acknowledging message.

Finally, the Inform message allows two SNMP managers to exchange MIB information about agents that they both manage.

SNMP MIBs

SNMP Versions 1 and 2 included a standard generic MIB, with initial MIB-I (version 1, RFC 1156) and MIB-II (version 2, RFC 1213). MIB-II was actually created in between the release of SNMPv1 and v2, with SNMPv1 supporting MIB-II as well. After the creation of the MIB-II specification, the IETF SNMP working group changed the strategy for MIB definition. Instead of the SNMP working group creating standard MIBs, other working groups, in many different technology areas, were tasked with creating MIB definitions for their respective technologies. As a result, hundreds of standardized MIBs are defined. Additionally, vendors create their own vendor-proprietary MIBs.

The Remote Monitoring MIB (RMON, RFC 2819) is a particularly important standardized MIB outside MIB-II. An SNMP agent that supports the RMON MIB can be programmed, through SNMP Set commands, to capture packets, calculate statistics, monitor thresholds for specific MIB variables, report back to the management station when thresholds are reached, and perform other tasks. With RMON, a network can be populated with a number of monitoring probes, with SNMP messaging used to gather the information as needed.

SNMP Security

SNMPv3 added solid security to the existing SNMPv2 and SNMPv2c specifications. SNMPv3 adds two main branches of security to SNMPv2: authentication and encryption. SNMPv3 specifies the use of MD5 and SHA to create a message digest for each SNMPv3 protocol message. Doing so enables authentication of endpoints, as well as prevent data modification and masquerade types of attacks. Additionally, SNMPv3 managers and agents can use Digital Encryption Standard (DES) to encrypt the messages, providing better privacy. (SNMPv3 suggests future support of Advanced Encryption Standard [AES] as well, but that is not a part of the original SNMPv3 specifications.) The encryption feature remains separate due to the U.S. government export restrictions on DES technology.

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most of the Cisco Press *Exam Certification Guides*, this book does not repeat information presented in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in the "Foundation Topics" section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 6-8 lists some of the SNMP-related protocol specifications mentioned in this chapter, and refers to their respective standards documents. Note that Table 6-4 lists most of the other protocols in the chapter.

Table 6-8 *Protocols and Standards for Chapter 6*

Name	Standardized in Many Documents, Including the following RFCs . . .
SNMP Version 1	1155, 1156, 1212, 1157, 1213, 1215
SNMP Version 2	1902–1907, 3416
SNMP Version 2c	1901
SNMP Version 3	2578–2580, 3410–3415
Good Starting Point:	3410

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http//www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

> passive mode FTP, active mode FTP, SNMP agent, SNMP manager, Get, GetNext, MRTT, GetBulk, MIB-I, MIB-II, Response, Trap, Set, Inform, SMI, MIB, CWND, SSThresh, window, Slow Start, Congestion Avoidance, MSS, MTU, socket, TCP code bits, TCP flags, receiver's advertised window, MIB walk

Refer to the CD-based glossary to check your answers.

Further Reading

Because this chapter focuses on TCP/IP protocols, much more information can be found in the RFCs mentioned earlier in the chapter.

A wonderful reference for the topics in this chapter is Douglas Comer's classic book *Internetworking with TCP/IP*, considered by many to be the bible of TCP/IP.

Any further reading of SNMP-related RFCs should begin with RFC 3410, which provides a great overview of the releases, and points to the more important of the vast number of SNMP-related RFCS.

Part III: IP Routing

Chapter 7 IP Forwarding (Routing)

Chapter 8 RIP Version 2

Chapter 9 EIGRP

Chapter 10 OSPF

Chapter 11 IGP Route Redistribution, Route Summarization, and Default Routing

Chapter 12 Fundamental BGP Operations

Chapter 13 BGP Routing Policies

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- IP Routing
  - — Route Filtering and Policy Routing
- WAN
  - — Frame Relay

In addition, this chapter covers information related to the following specific CCIE Routing and Switching exam topics:

- IP forwarding
- Classless and classful routing
- Cisco Express Forwarding (CEF)
- Frame Relay Inverse ARP
- Multilayer Switching (MLS)

CHAPTER 7

IP Forwarding (Routing)

Chapter 7 begins the largest part of the book. This part of the book, containing Chapters 7 through 13, focuses on the topics that are the most important and popular for both the CCIE Routing and Switching written and practical (lab) exams.

Chapter 7 begins with coverage of the details of the forwarding plane—the actual forwarding of IP packets. This process of forwarding IP packets is often called *IP routing*, or simply *routing*. Also, many people also refer to IP routing as the *data plane*, meaning the plane (topic) related to the end-user data.

Chapters 8 through 13 cover the details of the IP *control plane*. In contrast to the term data plane, the control plane relates to the communication of control information—in short, routing protocols like OSPF and BGP. These chapters cover the routing protocols on the exam, one chapter per routing protocol, plus an additional chapter on redistribution and route summarization.

"Do I Know This Already?" Quiz

Table 7-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 7-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
IP Forwarding	1–6	
Multilayer Switching	7–8	
Policy Routing	9–10	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. What command is used to enable CEF globally for IP packets?
   a. **enable cef**
   b. **ip enable cef**
   c. **ip cef**
   d. **cef enable**
   e. **cef enable ip**
   f. **cef ip**

2. Which of the follow triggers an update to a CEF FIB?
   a. Receipt of a Frame Relay InARP message with previously unknown information
   b. Receipt of a LAN ARP reply message with previously unknown information
   c. Addition of a new route to the IP routing table by EIGRP
   d. Addition of a new route to the IP routing table by adding an **ip route** command
   e. The removal of a route from the IP routing table by EIGRP

3. Router1 has a Frame Relay access link attached to its s0/0 interface. Router1 has a PVC connecting it to Router3. What action triggers Router3 to send an InARP message over the PVC to Router1?
   a. Receipt of a CDP multicast on the PVC connected to Router1
   b. Receipt of an InARP request from Router1
   c. Receipt of a packet that needs to be routed to Router1
   d. Receipt of a Frame Relay message stating the PVC to Router1 is up

4. Three routers are attached to the same Frame Relay network, have a full mesh of PVCs, and use IP addresses 10.1.1.1/24 (R1), 10.1.1.2/24 (R2), and 10.1.1.3 (R3). R1 has its IP address configured on its physical interface; R2 and R3 have their IP addresses configured on multipoint subinterfaces. Assuming all the Frame Relay PVCs are up and working, and the router interfaces have been administratively enabled, which of the following is true?
   a. R1 can ping 10.1.1.2.
   b. R2 cannot ping 10.1.1.3.
   c. R3 can ping 10.1.1.2.
   d. In this case, R1 must rely on mapping via InARP to be able to ping 10.1.1.3.

5. Three routers are attached to the same Frame Relay network, with a partial mesh of PVCs: R1-R2 and R1-R3. The routers use IP addresses 10.1.1.1/24 (R1), 10.1.1.2/24 (R2), and 10.1.1.3/24 (R3). R1 has its IP address configured on its physical interface; R2 has its IP address configured on a multipoint subinterface; and R3 has its IP address configured on a point-to-point subinterface. Assuming all the Frame Relay PVCs are up and working, and the router interfaces have been administratively enabled, which of the following is true? Assume no **frame-relay map** commands have been configured.

   a. R1 can ping 10.1.1.2.

   b. R2 can ping 10.1.1.3.

   c. R3 can ping 10.1.1.1.

   d. R3's **ping 10.1.1.2** command results in R3 not sending the ICMP Echo packet.

   e. R2's **ping 10.1.1.3** command results in R2 not sending the ICMP Echo packet.

6. Router1 has an OSPF-learned route to 10.1.1.0/24 as its only route to a subnet on class A network 10.0.0.0. It also has a default route. When Router1 receives a packet destined for 10.1.2.3, it discards the packet. Which of the following commands would make Router1 use the default route for those packets in the future?

   a. **ip classless** subcommand of **router ospf**

   b. **no ip classful** subcommand of **router ospf**

   c. **ip classless** global command

   d. **no ip classless** global command

   e. **no ip classful** global command

7. Which of the following commands is used on a Cisco IOS Layer 3 switch to use the interface as a *routed interface* instead of a *switched interface*?

   a. **ip routing** global command

   b. **ip routing** interface subcommand

   c. **ip address** interface subcommand

   d. **switchport access layer-3** interface subcommand

   e. **no switchport** interface subcommand

8. On a Cisco 3550 switch with Enterprise Edition software, the first line of the output of a **show interface vlan 55** command lists the state as "Vlan 55 is down, line protocol is down down." Which of the following might be causing that state to occur?

   a. VLAN interface has not been **no shut** yet.

   b. The **ip routing** global command is missing from the configuration.

   c. On at least one interface in the VLAN, a cable that was previously plugged in has been unplugged.

   d. VTP mode is set to transparent.

   e. The VLAN has not yet been created on this switch.

9. Imagine a route map used for policy routing, in which the route map has a **set interface default serial0/0** command. Serial0/0 is a point-to-point link to another router. A packet arrives at this router, and the packet matches the policy routing **route-map** clause whose only **set** command is the one just mentioned. Which of the following general characterizations is true?

   a. The packet will be routed out interface s0/0; if s0/0 is down, it will be routed using the default route from the routing table.

   b. The packet will be routed using the default route in the routing table; if there is no default, the packet will be routed out s0/0.

   c. The packet will be routed using the best match of the destination address with the routing table; if no match is made, the packet will be routed out s0/0.

   d. The packet will be routed out interface s0/0; if s0/0 is down, the packet will be discarded.

10. Router1 has an fa0/0 interface and two point-to-point WAN links back to the core of the network (s0/0 and s0/1, respectively). Router1 accepts routing information only over s0/0, which Router1 uses as its primary link. When s0/0 fails, Router1 uses policy routing to forward the traffic out the relatively slower s0/1 link. Which of the following **set** commands in Router1's policy routing route map could have been used to achieve this function?

    a. **set ip default next-hop**

    b. **set ip next-hop**

    c. **set ip default interface**

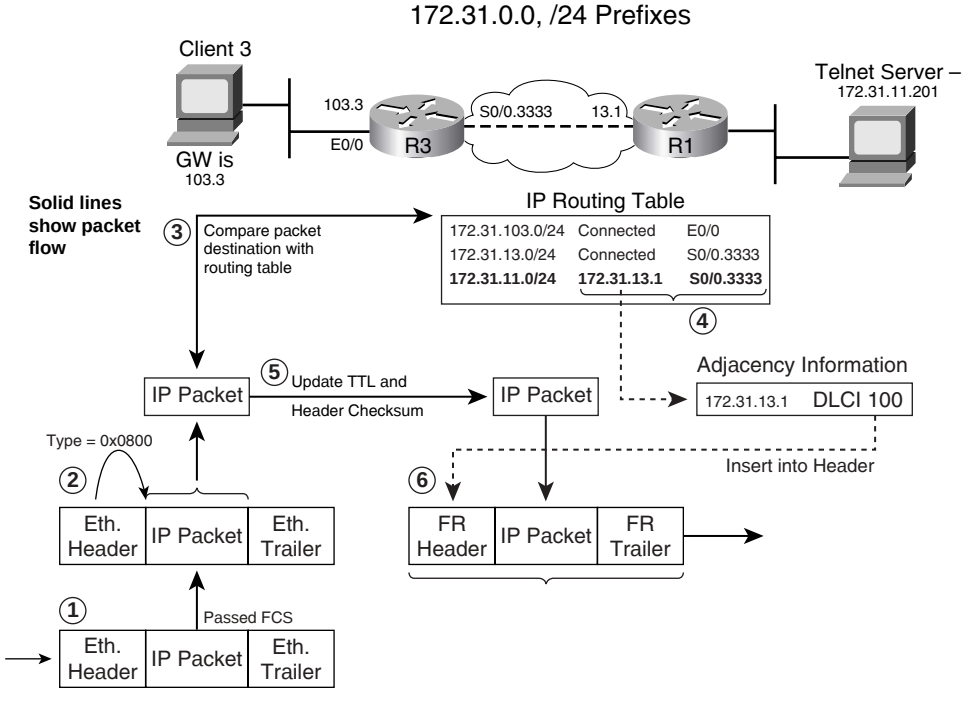
    d. **set ip interface**

Foundation Topics

IP Forwarding

IP forwarding, or *IP routing*, is simply the process of receiving an IP packet, making a decision of where to send the packet next, and then forwarding the packet. The forwarding process needs to be relatively simple, or at least streamlined, for a router to forward large volumes of packets. Ignoring the details of several Cisco optimizations to the forwarding process for a moment, the internal forwarding logic in a router works basically as shown in Figure 7-1.

Figure 7-1 *Forwarding Process at Router3, Destination Server1*



The following list summarizes the key steps shown in Figure 7-1.

KEY POINT

1. A router receives the frame and checks the received frame check sequence (FCS); if errors occurred, the frame is discarded. The router makes no attempt to recover the lost packet.
2. If no errors occurred, the router checks the Ethernet Type field for the packet type, and extracts the packet. The Data Link header and trailer can now be discarded.
3. Assuming an IP packet, the router checks its IP routing table for the most specific prefix match of the packet's destination IP address.

4. The matched routing table entry includes the outgoing interface and next-hop router; this information points the router to the adjacency information needed to build a new Data Link frame.
5. Before creating a new frame, the router updates the IP header TTL field, requiring a recomputation of the IP header checksum.
6. The router encapsulates the IP packet in a new Data Link header (including the destination address) and trailer (including a new FCS) to create a new frame.

The preceding list is a generic view of the process; next, a few words on how Cisco routers can optimize the routing process by using Cisco Express Forwarding (CEF).

Process Switching, Fast Switching, and Cisco Express Forwarding

Steps 3 and 4 from the generic routing logic shown in the preceding section are the most computation-intensive tasks in the routing process. A router must find the best route to use for every packet, requiring some form of table lookup of routing information. Also, a new Data Link header and trailer must be created, and the information to put in the header (like the destination Data Link address) must be found in another table.

Cisco has created several different methods to optimize the forwarding processing inside routers, termed *switching paths*. This section examines the two most likely methods to exist in Cisco router networks today: fast switching and CEF.

With fast switching, the first packet to a specific destination IP address is *process switched*, meaning that it follows the same general algorithm as in Figure 7-1. With the first packet, the router adds an entry to the *fast-switching cache*, sometimes called the *route cache*. The cache has the destination IP address, the next-hop information, and the data link header information that needs to be added to the packet before forwarding (as in Step 6 in Figure 7-1). Future packets to the same destination address match the cache entry, so it takes the router less time to process and forward the packet.

Although it is much better than process switching, fast switching has a few drawbacks. The first packet must be process switched. The cache entries are timed out relatively quickly, because otherwise the cache could get overly large as it has an entry per each destination address, not per destination subnet/prefix. Also, load balancing can only occur per destination with fast switching.

CEF overcomes the main shortcoming of fast switching. CEF optimizes the route lookup process by using a construct called the *Forwarding Information Base (FIB)*. The FIB contains information about all the known routes in the routing table. Rather than use a table that is updated when new flows appear, as did Cisco's earlier fast-switching technology, CEF loads FIB entries as routes are added and removed from the routing table. CEF does not have to time out the entries in the FIB, does not require the first packet to a destination to be process switched, and allows much more effective load balancing over equal-cost routes.

When a new packet arrives, CEF routers first search the FIB. Cisco designed the CEF FIB structure as a special kind of tree, called an *mtrie*, that significantly reduces the time taken to match the packet destination address to the right CEF FIB entry.

The matched FIB entry points to an entry in the CEF *adjacency table*. The adjacency table lists the outgoing interface, along with all the information needed to build the Data Link header and trailer before sending the packet. When a router forwards a packet using CEF, it easily and quickly finds the corresponding CEF FIB entry, after which it has a pointer to the adjacency table entry—which tells the router how to forward the packet.

Table 7-2 summarizes a few key points about the three main options for router switching paths.

Table 7-2 *Matching Logic and Load-Balancing Options for Each Switching Path*

KEY POINT

Switching Path	Tables that Hold the Forwarding Information	Load-Balancing Method
Process switching	Routing table	Per packet
Fast switching	Fast-switching cache (per flow route cache)	Per destination IP address
CEF	FIB and adjacency tables	Per a hash of the packet source and destination, or per packet

The **ip cef** global configuration command enables CEF for all interfaces on a Cisco router. The **no ip route-cache cef** interface subcommand can then be used to selectively disable CEF on an interface. On many of the higher-end Cisco platforms, CEF processing can be distributed to the linecards. Similarly, Cisco multilayer switches use CEF for Layer 3 forwarding, by loading CEF tables into the forwarding ASICs.

Building Adjacency Information: ARP and Inverse ARP

The CEF adjacency table entries list an outgoing interface and a Layer 2 and Layer 3 address reachable via that interface. The table also includes the entire data link header that should be used to reach that next-hop (adjacent) device.

The CEF adjacency table must be built based on the IP routing table, plus other sources. The IP routing table entries include the outgoing interfaces to use and the next-hop device's IP address. To complete the adjacency table entry for that next hop, the router needs to know the Data Link layer address to use to reach the next device. Once known, the router can build the CEF adjacency table entry for that next-hop router. For instance, for Router3 in Figure 7-1 to reach next-hop router 172.31.13.1 (Router1), out interface s0/0.3333, Router3 needed to know the right Data-Link connection identifier (DLCI) to use. So, to build the adjacency table entries, CEF uses the IP ARP cache, Frame Relay mapping information, and other sources of Layer 3-to-Layer 2 mapping information.

First, a quick review of IP ARP. The ARP protocol dynamically learns the MAC address of another IP host on the same LAN. The host that needs to learn the other host's MAC address sends an ARP request, sent to the LAN broadcast address, hoping to receive an ARP reply (a LAN unicast) from the other host. The reply, of course, supplies the needed MAC address information.

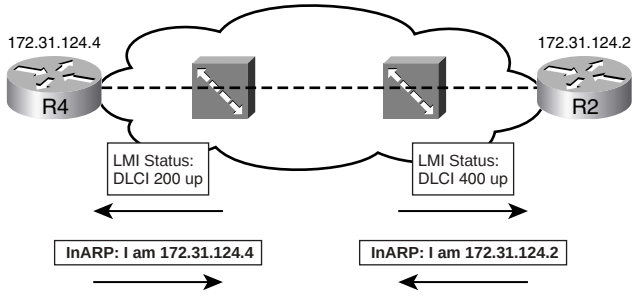
Frame Relay Inverse ARP

IP ARP is widely understood and relatively simple. As a result, it is hard to ask difficult questions about ARP on an exam. However, the topics of Frame Relay Inverse ARP (InARP), its use, defaults, and when static mapping must be used lend themselves to being the source of tricky exam questions. So, this section covers Frame Relay InARP to show some of the nuances of when and how it is used.

InARP discovers the DLCI to use to reach a particular adjacent IP address. However, as the term *Inverse* ARP implies, the process differs from ARP on LANs; with InARP, routers already know the Data Link address (DLCI), and need to learn the corresponding IP address. Figure 7-2 shows an example InARP flow.

Figure 7-2 *Frame Relay InARP*

KEY POINT



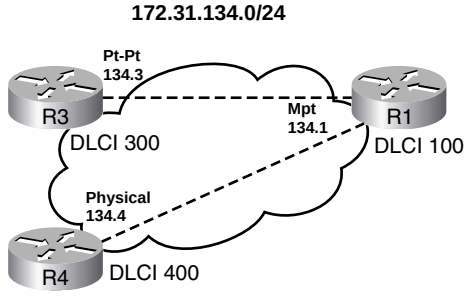
Unlike on LANs, a packet does not need to arrive at the router to trigger the InARP protocol; instead, an LMI status message triggers InARP. After receiving an LMI PVC Up message, each router announces its own IP address over the VC, using an InARP message, as defined in RFC 1293. Interestingly, if you disable LMI, then the InARP process no longer works, because nothing triggers a router to send an InARP message.

NOTE In production Frame Relay networks, the configuration details are chosen to purposefully avoid some of the pitfalls that are covered in the next several pages of this chapter. For example, when using mainly point-to-point subinterfaces, with a different subnet per VC, all the problems described in the rest of the Frame Relay coverage in this chapter can be avoided.

While InARP itself is relatively simple, implementation details differ based on the type of subinterface used in a router. For a closer look at implementation, Figure 7-3 shows an example Frame Relay topology with a partial mesh and a single subnet, in which each router uses a different interface type for its Frame Relay configuration. (You would not typically choose to configure

Frame Relay on physical, point-to-point and multipoint subinterfaces in the same design—indeed, it wreaks havoc with routing protocols if you do so. This example does so just to show in more detail in the examples how InARP really works.) Example 7-1 points out some of the basic **show** and **debug** commands related to Frame Relay InARP, and one of the oddities about InARP relating to point-to-point subinterfaces.

Figure 7-3 *Frame Relay Topology for Frame Relay InARP Examples*



NOTE All figures with Frame Relay networks in this book use Global DLCI conventions unless otherwise stated. For instance, in Figure 7-3, DLCI 300 listed beside Router3 means that, due to Local DLCI assignment conventions by the service provider, all other routers (like Router4) use DLCI 300 to address their respective VCs back to Router3.

Example 7-1 *Frame Relay InARP* **show** *and* **debug** *Commands*

```
! First, Router1 configures Frame Relay on a multipoint subinterface.
Router1# sh run
! Lines omitted for brevity
interface Serial0/0
 encapsulation frame-relay
interface Serial0/0.11 multipoint
 ip address 172.31.134.1 255.255.255.0
 frame-relay interface-dlci 300
 frame-relay interface-dlci 400
! Lines omitted for brevity
! Next, the serial interface is shut and no shut, and the earlier InARP entries
! are cleared, so the example can show the InARP process.
Router1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router1(config)# int s 0/0
Router1(config-if)# do clear frame-relay inarp
Router1(config-if)# shut
Router1(config-if)# no shut
Router1(config-if)# ^Z
```

continues

Example 7-1 *Frame Relay InARP* **show** *and* **debug** *Commands (Continued)*

```
! Messages resulting from the debug frame-relay event command show the
! received InARP messages on Router1. Note the hex values 0xAC1F8603 and
! 0xAC1F8604, which in decimal are 172.31.134.3 and 172.31.134.4 (Router3
! and Router4, respectively).
Router1# debug frame-relay events
*Mar  1 00:09:45.334: Serial0/0.11: FR ARP input
*Mar  1 00:09:45.334: datagramstart = 0x392BA0E, datagramsize = 34
*Mar  1 00:09:45.334: FR encap = 0x48C10300
*Mar  1 00:09:45.334: 80 00 00 00 08 06 00 0F 08 00 02 04 00 09 00 00
*Mar  1 00:09:45.334: AC 1F 86 03 48 C1 AC 1F 86 01 01 02 00 00
*Mar  1 00:09:45.334:
*Mar  1 00:09:45.334: Serial0/0.11: FR ARP input
*Mar  1 00:09:45.334: datagramstart = 0x392B8CE, datagramsize = 34
*Mar  1 00:09:45.338: FR encap = 0x64010300
*Mar  1 00:09:45.338: 80 00 00 00 08 06 00 0F 08 00 02 04 00 09 00 00
*Mar  1 00:09:45.338: AC 1F 86 04 64 01 AC 1F 86 01 01 02 00 00
! Next, note the show frame-relay map command output does include a "dynamic"
! keyword, meaning that the entries were learned with InARP.
Router1# show frame-relay map
Serial0/0.11 (up): ip 172.31.134.3 dlci 300(0x12C,0x48C0), dynamic,
              broadcast,, status defined, active
Serial0/0.11 (up): ip 172.31.134.4 dlci 400(0x190,0x6400), dynamic,
              broadcast,, status defined, active
! On Router3, show frame-relay map only lists a single entry as well, but
! the format is different. Because Router3 uses a point-to-point subinterface,
! the entry was not learned with InARP, and the command output does not include
! the word "dynamic." Also note the absence of any Layer 3 addresses.
Router3# show frame-relay map
Serial0/0.3333 (up): point-to-point dlci, dlci 100(0x64,0x1840), broadcast
          status defined, active
```

KEY POINT

KEY POINT

NOTE Example 7-1 included the use of the **do** command inside configuration mode. The **do** command, followed by any **exec** command, can be used from inside configuration mode to issue an **exec** command, without having to leave configuration mode.

The example **show** commands from Router1 detail the fact that InARP was used; however, the last **show** command in Example 7-1 details how Router3 actually did not use the received InARP information. Cisco IOS Software knows that only one VC is associated with a point-to-point subinterface; any other IP hosts in the same subnet as a point-to-point subinterface can be reached only by that single DLCI. So, any received InARP information related to that DLCI is unnecessary.

For instance, whenever Router3 needs to forward a packet to Router1 (172.31.134.1), or any other host in subnet 172.31.134.0/24, Router3 already knows from its configuration to send the packet

over the only possible DLCI on that point-to-point subinterface—namely, DLCI 100. So, although all three types of interfaces used for Frame Relay configuration support InARP by default, point-to-point subinterfaces ignore the received InARP information.

Static Configuration of Frame Relay Mapping Information

In Figure 7-3, Router3 already knows how to forward frames to Router4, but the reverse is not true. Router3 uses logic like this: "For packets needing to get to a next-hop router that is in subnet 172.31.124.0/24, send them out the one DLCI on that point-to-point subinterface—DLCI 100." The packet then goes over that VC to Router1, which in turn routes the packet to Router4.

In the admittedly poor design shown in Figure 7-3, however, Router4 cannot use the same kind of logic as Router3, as its Frame Relay details are configured on its physical interface. To reach Router3, Router4 needs to send frames over DLCI 100 back to Router1, and let Router1 forward the packet on to Router3. In this case, InARP does not help, because InARP messages only flow across a VC, and are not forwarded; note that there is no VC between Router4 and Router3.

The solution is to add a **frame-relay map** command to Router4's configuration, as shown in Example 7-2. The example begins before Router4 has added the **frame-relay map** command, and then shows the results after having added the command.

Example 7-2 *Using the* **frame-relay map** *Command—Router4*

```
! Router4 only lists a single entry in the show frame-relay map command
! output, because Router4 only has a single VC, which connects back to Router1.
! With only 1 VC, Router4 could only have learned of 1 other router via InARP.
Router4# sh run
! lines omitted for brevity
interface Serial0/0
 ip address 172.31.134.4 255.255.255.0
 encapsulation frame-relay
Router4# show frame-relay map
Serial0/0 (up): ip 172.31.134.1 dlci 100(0x64,0x1840), dynamic,
              broadcast,, status defined, active
! Next, proof that Router4 cannot send packets to Router3's Frame Relay IP address.
Router4# ping 172.31.134.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.31.134.3, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
! Next, static mapping information is added to Router4 using the frame-relay map
! interface subcommand. Note that the command uses DLCI 100, so that any packets
! sent by Router4 to 172.31.134.3 (Router3) will go over the VC to Router1, which
! will then need to route the packet to Router3. The broadcast keyword tells
```

continues

Example 7-2 *Using the* **frame-relay map** *Command—Router4 (Continued)*

```
! Router4 to send copies of broadcasts over this VC.
Router4# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router4(config)# int s0/0
Router4(config-if)# frame-relay map ip 172.31.134.3 100 broadcast
Router4(config-if)# ^Z
Router4# ping 172.31.134.3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.31.134.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/20/20 ms
```

NOTE Remember, Router3 did not need a **frame-relay map** command, due to the logic used for a point-to-point subinterface.

Keep in mind that you probably would not choose to build a network like the one shown in Figure 7-3 using different subinterface types on the remote routers, nor would you typically put all three non-fully-meshed routers into the same subnet unless you were seriously constrained in your IP address space.

In cases where you do use a topology like that shown in Figure 7-3, you can use the configuration described in the last few pages. Alternatively, if both Router3 and Router4 had used multipoint subinterfaces, they would both have needed **frame-relay map** commands, because these two routers could not have heard InARP messages from the other router. However, if both Router3 and Router4 had used point-to-point subinterfaces, neither would have required a **frame-relay map** command, due to the "use this VC to reach all addresses in this subnet" logic.

Disabling InARP

In most cases for production networks, using InARP makes sense. However, InARP can be disabled on multipoint and physical interfaces using the **no frame-relay inverse-arp** interface subcommand. InARP can be disabled for all VCs on the interface/subinterface, all VCs on the interface/subinterface for a particular Layer 3 protocol, and even for a particular Layer 3 protocol per DLCI.

Interestingly, the **no frame-relay inverse-arp** command not only tells the router to stop sending InARP messages, but also tells the router to ignore received InARP messages. For instance, the **no frame-relay inverse-arp ip 400** subinterface subcommand on Router1 in Example 7-2 not only prevents Router1 from sending InARP messages over DLCI 400 to Router4, but also causes Router1 to ignore the InARP received over DLCI 400.

Table 7-3 summarizes some of the key details about Frame Relay Inverse ARP settings in IOS.

Table 7-3 *Facts and Behavior Related to InARP*

KEY POINT

Fact/Behavior	Point-to-Point	Multipoint or Physical
Does InARP require LMI?	Always	Always
Is InARP enabled by default?	Yes	Yes
Can InARP be disabled?	No	Yes
Ignores received InARP messages?	Always[1]	When InARP is disabled

[1]Point-to-point interfaces ignore InARP messages because of their "send all packets for addresses in this subnet using the only DLCI on the subinterface" logic.

Classless and Classful Routing

So far this chapter has reviewed the basic forwarding process for IP packets in a Cisco router. The logic requires matching the packet destination with the routing table, or with the CEF FIB if CEF is enabled, or with other tables for the other options Cisco uses for route table lookup. (Those options include fast switching in routers and NetFlow switching in multilayer switches, both of which populate an optimized forwarding table based on flows, but not on the contents of the routing table.)

Classless routing and *classful routing* relate to the logic used to match the routing table, specifically for when the default route is used. Regardless of the use of any optimized forwarding methods (for instance, CEF), the following statements are true about classless and classful routing:

KEY POINT

- **Classless routing**—When a default route exists, and no specific match is made when comparing the destination of the packet and the routing table, the default route is used.
- **Classful routing**—When a default route exists, and the class A, B, or C network for the destination IP address does not exist at all in the routing table, the default route is used. If any part of that classful network exists in the routing table, but the packet does not match any of the existing subnets of that classful network, the router does not use the default route and thus discards the packet.

Typically, classful routing works well in enterprise networks only when all the enterprise routes are known by all routers, and the default is used only to reach the Internet-facing routers. Conversely, for enterprise routers that normally do not know all the routes—for instance, if a remote router has only a few connected routes to network 10.0.0.0 and a default route pointing back to a core site—classless routing is required. For instance, in an OSPF design using stubby areas, default routes are injected into the non-backbone areas, instead of advertising all routes to specific subnets. As a

result, classless routing is required in routers in the stubby area, because otherwise non-backbone area routers would not be able to forward packets to all parts of the network.

Classless and classful routing logic is controlled by the **ip classless** global configuration command. The **ip classless** command enables classless routing, and the **no ip classless** command enables classful routing.

NOTE Make sure to review Table 8-7 in Chapter 8, "RIP Version 2," for a comparison of classful addresses, classful routing, and classful routing protocols.

Multilayer Switching

Multilayer Switching (MLS) refers to the process by which a LAN switch, which operates at least at Layer 2, also uses logic and protocols from layers other than Layer 2 to forward data. The term *Layer 3 switching* refers specifically to the use of the Layer 3 destination address, compared to the routing table (or equivalent), to make the forwarding decision. (The latest switch hardware and software from Cisco uses CEF switching to optimize the forwarding of packets at Layer 3.)

MLS Logic

Layer 3 switching configuration works similarly to router configuration—IP addresses are assigned to interfaces, and routing protocols are defined. The routing protocol configuration works just like a router; however, the interface configuration on MLS switches differs slightly from routers, using VLAN interfaces, routed interfaces, and PortChannel interfaces.

VLAN interfaces give a Layer 3 switch a Layer 3 interface attached to a VLAN. Cisco sometimes refers to these interfaces as *switched virtual interfaces (SVIs)*. To route between VLANs, a switch simply needs a virtual interface attached to each VLAN, and each VLAN interface needs an IP address in the respective subnets used on those VLANs.

NOTE Although it is not a requirement, the devices in a VLAN are typically configured in the same single IP subnet. However, you can use secondary IP addresses on VLAN interfaces to configure multiple subnets in one VLAN, just like on other router interfaces.

KEY POINT When using VLAN interfaces, the switch must take one noticeable but simple additional step when routing a packet. Like typical routers, MLS makes a routing decision to forward a packet. As with routers, the routes in an MLS routing table entry list an outgoing interface (a VLAN interface in this case), as well as a next-hop layer 3 address. The adjacency information (for example,

the IP ARP table or the CEF adjacency table) lists the VLAN number and the next-hop device's MAC address to which the packet should be forwarded—again, typical of normal router operation.

At this point, a true router would know everything it needs to know to forward the packet. An MLS switch, however, then also needs to use Layer 2 logic to decide out which physical interface to physically forward the packet. The switch will simply find the next-hop device's MAC address in the CAM and forward the frame to that address based on the CAM.

Using Routed Ports and PortChannels with MLS

In some point-to-point topologies, VLAN interfaces are not required. For instance, when an MLS switch connects to a router using a cable from a switch interface to a router's LAN interface, and the only two devices in that subnet are the router and that one physical interface on the MLS switch, the MLS switch can be configured to treat that one interface as a *routed port*. (Another typical topology for using router ports is when two MLS switches connect for the purpose of routing between the switches, again creating a case with only two devices in the VLAN/subnet.)

A routed port on an MLS switch has the following characteristics:

- The interface is not in any VLAN (not even VLAN 1).
- The switch does not keep any Layer 2 switching table information for the interface.
- Layer 3 settings, such as the IP address, are configured under the physical interface—just like a router.
- The adjacency table lists the outgoing physical interface or PortChannel, which means that Layer 2 switching logic is not required in these cases.

Ethernet PortChannels can be used as routed interfaces as well. To do so, as on physical routed interfaces, the **no switchport** command should be configured. (For PortChannels, the physical interfaces in the PortChannel must also be configured with the **no switchport** command.) Also, when using a PortChannel as a routed interface, PortChannel load balancing should be based on Layer 3 addresses because the Layer 2 addresses will mostly be the MAC addresses of the two MLS switches on either end of the PortChannel. PortChannels may also be used as Layer 2 interfaces when doing MLS. In that case, VLAN interfaces would be configured with IP address, and the PortChannel would simply act as any other Layer 2 interface.

Table 7-4 lists some of the specifics about each type of Layer 3 interface.

Table 7-4 *MLS Layer 3 Interfaces*

KEY POINT

Interface	Forwarding to Adjacent Device	Configuration Requirements
VLAN interface	Uses Layer 2 logic and L2 MAC address table	Create VLAN interface; VLAN must also exist
Physical (routed) interface	Forwards out physical interface	Use **no switchport** command to create a routed interface
PortChannel (switched) interface	Not applicable; just used as another Layer 2 forwarding path	No special configuration; useful in conjunction with VLAN interfaces
PortChannel (routed) interface	Balances across links in PortChannel	Needs **no switchport** command in order to be used as a routed interface; optionally change load-balancing method

MLS Configuration

The upcoming MLS configuration example is designed to show all of the configuration options. The network design is shown in Figures 7-4 and 7-5. In Figure 7-4, the physical topology is shown, with routed ports, VLAN trunks, a routed PortChannel, and access links. Figure 7-5 shows the same network, with a Layer 3 view of the subnets used in the network.

Figure 7-4 *Physical Topology: Example Using MLS*

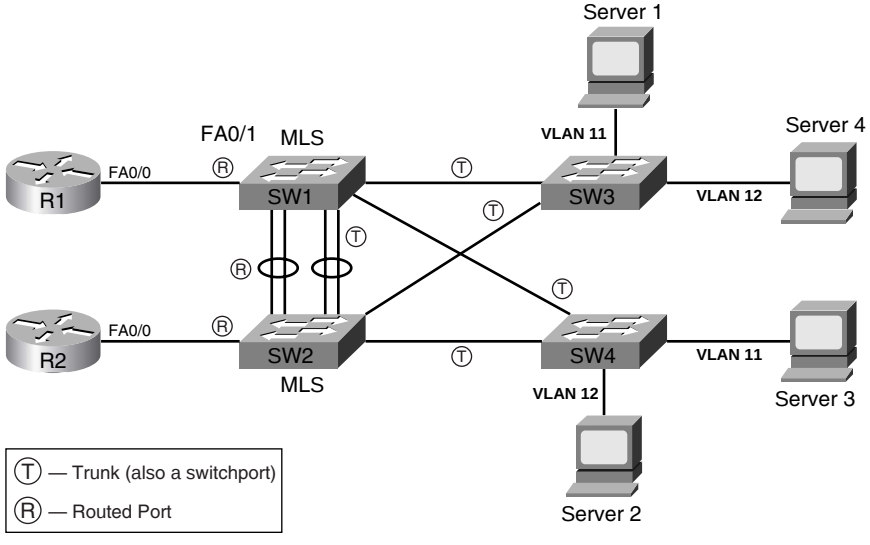
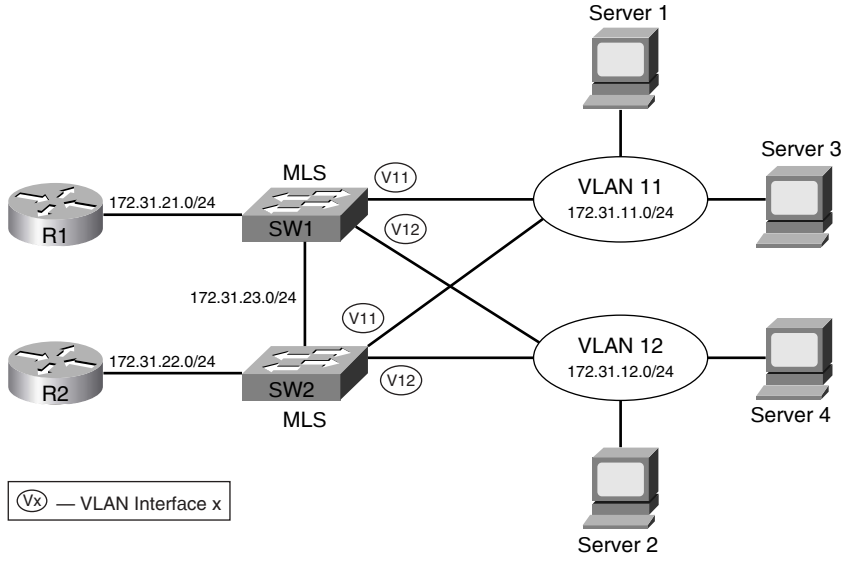


Figure 7-5 *Layer 3 Topology View: Example Using MLS*



A few design points bear discussion before jumping into the configuration. First, SW1 and SW2 need Layer 2 connectivity to support traffic in VLANs 11 and 12. In other words, you need a Layer 2 trunk between SW1 and SW2, and for several reasons. Focusing on the Layer 2 portions of the network on the right side of Figure 7-4, SW1 and SW2, both distribution MLS switches, connect to SW3 and SW4, which are access layer switches. SW1 and SW2 are responsible for providing full connectivity in VLANs 11 and 12. To fully take advantage of the redundant links, SW1 and SW2 need a Layer 2 path between each other. Additionally, this design uses SW1 and SW2 as Layer 3 switches, so the hosts in VLANs 11 and 12 will use SW1 or SW2 as their default gateway. For better availability, the two switches should use HSRP, VRRP, or GLBP. Regardless of which protocol is used, both SW1 and SW2 need to be in VLANs 11 and 12, with connectivity in those VLANs, to be effective as default gateways.

In addition to a Layer 2 trunk between SW1 and SW2, to provide effective routing, it makes sense for SW1 and SW2 to have a routed path between each other as well. Certainly, SW1 needs to be able to route packets to router R1, and SW2 needs to be able to route packets to router R2. However, routing between SW1 and SW2 allows for easy convergence if R1 or R2 fails.

Figure 7-4 shows two alternatives for routed connectivity between SW1 and SW2, and one option for Layer 2 connectivity. For Layer 2 connectivity, a VLAN trunk needs to be used between the two switches. Figure 7-4 shows a pair of trunks between SW1 and SW2 (labeled with a circled T) as a Layer 2 PortChannel. The PortChannel would support the VLAN 11 and 12 traffic.

To support routed traffic, the figure shows two alternatives: simply route over the Layer 2 PortChannel using VLAN interfaces, or use a separate routed PortChannel. First, to use the Layer 2

PortChannel, SW1 and SW2 could simply configure VLAN interfaces in VLANs 11 and 12. The alternative configuration uses a second PortChannel that will be used as a routed PortChannel. However, the routed PortChannel does not function as a Layer 2 path between the switches, so the original Layer 2 PortChannel must still be used for Layer 2 connectivity. Upcoming Example 7-3 shows both configurations.

Finally, a quick comment about PortChannels is needed. This design uses PortChannels between the switches, but they are not required. Most links between switches today use at least two links in a PortChannel, for the typical reasons—better availability, better convergence, and less STP overhead. This design includes the PortChannel to point out a small difference between the routed interface configuration and the routed PortChannel configuration.

Example 7-3 shows the configuration for SW1, with some details on SW2.

Example 7-3 *MLS-Related Configuration on Switch1*

```
! Below, note that the switch is in VTP transparent mode, and VLANs 11 and 12 are
! configured, as required. Also note the ip routing global command, without which
! the switch will not perform Layer 3 switching of IP packets.
vlan 11
!
vlan 12
! The ip routing global command is required before the MLS will perform
! Layer 3 forwarding.
ip routing
!
vtp domain CCIE-domain
vtp mode transparent
! Next the no switchport command makes PortChannel a routed port. On a routed
! port, an IP address can be added to the interface.
interface Port-channel1
 no switchport
 ip address 172.31.23.201 255.255.255.0
! Below, similar configuration on the interface connected to Router1.
interface FastEthernet0/1
 no switchport
 ip address 172.31.21.201 255.255.255.0
! Next, the configuration shows basic PortChannel commands, with the
! no switchport command being required due to the same command on PortChannel.
interface GigabitEthernet0/1
 no switchport
 no ip address
 channel-group 1 mode desirable
!
interface GigabitEthernet0/2
 no switchport
 no ip address
```

Example 7-3 *MLS-Related Configuration on Switch1 (Continued)*

```
 channel-group 1 mode desirable
! Next, interface VLAN 11 gives Switch1 an IP presence in VLAN11. Devices in VLAN
! 11 can use 172.31.11.201 as their default gateway. However, using HSRP is
! better, so Switch1 has been configured to be HSRP primary in VLAN11, and Switch2
! to be primary in VLAN12, with tracking so that if Switch1 loses its connection
! to Router1, HSRP will fail over to Switch2.
interface Vlan11
 ip address 172.31.11.201 255.255.255.0
 no ip redirects
 standby 11 ip 172.31.11.254
 standby 11 priority 90
 standby 11 track FastEthernet0/1
! Below, VLAN12 has similar configuration settings, but with a higher (worse)
! HSRP priority than Switch2's VLAN 12 interface.
interface Vlan12
 ip address 172.31.12.201 255.255.255.0
 no ip redirects
 standby 12 ip 172.31.12.254
 standby 12 priority 110
 standby 12 track FastEthernet0/1
```

NOTE For MLS switches to route using VLAN interfaces, two other actions are required: The corresponding VLANs must be created, and the **ip routing** global command must have been configured. (MLS switches will not perform Layer 3 routing without the **ip routing** command, which is not enabled by default.) If the VLAN interface is created before either of those actions, the VLAN interface sits in a "down and down" state. If the VLAN is created next, the VLAN interface is in an "up and down" state. Finally, after adding the **ip routing** command, the interface is in an "up and up" state.

As stated earlier, the routed PortChannel is not required in this topology. It was included to show an example of the configuration, and to provide a backdrop from which to discuss the differences. However, as configured, SW1 and SW2 are Layer 3 adjacent over the routed PortChannel as well as via their VLAN 11 and 12 interfaces. So, they could exchange IGP routing updates over three separate subnets. In such a design, the routed PortChannel was probably added so that it would be the normal Layer 3 path between SW1 and SW2; care should be taken to tune the IGP implementation so that this route is chosen instead of the routes over the VLAN interfaces.

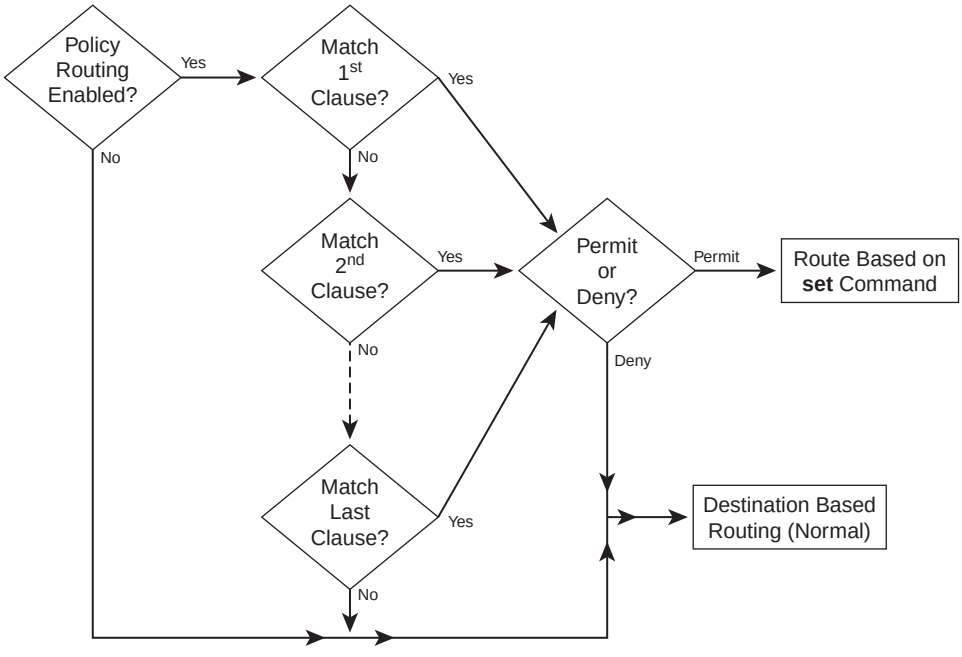
Policy Routing

All the options for IP forwarding (routing) in this chapter had one thing in common: The destination IP address in the packet header was the only thing in the packet that was used to determine how the packet was forwarded. Policy routing allows a router to make routing decisions based on information besides the destination IP address.

Policy routing's logic begins with the **ip policy** command on an interface. This command tells IOS to process incoming packets with different logic before the normal forwarding logic takes place. (To be specific, policy routing intercepts the packet after Step 2, but before Step 3, in the routing process shown in Figure 7-1.) IOS compares the received packets using the **route map** referenced in the **ip policy** command. Figure 7-6 shows the basic logic.

Figure 7-6 *Basic Policy Routing Logic*

KEY POINT



Specifying the matching criteria for policy routing is relatively simple compared to defining the routing instructions using the **set** command. The route maps used by policy routing must match either based on referring to an ACL (numbered or named IP ACL, using the **match ip address** command) or based on packet length (using the **match length** command). To specify the routing instructions—in other words, where to forward the packet next—the **set** command is used. Table 7-5 lists the **set** commands, and provides some insight into their differences.

Table 7-5 *Policy Routing Instructions (***set** *Commands)*

Command	Comments
set ip next-hop *ip-address* [. . . *ip-address*]	Next-hop addresses must be in a connected subnet; forwards to the first address in the list for which the associated interface is up.
set ip default next-hop *ip-address* [. . . *ip-address*]	Same logic as previous command, except policy routing first attempts to route based on the routing table.

Table 7-5 *Policy Routing Instructions (***set** *Commands) (Continued)*

Command	Comments
set interface *interface-type interface-number* [. . . *interface-type interface-number*]	Forwards packets using the first interface in the list that is up.
set default interface *interface-type interface-number* [. . . *interface-type interface-number*]	Same logic as previous command, except policy routing first attempts to route based on the routing table.
set ip precedence *number* | *name*	Sets IP precedence bits; can be decimal value or ASCII name.
set ip tos [*number*]	Sets entire ToS byte; numeric value is in decimal.

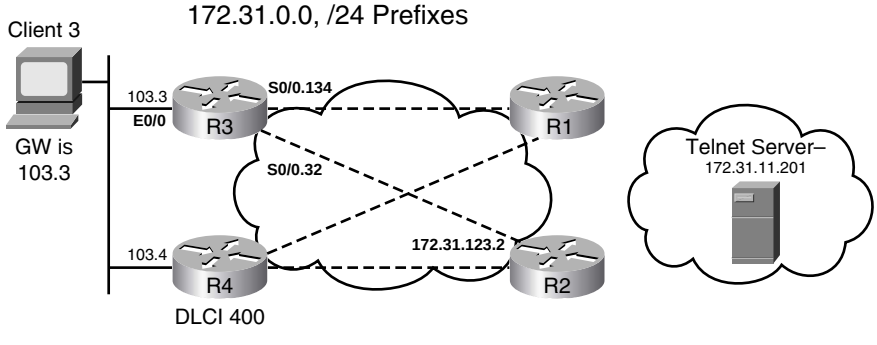
KEY POINT

The first four **set** commands in Table 7-5 are the most important ones to consider. Essentially, you set either the next-hop IP address or the outgoing interface. Use the outgoing interface option only when it is unambiguous—for instance, do not refer to a LAN interface or multipoint Frame Relay subinterface. Most importantly, note the behavior of the **default** keyword in the **set** commands. Use of the **default** keyword essentially means that policy routing tries the default (destination based) routing first, and resorts to using the **set** command details only when the router finds no matching route in the routing table.

The remaining **set** commands set the bits inside the ToS byte of the packet; refer to Chapter 14, "Classification and Marking," for more information about the ToS byte and QoS settings. Note that you can have multiple **set** commands in the same **route-map** clause. For instance, you may want to define the next-hop IP address and mark the packet's ToS at the same time.

Figure 7-7 shows a variation on the same network used earlier in this chapter. Router3 and Router4 are now at the same site, connected to the same LAN, and each has PVCs connecting to Router1 and Router2.

Figure 7-7 *Policy Routing Example on Router3*



Example 7-4 shows three separate policy routing configurations on Router3. The first configuration forwards Telnet traffic over the PVC to Router2 (next hop 172.31.123.2). The next configuration does the same thing, but this time using the **set interface** command. The final option shows a nonworking case with Router3 specifying its LAN interface as an outgoing interface.

Example 7-4 *Policy Routing Example on Router3*

KEY POINT

```
! Below, Router3 is configured with three route maps, one of which is enabled on
! interface e0/0 with the ip policy route-map to-R2-nexthop command. The two
! route maps that are not referenced in the ip policy command are used
! later in the configuration.
Router3# sh run
! Lines omitted for brevity
interface Ethernet0/0
 mac-address 0200.3333.3333
 ip address 172.31.104.3 255.255.255.0
 ip policy route-map to-R2-nexthop
!
interface Serial0/0.32 point-to-point
 ip address 172.31.123.3 255.255.255.0
 frame-relay interface-dlci 200
!
interface Serial0/0.3333 point-to-point
 ip address 172.31.134.3 255.255.255.0
 frame-relay interface-dlci 100
!
access-list 111 permit tcp any any eq telnet
! This route-map matches all telnet, and picks a route through R2.
route-map to-R2-nexthop permit 10
 match ip address 111
 set ip next-hop 172.31.123.2
! This route-map matches all telnet, and picks a route out E0/0.
route-map to-R4-outgoing permit 10
 match ip address 111
 set interface Ethernet0/0
! This route-map matches all telnet, and picks a route out S0/0.32.
route-map to-R2-outgoing permit 10
 match ip address 111
 set interface Serial0/0.32
! debugging is enabled to prove policy routing is working on Router3.
Router3# debug ip policy
Policy routing debugging is on
! Not shown, a Client3 tries to telnet to 172.31.11.201
! Below, a sample of the debug messages created for a single policy-routed packet.
06:21:57: IP: route map to-R2-nexthop, item 10, permit
06:21:57: IP: Ethernet0/0 to Serial0/0.32 172.31.123.2
```

Example 7-4 *Policy Routing Example on Router3 (Continued)*

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Next, Router3 uses a different route-map. This one sets the outgoing interface to
! S0/0.32.  The Outgoing interface option works, because it is a point-to-point
! subinterface
Router3# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router3(config)# int e 0/0
Router3(config-if)# ip policy route-map to-R2-outgoing
Router3(config-if)# ^Z
! Not shown, the same user with default gateway of Router3 tries to telnet again.
! Below, the sample debug messages are identical as the previous set of messages.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
06:40:51: IP: route map to-R2-outgoing, item 10, permit
06:40:51: IP: Ethernet0/0 to Serial0/0.32 172.31.123.2
! Next, switching to a third route-map that sets the outgoing interface to E0/0.
Router3# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router3(config)# int e 0/0
Router3(config-if)# ip policy route-map to-R4-outgoing
Router3(config-if)# ^Z
! Not shown, the same user with default gateway of Router3 tries to telnet again.
! Router3 actually sends an ARP request out e0/0, looking for
! the IP address in the destination of the packet – 172.31.11.201, the address
! to which the user is telnetting.  Also below, Router3 shows that the ARP table
! entry for 172.31.11.201 is incomplete.
Router3# sh ip arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  172.31.11.201           0   Incomplete      ARPA
Internet  172.31.104.3            –   0200.3333.3333  ARPA   Ethernet0/0
Internet  172.31.104.4            0   0200.4444.4444  ARPA   Ethernet0/0
```

The first two route maps in the example were relatively simple, with the last route map showing why specifying a multi-access outgoing interface is problematic. In the first two cases, the telnet works fine; to verify that it was working, the **debug ip policy** command was required.

The third route map (to-R4-outgoing) sets the output interface to Router3's E0/0 interface. Because Router3 does not have an associated next-hop IP address, Router3 sends an ARP request asking for 172.31.11.201's MAC address. As shown in the **show ip arp** command output, Router3 never completes its ARP entry. To work around the problem, assuming that the goal is to forward the packets to Router4 next, the configuration in Router3 should refer to the next-hop IP address instead of the outgoing interface E0/0.

NOTE Policy Routing for this particular topology fails due to a couple of tricky side effects of ARP. At first glance, you might think that the only thing required to make the to-R4-outgoing policy work is for R4 to enable proxy ARP. In fact, if R4 is then configured with an **ip proxy-arp** interface subcommand, R4 does indeed reply to R3's ARP for 172.31.11.201. R4 lists its own MAC address in the ARP reply. However, R3 rejects the ARP reply, because of a basic check performed on ARPs. R3's only IP route matching address 172.31.11.201 points over the WAN interface, and routers check ARP replies to make sure they list a sensible interface. From R3's perspective, the only sensible interface is one through which the destination might possibly be reached. So, R3's logic dictates that it should never hear an ARP reply regarding 172.31.11.201 coming in its fa0/0 interface, so R3 rejects the (proxy) ARP reply from R4. To see all of this working in a lab, re-create the topology, and use the **debug ip arp** and **debug policy** commands.

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most of the Cisco Press *Exam Certification Guides*, this book does not repeat information listed in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in the "Foundation Topics" section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 7-6 lists the protocols mentioned in or pertinent to this chapter and their respective standards documents.

Table 7-6 *Protocols and Standards for Chapter 7*

Name	Standardized In
Address Resolution Protocol (ARP)	RFC 826
Reverse Address Resolution Protocol (RARP)	RFC 903
Frame Relay Inverse ARP (InARP)	RFC 2390
Frame Relay Multiprotocol Encapsulation	RFC 2427
Differentiated Services Code Point (DSCP)	RFC 2474

Table 7-7 lists some of the key IOS commands related to the topics in this chapter. (The command syntax for switch commands was taken from the *Catalyst 3550 Multilayer Switch Command Reference, 12.1(20)EA2*.) Router-specific commands were taken from the IOS 12.3 mainline command reference.)

Table 7-7 *Command Reference for Chapter 7*

Command	Description
[**no**] **ip classless**	Enables classless (**ip classless**) or classful (**no ip classless**) forwarding
show ip arp	EXEC command that displays the contents of the IP ARP cache
show frame-relay map	Router **exec** command that lists the mapping information between Frame Relay DLCIs and Layer 3 addresses
frame-relay interface-dlci	Configuration command that associates a particular DLCI with a subinterface
[**no**] **switchport**	Switch interface subcommand that toggles an interface between a Layer 2 switched function (**switchport**) and a routed port (**no switchport**)

continues

Table 7-7 *Command Reference for Chapter 7 (Continued)*

Command	Description
clear frame-relay inarp	Router **exec** command that clears all InARP-learned entries from the Frame Relay mapping table
[**no**] **ip route-cache cef**	Interface subcommand that enables or disables CEF switching on an interface
[**no**] **ip cef**	Global configuration command to enable (or disable) CEF on all interfaces
debug frame-relay events	Displays messages about various events, including InARP messages
show frame-relay map	Displays information about Layer 3 to Layer 2 mapping with Frame Relay
frame-relay map *protocol protocol-address* {*dlci*} [**broadcast**] [**ietf** | **cisco**]	Interface subcommand that maps a Layer 3 address to a DLCI
[**no**] **frame-relay inverse-arp** [*protocol*] [*dlci*]	Interface subcommand that enables or disables InARP
[**no**] **ip routing**	Enables IP routing; defaults to **no ip routing** on a multilayer switch
ip policy route-map *map-tag*	Router interface subcommand that enables policy routing for the packets entering the interface

Refer to Table 7-5 for the list of **set** commands related to policy routing.

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

policy routing, process switching, CEF, MLS, ARP, proxy ARP, routed interface, InARP, fast switching, TTL, classless routing, classful routing, FIB, adjacency table, control plane, switched interface, data plane, IP routing, IP forwarding

Refer to the CD-based glossary to check your answers.

Further Reading

For a good reference on load balancing with CEF, refer to http://cisco.com/en/US/partner/tech/tk827/tk831/technologies_tech_note09186a0080094806.shtml. This website requires a CCO account.

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- IP Routing
  - — RIP Version 2
  - — The use of **show** and **debug** commands

CHAPTER 8

RIP Version 2

Chapters 8 through 11 each focus on a single routing protocol. This chapter covers Routing Information Protocol (RIP) Version 2, including most of the features, concepts, and commands. Chapter 11, "IGP Route Redistribution, Route Summarization, and Default Routing," covers some RIP details, in particular, route redistribution between RIP and other routing protocols, and route summarization.

"Do I Know This Already?" Quiz

Table 8-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 8-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
RIP Version 2 Basics	1–2	
RIP Convergence and Loop Prevention	3–5	
RIP Configuration	6–8	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. Which of the following items are true of RIP Version 2?

- **a.** Supports VLSM
- **b.** Sends Hellos to 224.0.0.9
- **c.** Allows for route tagging
- **d.** Defines infinity as 255 hops
- **e.** Authentication requires 3DES

2. In an internetwork that solely uses RIP, once the network is stable and converged, which of the following is true?

   a. Routers send RIP updates every 30 seconds.

   b. Routers send RIP updates every 90 seconds.

   c. Routers send Hellos every 10 seconds, and send updates only when routes change.

   d. A routing update sent out a router's s0/0 interface includes all RIP routes in the IP routing table.

   e. A RIP update's routes list the same metric as is shown in that router's IP routing table.

3. R1 previously had heard about only one route to 10.1.1.0/24, metric 3, via an update received on its s0/0 interface, so it put that route in its routing table. R1 gets an update from that same neighboring router, but the same route now has metric 16. R1 immediately sends a RIP update out s0/0 that advertises a metric 16 route for that same subnet. Which of the following are true for this scenario?

   a. Split horizon must have been disabled on R1's s0/0 interface.

   b. R1's update is a triggered update.

   c. R1's metric 16 route advertisement is an example of a poison reverse route.

   d. The incoming metric 16 route was the result of a counting-to-infinity problem.

4. R1 is in a network that uses RIPv2 exclusively, and RIP has learned dozens of subnets via several neighbors. Which of the following commands display the current value of at least one route's Invalid timer?

   a. **show ip route**

   b. **show ip rip database**

   c. **debug ip rip**

   d. **debug ip rip event**

5. R1 is in a network that uses RIPv2 exclusively, and RIP has learned dozens of subnets via several neighbors. From privileged exec mode, the network engineer types in the command **clear ip route**. What happens?

   a. R1 removes all routes from its IP routing table.

   b. R1 removes only RIP routes from its IP routing table.

   c. After the command, R1 will relearn its routes when the neighboring router's Update timers cause them to send their next updates.

   d. R1 immediately sends updates on all interfaces, poisoning all routes, so that all neighbors immediately send triggered updates—which allow R1 to immediately relearn its routes.

e. R1 will relearn its routes immediately by sending RIP requests out all its interfaces.

f. None of the other answers is correct.

6. R1 has learned a route to 10.1.1.0/24 using RIPv2, which is R1's only working routing protocol. Which of the following addresses point to information that definitively identifies the IP address of the neighboring router from which R1 learned this route?

   a. The "via" IP address listed in a route in the **show ip route** command output

   b. The "from" IP address listed for a route in the **show ip route 10.1.1.0** command output

   c. The "via" IP address listed in a route in the **show ip rip database** command output

   d. The source IP address listed in the debug heading message, before the detailed output listing each route, in messages from the **debug ip rip** command

7. R1 has been configured for RIPv2, including a **network 10.0.0.0** command. Which of the following statements are true about R1's RIP behavior?

   a. R1 will send advertisements out any of its interfaces in network 10.

   b. R1 will process received advertisements in any of its interfaces in network 10.

   c. R1 will send updates only after receiving a RIP Hello message from a neighboring router.

   d. R1 can disable the sending of routing updates on an interface using the **passive-interface** interface subcommand.

   e. R1 will advertise the subnets of any of its interfaces connected to subnets of network 10.

8. Which of the following represents a default setting for the Cisco IOS implementation of RIPv2?

   a. Split horizon is enabled on all types of interfaces.

   b. Split horizon is disabled on Frame Relay physical interfaces and multipoint subinterfaces.

   c. The default authentication mode, normally set with the **ip rip authentication mode** interface subcommand, is MD5 authentication.

   d. RIP will send triggered updates when a route changes.

Foundation Topics

RIP Version 2 Basics

RIP is the only routing protocol covered on the CCIE Routing and Switching exam that is not also covered on the CCNP exams. Although covered on the CCNA exams, in years past, RIPv2 was not part of CCNA, either. So, while many CCIE candidates might already know many of the features and configuration options of RIP, many CCNPs have never really had to study or use RIPv2 to any great extent in order to pass any Cisco exams. This chapter summarizes the protocol features and gives specific examples of many RIPv2 concepts and configuration options.

Note that the Routing and Switching blueprint specifically lists RIPv2 but does not list RIPv1. Regardless, just for completeness, some of the coverage in this chapter mentions the differences between RIP Versions 1 and 2. Table 8-2 lists and briefly describes the RIP features, noting with an asterisk those features that are found only in RIPv2.

Table 8-2 *RIP Feature Summary*

KEY POINT

Function	Description
Transport	UDP, port 520.
Metric	Hop count, with 15 as the maximum usable metric, and 16 considered to be infinite.
Hello interval	None; RIP relies on the regular full routing updates instead.
Update destination	Local subnet broadcast (255.255.255.255) for RIPv1; 224.0.0.9 multicast for RIPv2.
Update interval	30 seconds.
Full or partial updates	Full updates each interval. For on-demand circuits, allows RIP to send full updates once, and then remain silent until changes occur, per RFC 2091. Full updates each interval.
Triggered updates	Yes, when routes change.
Multiple routes to the same subnet	Allows installing 1 to 6 (default 4) equal-metric routes to the same subnet in a single routing table.
Authentication*	Allows both plain-text and MD5 authentication.
Subnet mask in updates*	RIPv2 transmits the subnet mask with each route, thereby supporting VLSM, making RIPv2 classless. This feature also allows RIPv2 to support discontiguous networks.
VLSM*	Supported as a result of the inclusion of subnet masks in the routing updates.

Table 8-2 *RIP Feature Summary (Continued)*

Function	Description
Route Tags*	Allows RIP to tag routes as they are redistributed into RIP.
Next Hop field*	Supports the assignment of a next-hop IP address for a route, allowing a router to advertise a next-hop router that is different from itself.

* RIPv2-only features

RIP exchanges routes by sending RIP updates on each interface based on an Update timer (update interval). A RIP router advertises its connected routes, as well as other RIP-learned routes that are in the router's IP routing table. Note that RIP does not keep a separate topology table. RIP routers do not form neighbor relationships, nor do they use a Hello protocol—each router simply sends updates, with destination address 224.0.0.9. (Note: RIPv1 uses broadcast address 255.255.255.255.)

RIPv2 uses the same hop-count metric as RIPv1, with 15 being the largest valid metric, and 16 considered to be infinity. Interestingly, a RIP router does not put its own metric in the route of a sent routing update; rather, it first adds 1 to each metric when building the update. For instance, if RouterA has a route with metric 2, it advertises that route with metric 3—in effect, telling the receiving router what its metric should be.

When Cisco RIP routers learn multiple routes to the same subnet, the lowest-metric route is chosen, of course. If multiple equal-hop routes exist, the router (by default) installs up to 4 such routes in its routing table, or between 1 and 6 of such routes, based on the **ip maximum-paths** *number* command under the **router rip** command.

RIP Convergence and Loop Prevention

The most interesting and complicated part of RIP relates to loop-prevention methods used during convergence after a route has failed. Some protocols, like OSPF, IS-IS, and EIGRP, include loop prevention as a side effect of their underlying route computations. However, RIP, like other distance vector protocols, uses several loop-prevention tools. Unfortunately, these loop-prevention tools also significantly increase convergence time—a fact that is certainly the biggest negative feature of RIP, even for RIPv2. Table 8-3 summarizes some of the key features and terms related to RIP convergence, with further explanations following the table.

Table 8-3 *RIP Features Related to Convergence and Loop Prevention*

KEY POINT

Function	Description
Split horizon	Instead of advertising all routes out a particular interface, RIP omits the routes whose outgoing interface field matches the interface out which the update would be sent.
Triggered update	The immediate sending of a new update when routing information changes, instead of waiting for the Update timer to expire.

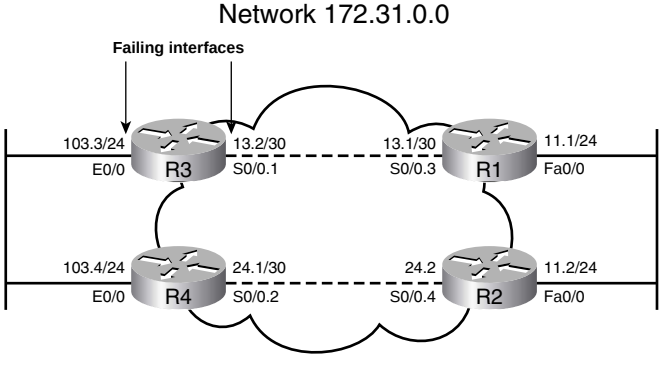
continues

Table 8-3 *RIP Features Related to Convergence and Loop Prevention (Continued)*

Function	Description
Route poisoning	The process of sending an infinite-metric (hop count 16) route in routing updates when that route fails.
Poison reverse	The act of advertising a poisoned route (metric 16) out an interface, but in reaction to receiving that same poisoned route in an update received on that same interface.
Update timer	The timer that specifies the time interval over which updates are sent. Each interface uses an independent timer, defaulting to 30 seconds.
Holddown timer	A per-route timer (default 180 seconds) that begins when a route's metric changes to a larger value. The router does not add an alternative route for this subnet to its routing table until the Holddown timer for that route expires.
Invalid timer	A per-route timer that increases until it receives a routing update that confirms the route is still valid, upon which the timer is reset to 0. If the updates cease, the Invalid timer will grow until it reaches the timer setting (default 180 seconds), after which the route is considered invalid.
Flush (Garbage) timer	A per-route timer that is reset and grows with the Invalid timer. When the Flush timer mark is reached (default 240 seconds), the router removes the route from the routing table and accepts new routes to the failed subnet.

The rest of this section shows examples of the convergence features, using RIP **show** and **debug** command output to show examples of their use. Figure 8-1 shows the sample internetwork that is used in these examples of the various loop-prevention tools.

Figure 8-1 *Sample Internetwork Used for Loop-Prevention Examples*



Converged Steady-State Operation

Example 8-1 shows a few details of R1's operation while all interfaces in Figure 8-1 are up and working. The example lists the basic (and identical) RIP configuration on all four routers; configuration will be covered in more detail later in the chapter. As configured, all four routers are

using only RIPv2, on all interfaces shown in Figure 8-1. Read the comments in Example 8-1 for explanations of the output.

Example 8-1 *Steady-State RIP Operation in Figure 8-1*

```
! All routers use the same three lines of RIP configuration.
router rip
 network 172.31.0.0
 version 2
! Below, the show ip protocol command lists many of RIP's operational settings,
! including RIP timers, version used, and neighbors from which RIP updates have
! been received (listed as "Routing Information Sources").
R1# show ip protocol
Routing Protocol is "rip"
  Sending updates every 30 seconds, next due in 24 seconds
  Invalid after 180 seconds, hold down 180, flushed after 240
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Redistributing: rip
  Default version control: send version 2, receive version 2
    Interface             Send  Recv  Triggered RIP  Key-chain
    FastEthernet0/0       2     2
    Serial0/0.3           2     2
  Automatic network summarization is in effect
  Maximum path: 4
  Routing for Networks:
    172.31.0.0
  Routing Information Sources:
    Gateway         Distance      Last Update
    172.31.11.2           120      00:00:15
    172.31.13.2           120      00:00:08
  Distance: (default is 120)
! Below, the current Invalid timer is listed by each RIP route. Note that it took
! about 3 seconds between the above show ip protocols command and the upcoming
! show ip route command, so the last update from 172.31.13.2 (above)
! was 8 seconds; 3 seconds later, the Invalid timer for a route learned from
! 172.31.13.2 is now 11 seconds.
R1# 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
```

continues

Example 8-1 *Steady-State RIP Operation in Figure 8-1 (Continued)*

```
Gateway of last resort is not set

     172.31.0.0/16 is variably subnetted, 4 subnets, 2 masks
R       172.31.24.0/30 [120/1] via 172.31.11.2, 00:00:18, FastEthernet0/0
C       172.31.11.0/24 is directly connected, FastEthernet0/0
C       172.31.13.0/30 is directly connected, Serial0/0.3
R       172.31.103.0/24 [120/1] via 172.31.13.2, 00:00:11, Serial0/0.3
! Below, the show ip rip database command lists information for each route
! considered by RIP.
R1# show ip rip database
172.31.0.0/16    auto-summary
172.31.11.0/24    directly connected, FastEthernet0/0
172.31.13.0/30    directly connected, Serial0/0.3
172.31.24.0/30
    [1] via 172.31.11.2, 00:00:01, FastEthernet0/0
172.31.103.0/24
[1] via 172.31.13.2, 00:00:23, Serial0/0.3
```

NOTE The **show ip rip database** command lists all RIP learned routes, and all connected routes that RIP is advertising.

Triggered (Flash) Updates and Poisoned Routes

When RIP knows for sure that a route to a subnet has failed, RIPv2 can converge to an alternate route typically in less than a minute. Example 8-2 details the steps behind one such example, using Figure 8-1, with the steps outlined in the following list (the comments in Example 8-2 refer to these steps by number):

1. RIP **debug** messages show R1's RIP updates, including R1's use of split horizon.
2. R3's E0/0 interface is shut down, simulating a failure.
3. R3 immediately sends a triggered update (also called a flash update), because R3 knows for sure that the route has failed. R3's advertised route is a poisoned route to 172.31.103.0/24.
4. R1 immediately (due to triggered updates) advertises a poison reverse route for 172.31.103.0/24, back to R3, and sends a triggered update out its fa0/0 interface.
5. R1 removes its route to 172.31.103.0/24 from its routing table.
6. R1 waits for R2's next update, sent based on R2's Update timer on its fa0/0 interface. That update includes a route to 172.31.103.0/24. R1 adds that route to its routing table.

Example 8-2 *R1's Convergence for 172.31.103.0/24 upon R3's E0/0 Interface Failure*

```
! First, the debug ip rip command enables RIP debugging. This command will show
! messages that show every route in the sent and received updates.
R1# debug ip rip
RIP protocol debugging is on
! (Step 1) Below, the output exhibits split horizon—for example, 172.31.103.0/24
! is not advertised out s0/0.3, but it is advertised out fa0/0.
*Mar  3 22:44:08.176: RIP: sending v2 update to 224.0.0.9 via Serial0/0.3 (172.31.13.1)
*Mar  3 22:44:08.176: RIP: build update entries
*Mar  3 22:44:08.176:  172.31.11.0/24 via 0.0.0.0, metric 1, tag 0
*Mar  3 22:44:08.176:  172.31.24.0/30 via 0.0.0.0, metric 2, tag 0
*Mar  3 22:44:12.575: RIP: sending v2 update to 224.0.0.9 via FastEthernet0/0 (172.31.11.1)
*Mar  3 22:44:12.575: RIP: build update entries
*Mar  3 22:44:12.575:  172.31.13.0/30 via 0.0.0.0, metric 1, tag 0
*Mar  3 22:44:12.575:  172.31.103.0/24 via 0.0.0.0, metric 2, tag 0
! Next, R1 receives a RIP update from R3. The metric 1 route in the update below
! is R1's best route, and is placed into R1's routing table. Note that the metric
! in the received update is R1's actual metric to reach the route.
*Mar  3 22:44:21.265: RIP: received v2 update from 172.31.13.2 on Serial0/0.3
*Mar  3 22:44:21.269:      172.31.24.0/30 via 0.0.0.0 in 2 hops
*Mar  3 22:44:21.269:      172.31.103.0/24 via 0.0.0.0 in 1 hops
! (Step 2) R3's E0/0 interface is shut down at this point. (Not shown).
! (Step 3) Below, R1 receives a triggered update, with two poison routes from R3—
! the same two routes that R3 advertised in the previous routing update above.
! Note that the triggered update only includes changed routes, with full updates
! continuing on the same update interval.
*Mar  3 22:44:46.338: RIP: received v2 update from 172.31.13.2 on Serial0/0.3
*Mar  3 22:44:46.338:      172.31.24.0/30 via 0.0.0.0 in 16 hops  (inaccessible)
*Mar  3 22:44:46.338:      172.31.103.0/24 via 0.0.0.0 in 16 hops  (inaccessible)
! (Step 4) Above, R1 reacts to its receipt of poisoned routes, sending a triggered
! update out its fa0/0 interface. Note that the debug refers to the triggered
! update as a flash update.
*Mar  3 22:44:48.341: RIP: sending v2 flash update to 224.0.0.9 via FastEthernet 0/0
  (172.31.11.1)
*Mar  3 22:44:48.341: RIP: build flash update entries
*Mar  3 22:44:48.341:  172.31.103.0/24 via 0.0.0.0, metric 16, tag 0
! (Step 4) R1 also sends a triggered update out s0/0.3 to R3, which includes
! a poison reverse route to 172.31.103.0/24, back to R3. R1 does not send back a
! poison route to 172.31.24.0, because R1's route to 172.31.24.0 was
! pointing towards R2, not R3—so R1's route to 172.31.24.0/24 did not fail.
*Mar  3 22:44:48.341: RIP: sending v2 flash update to 224.0.0.9 via Serial0/0.3
(172.31.13.1)
! (Step 5) Below, note the absence of a route to 103.0/24 in R1's routing table.
R1# show ip route 172.31.103.0
% Subnet not in table
! (Step 6) Below, 23 seconds since the previous debug message, R2's next routing
! update arrives at R1, advertising 172.31.103.0/24. Following that, R1 now has
! a 2-hop route, through R2, to 172.31.103.0/24.
```

continues

Example 8-2 *R1's Convergence for 172.31.103.0/24 upon R3's E0/0 Interface Failure (Continued)*

```
*Mar  3 22:45:11.271: RIP: received v2 update from 172.31.11.2 on FastEthernet0/0
*Mar  3 22:45:11.271:      172.31.24.0/30 via 0.0.0.0 in 1 hops
*Mar  3 22:45:11.271:      172.31.103.0/24 via 0.0.0.0 in 2 hops
R1# show ip route 172.31.103.0
Routing entry for 172.31.103.0/24
  Known via "rip", distance 120, metric 2
  Redistributing via rip
  Last update from 172.31.11.2 on FastEthernet0/0, 00:00:01 ago
  Routing Descriptor Blocks:
  * 172.31.11.2, from 172.31.11.2, 00:00:01 ago, via FastEthernet0/0
Route metric is 2, traffic share count is 1
```

If you examine the **debug** message time stamps in Example 8-2, you will see that between 25 and 45 seconds passed from when R1 heard the poisoned routes until R1 heard R2's new routing update with a now-best route to 172.31.103.0/24. While not on par with EIGRP or OSPF, this convergence is reasonably fast for RIP.

NOTE Do not confuse the term *triggered update* with the term *triggered extensions to RIP.* RFC 2091 defines how RIP can choose to send full updates only once, and then be silent, to support demand circuits. The feature is enabled per interface by the **ip rip triggered** interface subcommand.

RIP Convergence When Routing Updates Cease

When a router ceases to receive routing updates, RIP must wait for some timers to expire before it decides that routes previously learned from the now-silent router can be considered to be failed routes. To deal with such cases, RIP uses its Invalid, Flush, and Holddown timers to prevent loops. Coincidentally, RIP's convergence time increases to several minutes as a result.

Example 8-3 details just such a case, where R1 simply ceases to hear RIP updates from R3. (To create the failure, R3's s0/0.1 subinterface was shut down, simulating failure of a Frame Relay PVC.) The example uses the internetwork illustrated in Figure 8-1 again, and begins with all interfaces up, and all four routes known in each of the four routers. The example follows this sequence (the comments in Example 8-3 refer to these steps by number):

1. R3's s0/0.1 subinterface fails, but R1's Frame Relay subinterface stays up—so R1 must use its timers to detect route failures.
2. R1's Invalid and Flush timers for route 172.31.103.0/24 grow because it does not hear any further updates from R3.
3. After the Invalid timer expires (180 seconds) for R1's route to 172.31.103.0/24, R1 begins a Holddown timer for the route. Holddown starts at (default) 180 seconds, and counts down.

4. The Flush timer expires after a total 240 seconds, or 60 seconds past the Invalid timer. As a result, R1 flushes the route to 172.31.103.0/24 from its routing table, which also removes the Holddown timer for the route.

Example 8-3 *R1 Ceases to Hear R3's Updates: Invalid, Flush, and Holddown Timers Required*

```
! First, the debug ip rip event command is used, which displays messages when
! updates are sent and received, but does not display the contents of the updates.
R1# debug ip rip event
RIP event debugging is on
! (Step 1) Not Shown: R3's S0/0.1 subinterface is shut down.
! (Step 2) Below, the Invalid timer for 172.31.103.0/24 has reached 35, meaning
! that 35 seconds have passed since the last received update from which this route
! was learned. An Invalid timer over 30 seconds means that at least one RIP
! update was not received.
R1# 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

    172.31.0.0/16 is variably subnetted, 4 subnets, 2 masks
R       172.31.24.0/30 [120/1] via 172.31.11.2, 00:00:09, FastEthernet0/0
C       172.31.11.0/24 is directly connected, FastEthernet0/0
C       172.31.13.0/30 is directly connected, Serial0/0.3
R       172.31.103.0/24 [120/1] via 172.31.13.2, 00:00:35, Serial0/0.3
! Below, one example set of debug messages are shown. (Many more debug messages
! occurred while waiting for convergence, but those were omitted.) The messages
! about R1's received updates from R2 occur every 30 seconds or so. The contents
! include a 2-hop route to 172.31.103.0/24, which R1 ignores until the Flush timer
! expires.
*Mar  3 21:59:58.921: RIP: received v2 update from 172.31.11.2 on FastEthernet0/0
*Mar  3 21:59:58.921: RIP: Update contains 2 routes
! (Step 3) Below, the Invalid timer expires, roughly 3 minutes after the failure.
! Note that the route is listed as "possibly down," which occurs when the
! Invalid timer has expired but the Flush timer has not. Note that the show ip
! route command does not list the Flush timer settings, but the upcoming show
! ip route 172.31.103.0 command does.
R1# 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
```

continues

Example 8-3 *R1 Ceases to Hear R3's Updates: Invalid, Flush, and Holddown Timers Required (Continued)*

```
       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

        172.31.0.0/16 is variably subnetted, 4 subnets, 2 masks
R       172.31.24.0/30 [120/1] via 172.31.11.2, 00:00:20, FastEthernet0/0
C       172.31.11.0/24 is directly connected, FastEthernet0/0
C       172.31.13.0/30 is directly connected, Serial0/0.3
R       172.31.103.0/24 is possibly down,
          routing via 172.31.13.2, Serial0/0.3
! (Step 3) Next, the command shows the metric as inaccessible, meaning an
! infinite metric, as well as the current Flush timer (3:23), which counts up.
! Also, the Holddown timer for this route has started (at 180 seconds), with 159
! seconds in its countdown. The Holddown timer prevents R1 from using the route
! heard from R2.

R1# show ip route 172.31.103.0
Routing entry for 172.31.103.0/24
  Known via "rip", distance 120, metric 4294967295 (inaccessible)
  Redistributing via rip
  Last update from 172.31.13.2 on Serial0/0.3, 00:03:23 ago
  Hold down timer expires in 159 secs
! (Step 4) Below, just after 4 minutes has passed, the Flush timer has expired,
! and the route to 172.31.103.0/24 has been flushed from the routing table.
R1# show ip route 172.31.103.0
% Subnet not in table
```

At the end of the example, the only remaining step for convergence is for R1 to receive R2's next regular full routing update, which includes a two-hop route to 172.31.103.0/24. R2 will send that update based on R2's regular update interval. R1 would place that route in its routing table, completing convergence.

Note that either the Flush timer or the Holddown timer must expire before new routing information would be used in this case. Here, the Flush timer for route 172.31.103.0/24 expired first, resulting in the route being removed from R1's routing table. When the route is flushed (removed), any associated timers are also removed, including the Holddown timer. Had the Holddown timer been smaller, and had it expired before the Flush timer, R1 would have been able to use the route advertised by R2 at that point in time.

Convergence Extras

Convergence in Example 8-3 took a little over 4 minutes, but it could be improved in some cases. The RIP timers can be tuned with the **timers basic** *update invalid hold-down flush* subcommand

under **router rip**, although care should be taken when changing these timers. The timers should be consistent across routers, and smaller values increase the chance of routing loops being formed during convergence.

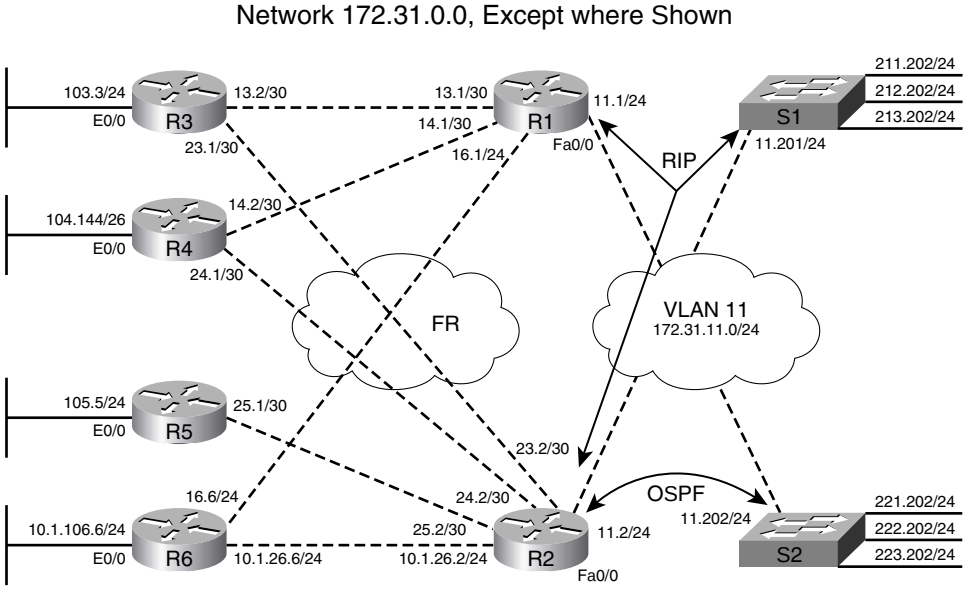
The **clear ip route *** command also speeds convergence by removing all routes from the routing table, along with any per-route timers. In Example 8-3, the **clear ip route 172.31.103.0** command would have worked as well, just deleting that one route. Because the **clear** command bypasses loop-prevention features by deleting the route and timers, it can be risky, but it certainly speeds convergence. Also, after the **clear** command, R1 would immediately issue RIP request packets, which cause the neighboring routers to send full routing updates to R1, instead of waiting on their next update time.

RIP Configuration

The second half of the chapter explains the majority of the options for configuring RIPv2. Make sure to review the full list of commands, and command syntax, listed in Table 8-6 of the "Foundation Summary" section for this chapter.

Figure 8-2 shows the internetwork that will be used to demonstrate RIP configuration throughout the rest of the chapter. Note that most of the subnets are part of network 172.31.0.0, except where noted.

Figure 8-2 *Sample Internetwork Used for RIP Configuration Examples*



The next three examples explain the configuration for the network in Figure 8-2. Before gathering the output for the three examples, all the routers were configured based on the following design goals:

- Only RIPv2 is used, except as noted.
- MD5 authentication is used between S1, R1, and R2 on the LAN. (Note: S1 will be intentionally misconfigured.)
- Simple text authentication is used between R1 and R4.
- S2 speaks only OSPF; only R2 will perform redistribution from OSPF into RIP, and advertise over the LAN, to demonstrate RIP's next-hop feature.
- R5's IP address on the Frame Relay cloud will be configured on the physical serial interface, not a subinterface, to show the effect of the default behavior of **no ip split-horizon**.

Enabling RIP and the Effects of Autosummarization

Example 8-4 covers basic RIP configuration, the meaning and implication of the RIP **network** command, and the effects of the default setting for autosummarization. To examine just those functions, Example 8-4 shows the related RIP configuration on R1, R2, and R6, along with some command output.

Example 8-4 *Basic RIP Configuration on R1, R2, R4, and S1*

```
! First, the three lines of configuration are the same on R1 and S1
! (Point 1): the version 2 command tells R1 to send and receive only RIPv2
! updates, and to ignore RIPv1 updates. The network command must have a classful
! network as the parameter.
router rip
  version 2
  network 172.31.0.0
! Next, the configuration for R2 and R6 is shown, which includes a network 10.0.0.0
! command, enabling RIP on their interfaces in network 10.0.0.0.
router rip
  version 2
  network 10.0.0.0
  network 172.31.0.0
! Below, R1 shows that only v2 updates are being sent and received, and that
! autosummarization is in effect. Note also the Key-chain information for
! authentication, which will be covered in a later example.
R1# sh ip protocol
Routing Protocol is "rip"
  Sending updates every 30 seconds, next due in 26 seconds
  Invalid after 180 seconds, hold down 180, flushed after 240
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
```

Example 8-4 *Basic RIP Configuration on R1, R2, R4, and S1 (Continued)*

```
  Redistributing: rip
  Default version control: send version 2, receive version 2
    Interface             Send  Recv  Triggered RIP  Key-chain
    FastEthernet0/0       2     2                   carkeys
    Serial0/0.3           2     2
    Serial0/0.4           2     2                   anothersetofkeys
    Serial0/0.6           2     2
  Automatic network summarization is in effect
  Maximum path: 4
  Routing for Networks:
! lines omitted for brevity
! Below, the show ip route 10.0.0.0 command lists all of R1's known routes to
! network 10.0.0.0; the only route is for 10.0.0.0/8, because R2 and R6
! automatically summarize (by default) at the classful network boundary.
R1# show ip route 10.0.0.0
Routing entry for 10.0.0.0/8
  Known via "rip", distance 120, metric 1
  Redistributing via rip
  Last update from 172.31.11.2 on FastEthernet0/0, 00:00:01 ago
  Routing Descriptor Blocks:
    172.31.16.6, from 172.31.16.6, 00:00:08 ago, via Serial0/0.6
        Route metric is 1, traffic share count is 1
  * 172.31.11.2, from 172.31.11.2, 00:00:01 ago, via FastEthernet0/0
Route metric is 1, traffic share count is 1
```

A couple of points from this example need a little more explanation. The RIP **network** command only allows for a classful network as a parameter, which in turn enables RIP on all of that router's interfaces that are part of that network. Enabling RIP on an interface makes the router begin sending RIP updates, listening for RIP updates (UDP port 520), and advertising that interface's connected subnet.

Because the RIP **network** command has no way to simply match one interface at a time, a RIP configuration may enable these three functions on an interface for which some or all of these functions are not required. The three RIP functions can be individually disabled on an interface with some effort. Table 8-4 lists these three functions, along with how to disable each feature.

Table 8-4 *RIP Per-Interface Actions, and How to Disable Them Once Enabled*

KEY POINT

RIP Function	How to Disable
Sending RIP updates	Make the interface passive: configure **router rip**, followed by **passive-interface** *type number*
Listening for RIP updates	Filter all incoming routes using a distribute list
Advertising the connected subnet	Filter outbound advertisements on other interfaces using distribute lists, filtering an interface's connected subnet

Another way you can limit advertisements on multiaccess networks is to use the **neighbor** *ip-address* RIP subcommand. This command tells RIP to send unicast RIP updates to that neighbor. For instance, when using a multipoint Frame Relay subinterface, there may be four routers reachable using that subinterface. If you want to send RIP updates to only one of them, make the interface passive, and then use the **neighbor** command to cause RIP to send updates, but only to that one neighbor.

RIP uses *autosummarization* at classful network boundaries by default. In Example 8-4, R2 and R6 connect to parts of classful networks 10.0.0.0/8 and network 172.31.0.0/16. Advertisements sent out interfaces in network 172.31.0.0/16 advertise a summarized route of the complete class A network 10.0.0.0/8. In the example, R2 and R6 both advertise a summarized network 10.0.0.0 to R1. As a result, as seen with the **show ip route 10.0.0.0** command on R1, R1 knows two equal-cost routes to classful network 10.0.0.0. In this case, R1 would send some packets meant for subnet 10.1.106.0/24 through R2 first, a seemingly poor choice. To advertise the subnets of network 10.0.0.0, R2 and R6 could be configured with the **no auto-summary** command under **router rip**.

Note that RIPv2 allows for discontiguous networks, but autosummarization must be disabled for a design using discontiguous networks to work.

RIP Authentication Configuration

RIP authentication, much like EIGRP and OSPF authentication, requires the creation of keys and requires authentication to be enabled on an interface. The keys are used either as clear-text passwords or as the secret (private) key used in an MD5 calculation.

Multiple keys are allowed, and are grouped together using a construct called a *key chain*. A key chain is simply a set of related keys, each of which has a different number and may be restricted to a time period. By allowing multiple related keys in a key chain, with each key valid during specified time periods, the engineer can easily plan for migration to new keys in the future. (NTP is recommended when keys are restricted by time ranges.)

Cisco IOS enables the RIP (and OSPF and EIGRP) authentication process on a per-interface basis, referring to the key chain that holds the keys with the **ip authentication key-chain** *name* interface subcommand. The router looks in the key chain and selects the key(s) valid at that particular time. With RIP, the type of authentication (clear-text password or MD5 digest) is chosen per interface as well, using the **ip rip authentication mode** {**text** | **md5**} interface subcommand. If this command is omitted, the authentication type defaults to **text**, meaning that the key is used as a clear-text password.

Example 8-5 shows the RIP authentication configuration for R1, R2, and R4, and includes a few additional comments.

Example 8-5 *RIP Authentication (R1, R2, R4, and S1)*

```
! First, R1 Config
! Chain "carkeys" will be used on R1's LAN. R1 will use key "fred" for
! about a month, and then start using "wilma."
key chain carkeys
  key 1
   key-string fred
   accept-lifetime 08:00:00 Jan 11 2005 08:00:00 Feb 11 2005
   send-lifetime 08:00:00 Jan 11 2005 08:00:00 Feb 11 2005
  key 2
   key-string wilma
   accept-lifetime 08:00:00 Feb 10 2005 08:00:00 Mar 11 2005
   send-lifetime 08:00:00 Feb 10 2005 08:00:00 Mar 11 2005
! Next, key chain "anothersetofkeys" defines the key to be used with R4.
key chain anothersetofkeys
 key 1
  key-string barney
! Next, R1's interface subcommands are shown. First, the key chain is referenced
! using the ip rip authentication command, and the ip rip authentication mode md5
! command causes the router to use an MD5 digest of the key string.
interface FastEthernet0/0
  ip address 172.31.11.1 255.255.255.0
  ip rip authentication mode md5
  ip rip authentication key-chain carkeys
! Below, R1 enables RIPv2 authentication on the subinterface connecting to R4,
! using simple text authentication in this case (default).
interface Serial0/0.4 point-to-point
  ip address 172.31.14.1 255.255.255.252
  ip rip authentication key-chain anothersetofkeys
! R2 Config - R2 Config - R2 Config
! Next, on R2, the key chain name (housekeys) differs with R1's key chain name (carkeys), but
! the key string "fred" is the same. R2's LAN interface has MD5 authentication
! enabled to match the authentication type used on R1's LAN interface.
key chain housekeys
 key 1
  key-string fred
interface FastEthernet0/0
 ip address 172.31.11.2 255.255.255.0
 ip rip authentication mode md5
 ip rip authentication key-chain housekeys
! R4 Config - R4 Config - R4 Config
! Next, R4 enables RIP authentication on its subinterface connecting to R1, but
! without a ip rip authentication mode md5 command, it uses the key as simple
! text.
key chain boatkeys
 key 1
```

continues

Example 8-5 *RIP Authentication (R1, R2, R4, and S1) (Continued)*

```
  key-string barney
!
interface Serial0/0.1 point-to-point
 ip address 172.31.14.2 255.255.255.252
 ip rip authentication key-chain boatkeys
! S1 Config - S1 Config - S1 Config
! No Authentication is Configured, in order to induce the authentication failure.
! Commands below are on R1
!Below, the show ip protocol command lists the key chains used for authentication.
R1# sh ip protocol
Routing Protocol is "rip"
  Sending updates every 30 seconds, next due in 26 seconds
  Invalid after 180 seconds, hold down 180, flushed after 240
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Redistributing: rip
  Default version control: send version 2, receive version 2
    Interface             Send  Recv  Triggered RIP  Key-chain
    FastEthernet0/0       2     2                    carkeys
    Serial0/0.3           2     2
    Serial0/0.4           2     2                    anothersetofkeys
    Serial0/0.6           2     2
  Automatic network summarization is in effect
  Maximum path: 4
  Routing for Networks:
    172.31.0.0
  Routing Information Sources:
    Gateway         Distance      Last Update
    172.31.16.6           120      00:00:09
    172.31.11.2           120      00:00:18
    172.31.13.2           120      00:00:04
    172.31.14.2           120      00:00:08
  Distance: (default is 120)
! Above, note that 172.31.11.201 (S1's LAN IP address) is not listed as an
! information source, because authentication failed with S1. (S1 simply omitted
! any RIP authentication configuration.)
! Below, the debug output shows a reaction to the unauthenticated update received
! from S1.
R1# debug ip rip events
RIP event debugging is on
Jan 11 10:57:34.914: RIP: ignored v2 packet from 172.31.11.201 (invalid authentication)
```

Although the comments in Example 8-5 explain the more important details, one other point needs to be made regarding the key lifetimes. The configuration shows that two of the keys' lifetimes overlap by a day. On that day, RIP would use the key with the lowest key number. By using such logic, you could start by configuring one key. Later, you could then add a second key on all

the routers, with overlapping time periods—but still use the original key. Finally, you could either let the first key expire or delete the first key, allowing for easy migration.

RIP Next-Hop Feature and Split Horizon

This section covers the split horizon and next-hop features of RIPv2. These two features do not typically need to be considered at the same time, but in some cases they do. The example used in this section shows how the two features may be needed in the same design.

First, Cisco IOS controls the split horizon setting per interface, using the [**no**] **ip split-horizon** interface subcommand. Split horizon is on by default, except for cases in which Frame Relay is configured with the IP address on the physical interface.

The RIPv2 next-hop feature allows a RIP router to advertise a different next-hop router than the advertising router. To demonstrate the next-hop feature in Example 8-6, S2 will not run RIP at all, but rather will use OSPF to advertise routes to R2. R2 will in turn redistribute the OSPF routes into RIP, and advertise those routes to the other routers. When R2 advertises the OSPF-learned routes over the LAN, R1 learns these routes. However, because of the next-hop feature, R1 will point to S2 as the next-hop router. To cause that to happen, R2 advertises S2 as the next-hop router in R2's RIP updates.

Interestingly, for this example to work, the split horizon feature must be disabled on R2's fa0/0 interface. With split horizon enabled on fa0/0 (the default configuration), R1 would not normally advertise the redistributed routes out R2's fa0/0 interface, because R2's outgoing interface for those routes is also fa0/0. So, to force R2 to advertise those routes out fa0/0, thereby showing an example of RIP's advertisement of a different next-hop router, R2 must disable split horizon on fa0/0. Example 8-6 shows the details.

Example 8-6 *RIP Next-Hop Feature*

```
! R2 Config
! Note that split horizon is disabled on R2's fa0/0.
interface FastEthernet0/0
 ip address 172.31.11.2 255.255.255.0
 no ip split-horizon
! Next, R2 has a basic OSPF configuration, with simple redistribution into RIP.
router ospf 1
 network 172.31.11.0 0.0.0.255 area 0
!
router rip
 version 2
 redistribute ospf 1 metric 2
 network 10.0.0.0
 network 172.31.0.0
```

continues

Example 8-6 *RIP Next-Hop Feature (Continued)*

```
! On R2, the three OSPF-learned routes are listed; these will be the routes that
! RIP redistributes to R1 and the rest of the routers.
R2# show ip route ospf
     172.31.0.0/16 is variably subnetted, 13 subnets, 2 masks
O       172.31.223.0/24 [110/2] via 172.31.11.202, 02:01:40, FastEthernet0/0
O       172.31.222.0/24 [110/2] via 172.31.11.202, 02:01:40, FastEthernet0/0
O       172.31.221.0/24 [110/2] via 172.31.11.202, 02:01:40, FastEthernet0/0
! lines omitted for brevity
! Below, R2's first debug message mentions "172.31.11.2" as the update source of
! the next update. After that, the three messages, one per route, highlight the
! Next Hop field listing 172.31.11.202 (S2) as the next hop.
R2# debug ip rip
RIP protocol debugging is on
Jan 11 10:58:12.874: RIP: sending v2 update to 224.0.0.9 via FastEthernet0/0
    (172.31.11.2)
! some lines omitted for brevity.
Jan 11 10:58:12.878:     172.31.221.0/24 via 172.31.11.202, metric 2, tag 0
Jan 11 10:58:12.878:     172.31.222.0/24 via 172.31.11.202, metric 2, tag 0
Jan 11 10:58:12.878:     172.31.223.0/24 via 172.31.11.202, metric 2, tag 0
! Below, on R1, note the route to 172.31.221.0/24 points to next hop of S2. In a
! bit of a misnomer, the output lists the "last update from" information as if S2
! (172.31.11.202) sent the last routing update, but S2 is not even running RIP.
! The second-to-last line lists the actual update's source ("from 172.31.11.2").
R1# show ip route 172.31.221.0
Routing entry for 172.31.221.0/24
 Known via "rip", distance 120, metric 2
 Redistributing via rip
 Last update from 172.31.11.202 on FastEthernet0/0, 00:00:05 ago
 Routing Descriptor Blocks:
 * 172.31.11.202, from 172.31.11.2, 00:00:05 ago, via FastEthernet0/0
Route metric is 2, traffic share count is 1
```

RIP Offset Lists

KEY POINT

RIP *offset lists* allow RIP to add to a route's metric, either before sending an update, or for routes received in an update. The offset list refers to an ACL (standard, extended, or named) to match the routes; any matched routes have the specified *offset*, or extra metric, added to their metrics. Any routes not matched by the offset list are unchanged. The offset list also specifies which routing updates to examine by referring to a direction (in or out) and, optionally, an interface. If the interface is omitted from the command, all updates for the defined direction will be examined.

Example 8-7 shows R1, which offsets the metric by 13 for any routes in the range 172.31.208.0/21, for R1's updates sent out toward the branches (R3–R6). (The example uses the same network from Figure 8-2.) Similarly, R2 uses the same ACL matching logic, but R2 offsets the metric upon receipt of inbound routing updates on its LAN interface (fa0/0).

The routes in the 172.31.208.0/21 range are loopbacks on S1, advertised by S1 with metric 1; unchanged, R1 and R2 would have routes with a metric of 1 to these subnets, and the branch

routers would have routes with a metric of 2. After configuring the offset lists, the branches will have a metric of 15 for these routes. As a result, the branch routers will not advertise these routes any farther, because the advertised metric would have to be 16—which of course is infinite in RIP.

Example 8-7 *RIP Offset Lists*

```
! R1 Config
! Note that the offset-list command is a subcommand of the router rip command.
router rip
 version 2
 offset-list 11 out 13 Serial0/0.3
 offset-list 11 out 13 Serial0/0.4
 offset-list 11 out 13 Serial0/0.6
 network 172.31.0.0
!
access-list 11 permit 172.31.208.0 0.0.7.255
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! R2 Config
router rip
 version 2
 offset-list 11 in 13 FastEthernet0/0
 network 10.0.0.0
 network 172.31.0.0
!
access-list 11 permit 172.31.208.0 0.0.7.255
! Next, R1 lists the three routes learned from S1 with metric 1. The offset list
! on R1 did not change the offset (metric) for inbound updates to R1.
R1# show ip route
! lines omitted for brevity.
172.31.0.0/16 is variably subnetted, 13 subnets, 2 masks
R       172.31.211.0/24 [120/1] via 172.31.11.201, 00:00:28, FastEthernet0/0
R       172.31.213.0/24 [120/1] via 172.31.11.201, 00:00:28, FastEthernet0/0
R       172.31.212.0/24 [120/1] via 172.31.11.201, 00:00:28, FastEthernet0/0
! Next, R1's next update to R4, out interface s0/0.4, is shown. Without the
! offset list, R1 would advertise the listed routes with metric 2; in this
! case, it added 13 more to the metric, for a total metric of 15.
R1# debug ip rip
RIP protocol debugging is on
R1#
Jan 11 16:51:02.659: RIP: sending v2 update to 224.0.0.9 via Serial0/0.4 (172.31.14.1)
Jan 11 16:51:02.659: RIP: build update entries
! lines omitted for brevity
Jan 11 16:51:02.663:  172.31.211.0/24 via 0.0.0.0, metric 15, tag 0
Jan 11 16:51:02.663:  172.31.212.0/24 via 0.0.0.0, metric 15, tag 0
Jan 11 16:51:02.663:  172.31.213.0/24 via 0.0.0.0, metric 15, tag 0
```

```
! Next, R2 lists the three routes learned from S1 with metric 14, because the
! offset list on R2 processes incoming updates. The offset list dictated that
! R2 add 13 to the received metric, which was 1 for these three routes.
```

continues

Example 8-7 *RIP Offset Lists (Continued)*

```
R2# show ip route
! lines omitted for brevity.
     172.31.0.0/16 is variably subnetted, 13 subnets, 2 masks
R       172.31.211.0/24 [120/14] via 172.31.11.201, 00:00:28, FastEthernet0/0
R       172.31.213.0/24 [120/14] via 172.31.11.201, 00:00:28, FastEthernet0/0
R       172.31.212.0/24 [120/14] via 172.31.11.201, 00:00:28, FastEthernet0/0
```

```
! Finally, R4 learned metric 15 routes from both R1 and R2 for these routes. R4
! will not advertise these routes to other routers, as they would be infinite
! metric routes. Note that R4 puts both equal-metric routes into its routing
! table below.
R4# show ip route 172.31.208.0 255.255.248.0 longer-prefixes
! lines omitted for brevity
      172.31.0.0/16 is variably subnetted, 13 subnets, 2 masks
R       172.31.211.0/24 [120/15] via 172.31.13.1, 00:00:13, Serial0/0.1
                        [120/15] via 172.31.23.2, 00:00:09, Serial0/0.2
! Lines omitted for brevity
```

Route Filtering with Distribute Lists and Prefix Lists

KEY POINT

Outbound and inbound RIP updates can be filtered at any interface, or for the entire RIP process. To filter the routes, the **distribute-list** command is used under **router rip**, referencing an IP ACL or an IP prefix list. Any subnets matched with a **permit** clause in the ACL make it through; any that match with a **deny** action are filtered. The distribution list filtering can be performed for either direction of flow (in or out) and, optionally, for a particular interface. If the interface option is omitted, all updates coming into or out of the RIP process are filtered. (Routes can also be filtered at redistribution points, a topic covered in Chapter 11.)

The generic command, when creating a RIP distribution list that uses an ACL, is

distribute-list {*access-list-number* | *name*} {**in** | **out**} [*interface-type interface-number*]

Example 8-8 shows an inbound distribution list on router R2, filtering routes in the 172.31.196.0/22 range. For this example, R2 now receives several /24 and /30 routes from S2, using RIPv2. The routes are in the range of 172.31.192.0/21, and the goal is to filter the upper half of that numeric range. (Again, the network of Figure 8-2 is used for this example.)

Example 8-8 *RIP Distribution List*

```
! The example begins with a list of the routes that should be filtered. Note that
! the longer-prefixes option below makes the command list all routes in the range.
! The highlighted lines are the ones that will be filtered.
R2# show ip route 172.31.192.0 255.255.248.0 longer-prefixes
! Lines omitted for brevity; in this case, the legend was deleted
```

Example 8-8 *RIP Distribution List (Continued)*

```
172.31.0.0/16 is variably subnetted, 24 subnets, 3 masks
R       172.31.195.0/30 [120/1] via 172.31.11.202, 00:00:18, FastEthernet0/0
R       172.31.194.0/24 [120/1] via 172.31.11.202, 00:00:18, FastEthernet0/0
R       172.31.196.4/30 [120/1] via 172.31.11.202, 00:00:18, FastEthernet0/0
R       172.31.195.4/30 [120/1] via 172.31.11.202, 00:00:18, FastEthernet0/0
R       172.31.197.0/24 [120/1] via 172.31.11.202, 00:00:19, FastEthernet0/0
R       172.31.196.0/30 [120/1] via 172.31.11.202, 00:00:19, FastEthernet0/0
R       172.31.195.8/30 [120/1] via 172.31.11.202, 00:00:19, FastEthernet0/0
! R2's Configuration follows. access-list 2 denies all subnets in the
! 172.31.196.0/22 range, which is the set of subnets that need to be filtered.
! The distribute-list 2 in fastethernet0/0 command tells RIP to filter inbound
! RIP updates that come in fa0/0.
router rip
 version 2
 network 10.0.0.0
 network 172.31.0.0
 distribute-list 2 in FastEthernet0/0
!
access-list 2 deny   172.31.196.0 0.0.3.255
access-list 2 permit any
! Below, the results show three less subnets in the larger 172.31.192.0/21 range.
R2# show ip route 172.31.192.0 255.255.248.0 longer-prefixes
! Lines omitted for brevity; in this case, the legend was deleted
    172.31.0.0/16 is variably subnetted, 21 subnets, 3 masks
R       172.31.195.0/30 [120/1] via 172.31.11.202, 00:00:22, FastEthernet0/0
R       172.31.194.0/24 [120/1] via 172.31.11.202, 00:00:22, FastEthernet0/0
R       172.31.195.4/30 [120/1] via 172.31.11.202, 00:00:22, FastEthernet0/0
R       172.31.195.8/30 [120/1] via 172.31.11.202, 00:00:22, FastEthernet0/0
```

A RIP distribute list might refer to a prefix list instead of an ACL to match routes. Prefix lists are designed to match a range of subnets, as well as a range of subnet masks associated with the subnets. The distribute list must still define the direction of the updates to be examined (in or out), and optionally an interface.

Chapter 11 includes a more complete discussion of the syntax and formatting of prefix lists; this chapter focuses on how to call and use a prefix list for RIP. To reference a prefix list, use the following **router rip** subcommand:

distribute-list {**prefix** *list-name*} {**in** | **out** } [*interface-type interface-number*]

Example 8-9 shows the syntax, with the prefix list denying all /30 routes from the range 172.31.192.0/21. The prefix list permits all other subnets.

Example 8-9 *RIP Prefix List*

```
! The example begins with a list of the routes that should be filtered. Note that
! the longer-prefixes option below makes the command list all routes in the range.
! The highlighted lines are the ones that will be filtered.
R2# show ip route 172.31.192.0 255.255.248.0 longer-prefixes
! Lines omitted for brevity; in this case, the legend was deleted
      172.31.0.0/16 is variably subnetted, 24 subnets, 3 masks
R       172.31.195.0/30 [120/1] via 172.31.11.202, 00:00:18, FastEthernet0/0
R       172.31.194.0/24 [120/1] via 172.31.11.202, 00:00:18, FastEthernet0/0
R       172.31.196.4/30 [120/1] via 172.31.11.202, 00:00:18, FastEthernet0/0
R       172.31.195.4/30 [120/1] via 172.31.11.202, 00:00:18, FastEthernet0/0
R       172.31.197.0/24 [120/1] via 172.31.11.202, 00:00:19, FastEthernet0/0
R       172.31.196.0/30 [120/1] via 172.31.11.202, 00:00:19, FastEthernet0/0
R       172.31.195.8/30 [120/1] via 172.31.11.202, 00:00:19, FastEthernet0/0
! R2's configuration follows. The "wo2" prefix list limits the mask range to
! only /30 with the "ge 30 le 30" parameters. It matches any subnets between
! 172.31.192.0 and 172.31.199.255. Note that the prefix-list commands are global commands.
router rip
 version 2
 network 10.0.0.0
 network 172.31.0.0
 distribute-list prefix wo2 in FastEthernet0/0
!
ip prefix-list wo2 seq 5 deny 172.31.192.0/21 ge 30 le 30
ip prefix-list wo2 seq 10 permit 0.0.0.0/0 le 32
! Below, note the absence of /30 routes in the specified range, and the presence
! of the two /24 routes seen at the beginning of Example 8-8.
R2# show ip route 172.31.192.0 255.255.248.0 longer-prefixes
! Lines omitted for brevity; in this case, the legend was deleted
     172.31.0.0/16 is variably subnetted, 19 subnets, 3 masks
R       172.31.194.0/24 [120/1] via 172.31.11.202, 00:00:23, FastEthernet0/0
R       172.31.197.0/24 [120/1] via 172.31.11.202, 00:00:23, FastEthernet0/0
```

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most of the Cisco Press *Exam Certification Guides*, this book does not repeat information presented in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 8-5 lists the protocols mentioned in this chapter and their respective standards documents.

Table 8-5 *Protocols and Standards for Chapter 8*

Protocol or Feature	Standard
RIP (Version 1)	RFC 1058
RIP (Version 2)	RFC 2453
RIP Update Authentication	RFC 2082
RIP Triggered Extensions for On-Demand Circuits	RFC 2091

Table 8-6 lists some of the most significant Cisco IOS commands related to the topics in this chapter.

Table 8-6 *Command Reference for Chapter 8*

Command	Command Mode and Description
router rip	Global config; puts user in RIP configuration mode
network *ip-address*	RIP config mode; defines classful network, with all interfaces in that network sending and able to receive RIP advertisements
distribute-list [*access-list-number* | *name* | **prefix** *name*] | {**in** | **out**} [*interface-type* | *interface-number*]	RIP config mode; defines ACL or prefix list to filter RIP updates
ip split-horizon	Interface subcommand; enables or disables split horizon
passive-interface [**default**] {*interface-type interface-number*}	RIP config mode; causes RIP to stop sending updates on the specified interface
timers basic *update invalid holddown flush*	RIP config mode; sets the values for RIP timers
version {**1** | **2**}	RIP config mode; sets the RIP version to version 1 or version 2

continues

Table 8-6 *Command Reference for Chapter 8 (Continued)*

Command	Command Mode and Description
offset-list {*access-list-number* | *access-list-name*} {**in** | **out**} *offset* [*interface-type interface-number*]	RIP config mode; defines rules for RIP to add to the metrics of particular routes
neighbor *ip-address*	RIP config mode; identifies a neighbor to which unicast RIP updates will be sent
show ip route rip	User mode; displays all routes in the IP routing table learned by RIP
show ip rip database	User mode; lists all routes learned by RIP, even if a route is not in the routing table because of a route with lower administrative distance
debug ip rip	Enable mode; displays details of RIP processing
show ip protocols	User mode; lists RIP timer settings, current protocol status, autosummarization actions, and update sources
clear ip route {*network* [*mask*] | *****}	Enable mode; clears the routing table entry, and with RIP, sends RIP requests, quickly rebuilding the routing table
show ip interface [*type number*] [**brief**]	User mode; lists many interface settings, including split horizon
key chain *name-of-chain*	Global config; defines name of key chain for routing protocol authentication
key *key-id*	Key config mode; identifies a key by number
key *string*	Key config mode; defines the text of the key
send-lifetime [*start-time* {**infinite** | *end-time* | **duration** *seconds*}]	Key config mode; defines when the key is valid to be used for sent updates
accept-lifetime [*start-time* {**infinite** | *end-time* | **duration** *seconds*}]	Key config mode; defines when the key is valid for received updates
ip rip authentication key-chain *name-of-chain*	Interface mode; enables RIP authentication on the interface
ip rip authentication mode {**text** | **md5**}	Interface mode; defines RIP authentication as clear text (default) or MD5

No single chapter in this book covers the details of the three uses of the terms classful and classless. This chapter mentions the final of the three ways in which the terms are used—specifically, their use regarding routing protocols. (IPv1 and IGRP are classful routing protocols, and RIPv2, EIGRP, OSPF, and IS-IS are classless routing protocols.) Table 8-7 summarizes and compares the three uses of these terms.

Table 8-7 *Comparing the Use of the Terms Classless and Classful*

As the terms pertain to . . .	Meaning of "Classless"	Meaning of "Classful"
Addressing (Chapter 4)	Class A, B, and C rules are not used; addresses have two parts, the prefix and host.	Class A, B, and C rules are used; addresses have three parts, the network, subnet, and host.
Routing (Chapter 7)	If no specific routes are matched for a given packet, the router forwards based on the default route.	The router first attempts a match of the classful network. If found, but none of the routes in that classful network matches the destination of a given packet, the default route is not used.
Routing protocols (Chapter 8)	Routing protocol does not need to assume details about the mask, as it is included in the routing updates; supports VLSM and discontiguous networks.	Routing protocol does need to assume details about the mask, as it is not included in the routing updates; does not support VLSM and discontiguous networks.

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Topic icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

Holddown timer, Invalid timer, Flush timer, Garbage timer, authentication, Update timer, triggered updates, flash updates, split horizon, route poisoning, poison reverse, counting to infinity, hello interval, full update, partial update, Route Tag field, Next Hop field, Triggered Extensions to RIP for On-Demand Circuits, MD5, offset list, prefix list, distribution list, distance vector, metric

Refer to the CD-based glossary to check your answers.

Further Reading

This chapter focuses on TCP/IP protocols; much more information can be found in the RFCs mentioned throughout the chapter.

The RIP RFCs listed in Table 8-5 provide good references for RIP concepts.

Jeff Doyle's *Routing TCP/IP*, Volume I, Second Edition, (Cisco Press) has several wonderful configuration examples and provides a good explanation of the concepts.

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- IP Routing
  - EIGRP
  - The use of **show** and **debug** commands

CHAPTER 9

EIGRP

This chapter covers most of the features, concepts, and commands related to EIGRP. Chapter 11, "IGP Route Redistribution, Route Summarization, and Default Routing," covers a few other details of EIGRP—in particular, route redistribution, route filtering when redistributing, and route summarization.

"Do I Know This Already?" Quiz

Table 9-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 9-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
EIGRP Basics and Steady-State Operation	1–4	
EIGRP Convergence	5–7	
EIGRP Configuration	8–9	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. Which of the following items are true of EIGRP?
   a. Authentication can be done using MD5 or clear text.
   b. Uses UDP port 88.
   c. Sends full or partial updates as needed.
   d. Multicasts updates to 224.0.0.10.

2. Four routers (R1, R2, R3, and R4) are attached to the same VLAN. R1 has been configured for an EIGRP Hello timer of 3. R2 has been configured with a **metric weights 0 0 0 1 0 0** command. R3 has been configured with a hold time of 11 seconds. Their IP addresses are

10.1.1.1, 10.1.1.2, 10.1.1.3, and 10.1.1.4, with /24 prefixes, except R4, which has a /23 prefix configured. All other related parameters are set to their default. Select the routers that are able to collectively form neighbor relationships.

a. R1

b. R2

c. R3

d. R4

e. None of them can form a neighbor relationship.

3. In the following command output, what do the numbers in the column labeled "H" represent?

```
R1# show ip eigrp neighbors
IP-EIGRP neighbors for process 1
H   Address                 Interface       Hold Uptime    SRTT     RTO   Q   Seq
                                            (sec)          (ms)         Cnt  Num
2   172.31.11.2             Fa0/0            4 00:03:10      1    4500    0   233
1   172.31.11.202           Fa0/0           11 00:04:43      1    4500    0    81
0   172.31.11.201           Fa0/0           14 00:05:11   1927    5000    0    84
```

a. The current Hold Time countdown

b. The number of seconds before a Hello is expected

c. The order in which the neighbors came up

d. None of the other answers are correct

4. Which of the following is not true regarding the EIGRP Update message?

a. Updates require an acknowledgement with an Ack message.

b. Updates can be sent to multicast address 224.0.0.10.

c. Updates are sent as unicasts when they are retransmitted.

d. Updates always include all routes known by a router, with partial routing information distributed as part of the EIGRP Reply message.

5. The output of a **show ip eigrp topology** command lists information about subnet 10.1.1.0/24, with two successors, and three routes listed on lines beginning with "via." How many feasible successor routes exist for 10.1.1.0/24?

a. 0

b. 1

c. 2

d. 3

e. Cannot determine from the information given

6. The following command output shows R11's topology information for subnet 10.1.1.0/24. Then R11 and R12 (IP address 10.1.11.2) are connected to the same LAN segment. Then R11's EIGRP Hold Time expires for neighbor R12. Which of the following is true about R11's first reaction to the loss of its neighbor R12?

```
R11# show ip eigrp topology
! lines omitted for brevity
P 10.1.1.0/24, 1 successors, FD is 1456
        via 10.1.11.2 (1456/1024), FastEthernet0/0
        via 10.1.14.2 (1756/1424), Serial0/0.4
```

   a. R11 sends Updates to all neighbors poisoning its route to 10.1.1.0/24.

   b. R11 replaces the old route through 10.1.11.2 with the feasible successor route through 10.1.14.2.

   c. R11 sends Query messages to all other neighbors to ensure that the alternate route through 10.1.14.2 is loop free, before using the route.

   d. R11 first Queries only neighbors on interface fa0/0 for alternative routes before Querying the rest of its neighbors.

7. EIGRP router R11 has just changed its route to subnet 10.1.2.0/24 to the active state, and has sent a Query to five neighbors. Which of the following is true about the next step taken by R11?

   a. R11 adds a new route to 10.1.2.0/24 to the routing table as soon as it receives an EIGRP Reply that describes a new route to 10.1.2.0/24.

   b. R11 can add a new route to 10.1.2.0/24 after receiving Reply messages from all 5 neighbors.

   c. R11 can add a new route for 10.1.2.0/24 to the routing table, even without 5 Reply messages, once the Hold timer expires.

   d. R11 can add a new route for 10.1.2.0/24 to the routing table, even without 5 Reply messages, once the Dead timer expires.

8. EIGRP router R11 has five interfaces, with IP address 10.1.1.11/24 (interface fa0/0), 10.1.2.11/24, 10.1.3.11/24, 10.1.4.11/24, and 10.1.5.11/24. Its EIGRP configuration is shown below. Which of the following answers is true regarding this router?

```
router eigrp 1
 network 10.1.0.0 0.0.3.255
 passive-interface fa0/0
```

   a. R11 will send EIGRP Updates out fa0/0, but not process received EIGRP Updates.

   b. R11 will advertise connected subnets 10.1.3.0/24 and 10.1.4.0/24.

   c. R11 will advertise subnets 10.1.1.0/24 and 10.1.2.0/24, as well as attempt to send Hellos and Updates on the related interfaces.

   d. The **network** command does not match any interfaces, so EIGRP will essentially do nothing.

9. EIGRP router Br1 is a branch router with two Frame Relay subinterfaces (s0/0.1 and s0/0.2) connecting it to distribution routers. It also has one LAN interface, fa0/0. No other routers connect to the Br1 LAN. Which of the following scenarios prevent router Br1 from sending EIGRP Hellos out its fa0/0 interface?

   a. The inclusion of the **passive-interface fa0/0** command on Br1

   b. The inclusion of the **eigrp stub** command on Br1

   c. The inclusion of the **eigrp stub receive-only** command on Br1

   d. The lack of a **network** command that matches the IP address of Br1's fa0/0 interface

Foundation Topics

EIGRP Basics and Steady-State Operation

Many CCIE candidates have known, at least at some point, many of the details of EIGRP operation and configuration. EIGRP is widely deployed and is thoroughly covered on the CCNP BSCI exam. With that in mind, this chapter strives to review the key terms and concepts briefly, and get right to specific examples that detail EIGRP operation on a Cisco router. To that end, the chapter begins with Table 9-2, which lists some of the key features related to EIGRP.

Table 9-2 *EIGRP Feature Summary*

KEY POINT

Feature	Description
Transport	IP, protocol type 88 (does not use UDP or TCP).
Metric	Based on constrained bandwidth and cumulative delay by default, and optionally load, reliability, and MTU.
Hello interval	Interval at which a router sends EIGRP Hello messages on an interface.
Hold timer	Timer used to determine when a neighboring router has failed, based on a router not receiving any EIGRP messages, including Hellos, in this timer period.
Update destination address	Normally sent to 224.0.0.9, with retransmissions being sent to each neighbor's unicast IP address.
Full or partial updates	Full updates are used when new neighbors are discovered; otherwise, partial updates are used.
Authentication	Supports MD5 authentication only.
VLSM/classless	EIGRP includes the mask with each route, also allowing it to support discontiguous networks and VLSM.
Route Tags	Allows EIGRP to tag routes as they are redistributed into EIGRP.
Next-hop field	Supports the advertisement of routes with a different next-hop router than the advertising router.
Manual route summarization	Allows route summarization at any point in the EIGRP network.
Multiprotocol	Supports the advertisement of IPX and AppleTalk routes.

Hellos, Neighbors, and Adjacencies

After a router has been configured for EIGRP, and its interfaces come up, it attempts to find neighbors by sending EIGRP *Hellos* (destination 224.0.0.10). Once a pair of routers have heard each other say Hello, they become adjacent—assuming several key conditions are met. Once

neighbors pass the checks in the following list, they are considered to be adjacent. At that point, they can exchange routes and are listed in the output of the **show ip eigrp neighbor** command.

KEY POINT

- Must pass the authentication process
- Must use the same configured AS number
- Must believe that the source IP address of a received Hello is in that router's primary connected subnet on that interface
- K values must match

The wording of the third item in the list bears a little further scrutiny. The *primary subnet* of an interface is the subnet as implied by the **ip address** command that does not have the **secondary** keyword. An EIGRP router looks at the source IP address of a Hello; if the source IP address is a part of that router's primary subnet of the incoming interface, the Hello passes the IP address check.

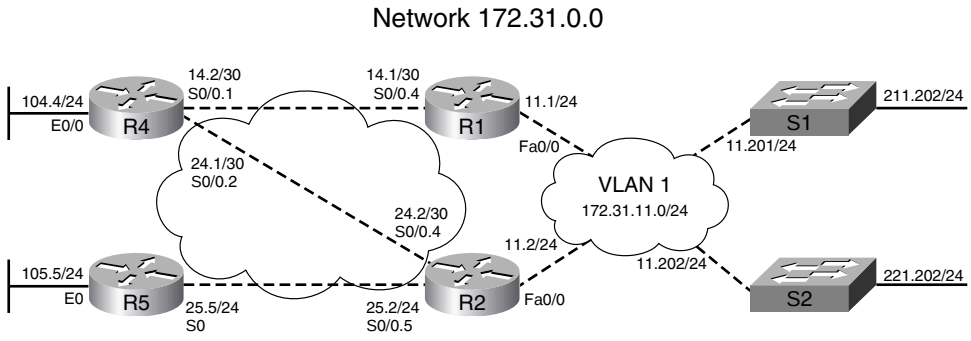
This logic leaves open some interesting possibilities. For example, if the routers are misconfigured with different subnet masks, the check may still pass. If one router has configured 10.1.2.1/24, and the other has configured 10.1.2.2/23, they could become adjacent, assuming all the other checks pass. While EIGRP supports secondary IP addresses and subnets, EIGRP sources its messages from the address in the primary subnet, and the IP addresses of neighbors must be in the subnet of the primary subnets.

The last item in the list mentions K values; *K values* are constants that define the multipliers used by EIGRP when calculating metrics. The settings can be changed with a **router eigrp** subcommand **metric weights** ***tos*** *k1 k2 k3 k4 k5*. The command defaults to a setting of **0 1 0 1 0 0**, meaning that only bandwidth and delay are used to calculate the metric. (The examples in this chapter habitually change the settings to **0 0 0 1 0 0**, which removes bandwidth from the calculation and makes the metrics in the examples a little more obvious.)

Besides simply checking to see if the right parameters agree, the Hello messages also serve as an EIGRP keepalive. Adjacent routers continue to multicast Hellos based on each interface's EIGRP *hello interval*. If a router fails to hear from a neighbor for a number of seconds, defined by the EIGRP *Hold Time* for that neighbor, all routes through the neighbor are considered to have failed.

Example 9-1 shows how a router displays some of the basic information regarding EIGRP operations based on Figure 9-1. The example begins with four routers (R1, R2, S1, and S2) that have only their common LAN interfaces up, just to show the Hello process. By the end of the example, the R2-to-R5 PVC will come up, but the EIGRP adjacency will fail due to a K-value mismatch.

Figure 9-1 *Sample Internetwork Used for EIGRP Examples*



Example 9-1 *Forming EIGRP Adjacencies*

```
! First, a debug is initiated on R1.
R1# debug eigrp packet hello
EIGRP Packets debugging is on
    (HELLO)
Jan 11 13:27:19.714: EIGRP: Received HELLO on FastEthernet0/0 nbr 172.31.11.201
Jan 11 13:27:19.714:   AS 1, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ
  un/rely 0/0
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! S2's LAN interface brought up, not shown
! Below, a pair of log messages appear, announcing the new neighbor; this message
! appears due to the default router eigrp subcommand eigrp log-neighbor-changes.
Jan 11 13:27:19.995: EIGRP: New peer 172.31.11.202
Jan 11 13:27:19.995: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 172.31.11.202
  (FastEthernet0/0) is up: new adjacency
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Next, only neighbors who become adjacent—those that pass all the required
! checks for the parameters—are listed. The Hold timer is shown; it starts at
! its maximum, and decrements towards 0, being reset upon the receipt of any EIGRP
! packet from that neighbor. The "H" column on the left states the order in
! which the neighbors became adjacent.
R1# show ip eigrp neighbors
IP-EIGRP neighbors for process 1
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
2   172.31.11.2             Fa0/0              4 00:03:10    1  4500  0  233
1   172.31.11.202           Fa0/0             11 00:04:43    1  4500  0  81
0   172.31.11.201           Fa0/0             14 00:05:11 1927  5000  0  84
! Below, the PVC between R2 and R5 came up, but R5's K values do not match R2's.
! Both messages below are log messages, with no debugs enabled on either router.
! Next message on R5 !!!!!!!!!!
03:55:51: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 172.31.25.2 (Serial0) is down: K-value
  mismatch
! Next message on R2 !!!!!!!!!!
Jan 11 13:21:45.643: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 172.31.25.5 (Serial0/0.5)
  is down: Interface Goodbye received
```

Note that when the PVC between R2 and R5 comes up, the message on R5 is pretty obvious, but the message at R2 says nothing about K values. Some later releases of Cisco IOS mistake invalid EIGRP K-value settings as a newer EIGRP message called a *Goodbye* message. Goodbye messages allow routers to tell each other that they are shutting down in a graceful fashion; be aware that this message may simply be the result of a K-value mismatch.

KEY POINT

Interestingly, the Hello and Hold time parameters do not need to match for EIGRP neighbor relationships to form. In fact, a router does not use its own timers when monitoring a neighbor relationship—instead, it uses each neighbor's stated timers, as exchanged in the Hello messages. For example, in Example 9-1, R2 has been configured with Hello and Hold timer settings at 2 and 6 seconds, respectively, with R1 defaulting to 5 and 15 seconds. As R1 monitors its neighbor connection to R2, R1 resets the Hold timer to 6 seconds upon receipt of an EIGRP message. With a hello interval of 2 seconds, R1's listing for hold time for R2 shows it fluctuating between 6 and 4, assuming no Hellos are lost. Note the **show ip eigrp neighbors** command on R1 near the end of the example—under normal operation, this value fluctuates between 6 and 4 seconds. The other neighbors default to Hello and Hold time of 5 and 15, so R1's Hold time in the command output fluctuates between 15 and 10 for these neighbors, assuming no Hellos are lost.

EIGRP Updates

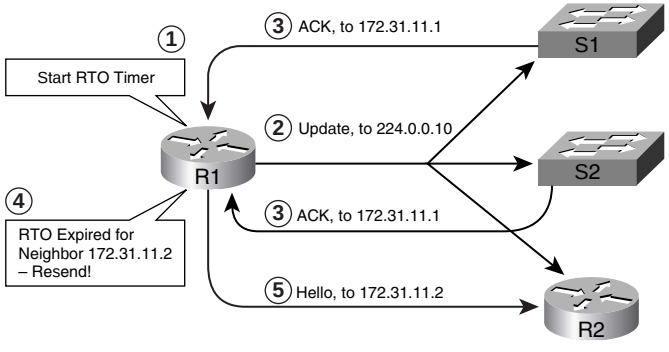
Once routers are adjacent, they can exchange routes using EIGRP *Update* messages. The process follows this general sequence:

KEY POINT

1. Initially, full updates are sent, including all routes except those omitted due to split horizon.
2. Once all routes have been exchanged, the updates cease.
3. Future partial updates occur when one or more routes change.
4. If neighbors fail and recover, or new neighbor adjacencies are formed, full updates are sent.

EIGRP uses the *Reliable Transport Protocol (RTP)* to send the multicast EIGRP updates. EIGRP sends updates, waiting on a unicast EIGRP ACK message from each recipient. Figure 9-2 shows the general idea over a LAN.

Figure 9-2 *EIGRP Use of RTP on a LAN*



RTP allows the Updates to be sent as multicasts. If any neighbors fail to acknowledge receipt of the multicasted update, RTP resends Updates as unicasts just to those neighbors. The steps run as follows, using Figure 9-2 as an example:

1. The EIGRP sender (R1 in Figure 9-2) starts a *Retransmission Timeout (RTO)* timer for each neighbor when sending a reliable message like an Update. (Cisco IOS actually calculates a *Smoothed Round-Trip Time*, or SRTT, to each neighbor, and derives RTO from the SRTT; both values are shown in the **show ip eigrp neighbor** output. These values vary over time.)
2. R1 sends the multicast EIGRP Update.
3. R1 notes from which neighbors it receives an EIGRP ACK for the Update.
4. RTO expired before router R2 sent its EIGRP ACK.
5. R1 resends the Update, this time as a unicast, and only to the neighbor(s) that did not reply within the RTO time (R2 in this case).

This process allows efficient multicasting of updates under normal circumstances, and efficient retransmission when ACKs do not arrive in time.

EIGRP and RTP use a simple acknowledgement process with a window size of one message. Each Update packet has a sequence number, with the returned ACK message confirming receipt of the message by listing that same sequence number. Example 9-2 shows the location of the sequence number information in both **show** and **debug** commands. (In the example, R1 does a **no shut** on a loopback interface [IP address 172.31.151.1/24], with R1 sending an update advertising the newly-available route.)

Example 9-2 *Sequence Numbers in EIGRP Updates and ACKs*

```
! First, note the show ip eigrp neighbor output on router R2. The last column
! lists the sequence number last used by that neighbor to send a "reliable"
! packet. So, R2 expects R1's next reliable EIGRP message to have sequence number
! 225. Also, the RTO calculations are listed for each neighbor. Note
! that the SRTT value is 0 until some reliable packets are exchanged, as SRTT
! is calculated based on actual round-trip time measurements.
R2# sh ip eigrp neighbor
IP-EIGRP neighbors for process 1
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
2   172.31.11.1             Fa0/0            5 01:14:03    1     200  0  224
1   172.31.11.202           Fa0/0           13 01:15:36    1     200  0  92
0   172.31.11.201           Fa0/0           13 01:16:04  257    1542  0  96
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! R1 - R1 - R1 - R1
! Next, the debug command on R1 enables debug for Update and Ack packets.
```

continues

Example 9-2 *Sequence Numbers in EIGRP Updates and ACKs (Continued)*

```
R1# debug eigrp packet update ack
EIGRP Packets debugging is on
    (UPDATE, ACK)
! Not Shown: R1's loop0 is "no shutdown," interface address 172.31.151.1/24.
! Below, the debug messages show R1's update, and each of the other three routers'
! Acks. Note R1's update has "sequence" 225, and the Acks list that same sequence
! number after the slash.
Jan 11 14:43:35.844: EIGRP: Enqueueing UPDATE on FastEthernet0/0 iidbQ un/rely 0/1 serno
  207-207
Jan 11 14:43:35.848: EIGRP: Sending UPDATE on FastEthernet0/0
Jan 11 14:43:35.848:   AS 1, Flags 0x0, Seq 225/0 idbQ 0/0 iidbQ un/rely 0/0 serno 207-207
Jan 11 14:43:35.848: EIGRP: Received ACK on FastEthernet0/0 nbr 172.31.11.202
Jan 11 14:43:35.852:   AS 1, Flags 0x0, Seq 0/225 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely
  0/1
Jan 11 14:43:35.852: EIGRP: Received ACK on FastEthernet0/0 nbr 172.31.11.2
Jan 11 14:43:35.852:   AS 1, Flags 0x0, Seq 0/225 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely
  0/1
```

The EIGRP Topology Table

EIGRP uses three tables: the neighbor table, the topology table, and the IP routing table. The neighbor table keeps state information regarding neighbors, and is displayed using the **show ip eigrp neighbors** command. EIGRP Update messages fill the routers' EIGRP topology tables. Based on the contents of the topology table, each router chooses its best routes and installs these routes in its respective IP routing table.

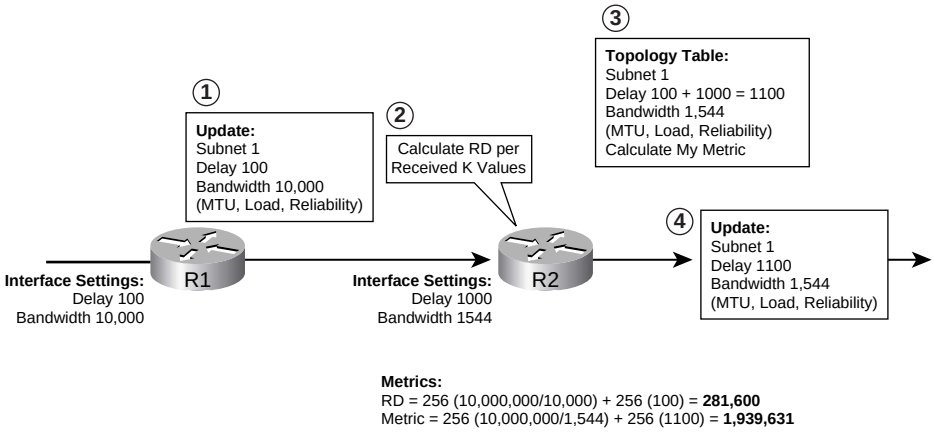
An EIGRP router calculates the metric for each route based on the components of the metric. When a neighboring router advertises a route, the Update includes the metric component values for each route. The router then considers the received metric values, as well its own interfaces settings, to calculate its own metric for each route. The default metric components are cumulative delay, in tens of microseconds, and the constraining bandwidth for the entire route, in bits per second. By setting the correct K values in the **metric weights** command, EIGRP can also consider link load, reliability, and MTU. Cisco recommends not using those values, in large part due to the fluctuation created by the rapidly changing calculated metrics and repeated routing reconvergence.

Figure 9-3 depicts the general logic relating to the metric components in a routing update, showing the units on the **bandwidth** and **delay** commands versus the contents of the updates.

NOTE A router considers its interface delay settings, as defined with the **delay** interface subcommand, when calculating EIGRP metrics. The **delay** command's units are tens of microseconds, so a **delay 1** command sets the interface delay as 10 microseconds.

Figure 9-3 *EIGRP Update and Computing the Metric*

KEY POINT



Because the received update includes the neighbor's metric components, a router can calculate the advertising neighbor's metric for a route—called the *reported distance (RD)*. A router can, of course, also calculate its own metric for a particular route, after adding its own interface delay and considering whether it should adjust the value for the constraining bandwidth. For example, consider the four steps outlined in Figure 9-3:

1. R1 advertises a route, with bandwidth = 10,000 and delay = 100.
2. R2 calculates the RD for this route per the received K values.
3. R2 updates its topology table, adding delay 1000 because the interface on which R2 received the update has a delay setting of 1000. It also uses a new bandwidth setting, because the received Update's bandwidth (10,000) was greater than R2's incoming interface's bandwidth (1544).
4. R2's update to another neighbor includes the new (cumulative) delay and the new (constraining) bandwidth.

Assuming default K-value settings, the EIGRP formula for the metric calculation is

$$\text{Metric} = 256 \left(10^7/\text{bandwidth}\right) + 256 \left(\text{delay}\right)$$

The **show ip eigrp topology** command lists the RD and the locally computed metric for the entries in the EIGRP topology table. Example 9-3 shows a few details of where the RD and local metric can be seen in **show** command output. The example is based on Figure 9-1, with all routers and interfaces now working properly. Also, to keep things simple, the **delay** command has been used to set all links to **delay 1** (LANs), **delay 2** (WANs), or **delay 3** (loopbacks). Also, the **metric weights 0 0 0 1 0 0** command was used on each router, taking bandwidth out of the calculation, making the calculated metrics a little more meaningful in the command output.

Example 9-3 *EIGRP Topology Table*

```
! First, the numbers in parentheses show this router's (R1's) calculated metric,
! then a "/", then the RD. For example, S1 advertised the route to 211.0/24, with
! R1 calculating S1's metric (the RD) as 768. Delay 3 was set on S1's loopback
! (where 211.0/24 resides), so its metric was 3*256=768. R1's metric adds delay 1,
! for a metric of 4*256=1024.
R1# show ip eigrp topology
IP-EIGRP Topology Table for AS(1)/ID(172.31.16.1)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status
P 172.31.151.0/24, 1 successors, FD is 768
         via Connected, Loopback1
P 172.31.211.0/24, 1 successors, FD is 1024
        via 172.31.11.201 (1024/768), FastEthernet0/0
P 172.31.24.0/30, 1 successors, FD is 768
        via 172.31.11.2 (768/512), FastEthernet0/0
        via 172.31.14.2 (1024/512), Serial0/0.4
! Lines omitted for brevity
! Below, the metric in the IP routing table entries match the first number in
! the parentheses, as well as the number listed as "FD is…" in the output above.
R1# show ip route
! omitted legend for brevity
     172.31.0.0/16 is variably subnetted, 9 subnets, 2 masks
D       172.31.211.0/24 [90/1024] via 172.31.11.201, 00:29:42, FastEthernet0/0
D       172.31.24.0/30 [90/768] via 172.31.11.2, 00:29:44, FastEthernet0/0
! Lines omitted for brevity
```

KEY POINT

The **show ip eigrp topology** command lists a few additional very important concepts and terms related to how EIGRP chooses between multiple possible routes to the same prefix. First, the term *feasible distance (FD)* refers to this router's best calculated metric among all possible routes to reach a particular prefix. The FD is listed as "FD is *x*" in the command output. The route that has this best FD is called the *successor route*, and is installed in the routing table. The successor route's metric is by definition called the feasible distance, so that metric is what shows up in the routes shown with the **show ip route** command. These additional terms all relate to how EIGRP processes convergence events, which is explained next.

EIGRP Convergence

Once all the EIGRP routers have learned all the routes in the network, and placed the best routes (the successor routes) in their IP routing tables, their EIGRP processes simply continue to send Hellos, expect to receive Hellos, and look for any changes to the network. When those changes do occur, EIGRP must converge to use the best available routes. This section covers the three major

components of EIGRP convergence: input events, local computation (which includes looking for feasible successors), and using active querying to find alternative routes.

Table 9-3 lists several of the key EIGRP terms related to convergence. Following the table, the text jumps right into what EIGRP does when a topology or metric change occurs.

Table 9-3 *EIGRP Features Related to Convergence*

KEY POINT

EIGRP Convergence Function	Description
Reported distance (RD)	The metric (distance) of a route as reported by a neighboring router
Feasible distance (FD)	The metric value for the lowest-metric path to reach a particular subnet
Feasibility condition	When multiple routes to reach one subnet exist, the case in which one route's RD is lower than the FD
Successor route	The route to each destination prefix for which the metric is the lowest metric
Feasible successor (FS)	A route that is not a successor route but meets the feasibility condition; can be used when the successor route fails, without causing loops
Input event	Any occurrence that could change a router's EIGRP topology table
Local computation	An EIGRP router's reaction to an input event, leading to the use of a feasible successor or going active on a route

Input Events and Local Computation

An EIGRP router needs to react when an *input event* occurs. The obvious input events are when a router learns of new prefixes via newly received routing updates, when an interface fails, or when a neighbor fails. Because EIGRP sends updates only as a result of changed or new topology information, a router must consider the update and decide if any of its routes have changed.

When an input event implies that a route has failed, the router performs *local computation*, a fancy term for a process that can be boiled down to relatively simple logic. In short, the result of local computation is that the router either is able to choose a replacement route locally, without having to ask any neighbors, or is required to ask neighbors for help. Simply put, for a failed route, local computation does the following:

KEY POINT

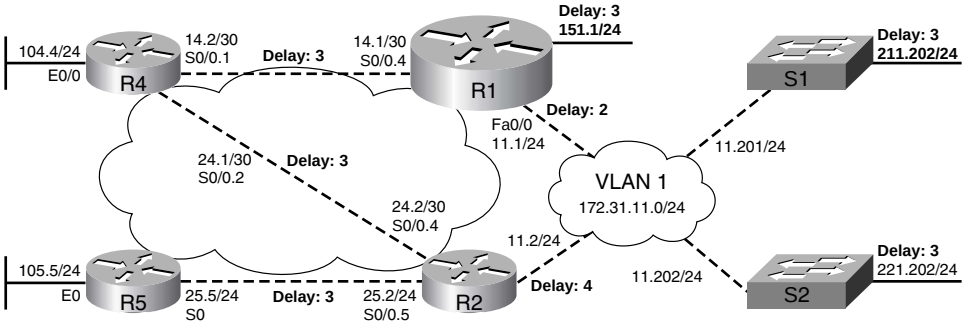
- If FS routes exist, install the lowest-metric FS route into the routing table, and send Updates to neighbors to notify them of the new route.
- If no FS route exists, actively query neighbors for a new route.

To be an FS route, a route must meet the *feasibility condition*, defined as follows:

The RD must be lower than this router's current FD for the route.

The local computation is best understood by looking at an example. Figure 9-4 shows the same network as in Figure 9-1, but with delay values shown. Example 9-4 begins with R4 using a successor route to 172.31.211.0/24, through R1. R4 also has an FS route to 172.31.211.0/24 through R2. The example shows what happens when the PVC from R1 to R4 fails, and R4's neighbor relationship with R1 fails, causing R4 to perform local computation and start using its FS route through R2.

Figure 9-4 *Network Used for EIGRP Convergence Examples*



NOTE The routers have disabled the use of bandwidth in the EIGRP metric calculation, so all metrics in Example 9-4 are multiples of 256.

Example 9-4 *Local Computation: R1-R4 Link Fails; R4 Finds an FS to 172.31.211.0/24 Through R2*

```
! First, the current successor route on R4 points out S0/0.1, to R1, metric 2048.
R4# show ip route
! lines omitted for brevity
     172.31.0.0/16 is variably subnetted, 9 subnets, 2 masks
D       172.31.211.0/24 [90/2048] via 172.31.14.1, 00:01:46, Serial0/0.1
! Below, the FD is listed as 2048 as well. The topology entry for the successor
! has the same 2048 metric listed as the first number in parentheses; the second
! number is the RD on R1 (1280). The second topology entry for this route lists
! metric 2560, RD 1280; with RD in the second route being less than the FD, this
! second route meets the feasibility condition, making it an FS route.
R4# show ip eigrp topology
IP-EIGRP Topology Table for AS(1)/ID(172.31.104.4)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
r - reply Status, s - sia Status
! lines omitted for brevity
P 172.31.211.0/24, 1 successors, FD is 2048
        via 172.31.14.1 (2048/1280), Serial0/0.1
        via 172.31.24.2 (2560/1792), Serial0/0.2
! Next, R4 loses Neighbor R1, with EIGRP Finite State Machine (FSM) debug on.
R4# debug eigrp fsm
```

Example 9-4 *Local Computation: R1-R4 Link Fails; R4 Finds an FS to 172.31.211.0/24 Through R2 (Continued)*

```
EIGRP FSM Events/Actions debugging is on
Jan 12 07:17:42.391: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 172.31.14.1 (Serial0/0.1)
  is down: holding time expired
! Below, debug messages have been edited to only show messages relating to
! the route to 172.31.211.0/24. R4 looks for an FS, finds it, replaces the old
! successor with the FS, and sends updates telling neighbors about the new route.
Jan 12 07:17:42.399: DUAL: Destination 172.31.211.0/24
Jan 12 07:17:42.399: DUAL: Find FS for dest 172.31.211.0/24. FD is 2048, RD is 2048
Jan 12 07:17:42.399: DUAL: 172.31.14.1 metric 4294967295/4294967295
Jan 12 07:17:42.399: DUAL: 172.31.24.2 metric 2560/1792 found Dmin is 2560
Jan 12 07:17:42.399: DUAL: Removing dest 172.31.211.0/24, nexthop 172.31.14.1
Jan 12 07:17:42.403: DUAL: RT installed 172.31.211.0/24 via 172.31.24.2
Jan 12 07:17:42.403: DUAL: Send update about 172.31.211.0/24.  Reason: metric chg
Jan 12 07:17:42.403: DUAL: Send update about 172.31.211.0/24.  Reason: new if
! Finally, note that the FD is unchanged; the FD is never raised until the route
! has been actively queried. The new route info has been put in the routing table.
R4# show ip eigrp topology
! lines omitted for brevity
P 172.31.211.0/24, 1 successors, FD is 2048
        via 172.31.24.2 (2560/1792), Serial0/0.2
R4# show ip route
! Lines omitted for brevity
D       172.31.211.0/24 [90/2560] via 172.31.24.2, 00:00:25, Serial0/0.2
```

KEY POINT

Going Active on a Route

The second branch in the local computation logic causes the EIGRP router to ask its neighbors about their current best route to a subnet, hoping to find an available, loop-free alternative route to that subnet. When no FS route is found, the EIGRP router goes active for the route. *Going active* is jargon for the process of changing a route's status to active. Once the router is active, EIGRP multicasts *Query* messages to its neighbors, asking the neighbors if they have a valid route to the subnet. The neighbors should unicast EIGRP *Reply* packets back to the original router, stating whether or not they have a current loop-free route with which to reach that prefix.

Once a router receives Reply messages from all the neighbors to which it sent Queries, the router updates its topology table with all the new information learned in the Reply messages, recomputes metrics for any known routes, and chooses a new successor. Of course, if no routes to that subnet are found, this router simply does not add a route to the routing table.

NOTE The EIGRP term "active" refers to a route for which a router is currently using the Query process to find a loop-free alternative route. Conversely, a route is in passive state when it is not in an active state.

The neighboring routers view any received Query messages as an input event. Each neighbor router's behavior when receiving a Query can be summarized as follows:

KEY POINT

1. If the router does not have an entry in its topology table for that subnet, it sends an EIGRP Reply packet stating that it has no route.
2. If the router's successor for that subnet is unchanged, or an FS is found, the neighbor sends back an EIGRP Reply message with the details of the route.
3. If the conditions in step 1 or 2 do not exist, the router itself goes active, and withholds its EIGRP response to the original Query, until all of its neighbors respond.

Note that the logic in the third step can result in a route for which the Active Querying process never completes. Routes that stay in active state too long are considered to be *stuck-in-active* routes. The related concepts are covered in the next section.

Example 9-5 shows an example of the Query process. The example is again based on Figure 9-4, with R4 again losing its neighbor relationship with R1. In this case, R4's local computation will not find an FS for its failed route to 172.31.151.0/24, so it must go active.

Example 9-5 *R1-R4 Link Fails; R4 Actively Queries for 172.31.151.0/24*

```
! First, the show ip eigrp topology command only lists the successor route, and no
! FS routes. This command does not list non-FS routes.
R4# show ip eigrp topo
! Lines omitted for brevity

P 172.31.151.0/24, 1 successors, FD is 1536
        via 172.31.14.1 (1536/768), Serial0/0.1
! Below, the show ip eigrp topology all-links command includes non-FS routes,
! in this case including the non-FS route to 151.0/24 through R2. Note that this
! alternate non-FS route's RD is 1792, which is more than the FD of 1536.
R4# show ip eigrp topology all-links
! Lines omitted for brevity

P 172.31.151.0/24, 1 successors, FD is 1536, serno 175
        via 172.31.14.1 (1536/768), Serial0/0.1
        via 172.31.24.2 (2560/1792), Serial0/0.2
! Next, the FSM debug is again enabled, and R4 loses neighbor R1.
R4# debug eigrp fsm
Jan 12 07:16:04.099: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 172.31.14.1 (Serial0/0.1)
  is down: holding time expired
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Below, R4 looks for an FS for route 172.31.151.0/24, and does not find one—
! so it enters active state. R4 sends a query to its one remaining neighbor (R2),
! and keeps track of the number of outstanding Queries (1). Upon receiving the
! Reply from R2, it can update its topology table, and repeat local computation,
! and use the now-best route through R2.
Jan 12 07:17:42.391: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 172.31.14.1 (Serial0/0.1)
  is down: holding time expired
```

KEY POINT

Example 9-5 *R1-R4 Link Fails; R4 Actively Queries for 172.31.151.0/24 (Continued)*

```
Jan 12 07:17:42.391: DUAL: linkdown: start - 172.31.14.1 via Serial0/0.1
Jan 12 07:17:42.391: DUAL: Destination 172.31.151.0/24
Jan 12 07:17:42.391: DUAL: Find FS for dest 172.31.151.0/24. FD is 1536, RD is 1536
Jan 12 07:17:42.395: DUAL:  172.31.14.1 metric 4294967295/4294967295
Jan 12 07:17:42.395: DUAL:  172.31.24.2 metric 2560/1792 not found Dmin is 2560
Jan 12 07:17:42.395: DUAL: Dest 172.31.151.0/24 entering active state.
Jan 12 07:17:42.395: DUAL: Set reply-status table. Count is 1.
Jan 12 07:17:42.395: DUAL: Not doing split horizon
Jan 12 07:17:42.459: DUAL: rcvreply: 172.31.151.0/24 via 172.31.24.2 metric 2560/1792
Jan 12 07:17:42.459: DUAL: reply count is 1
Jan 12 07:17:42.459: DUAL: Clearing handle 0, count now 0
Jan 12 07:17:42.463: DUAL: Freeing reply status table
Jan 12 07:17:42.463: DUAL: Find FS for dest 172.31.151.0/24. FD is 4294967295, RD is
  4294967295 found
Jan 12 07:17:42.463: DUAL: Removing dest 172.31.151.0/24, nexthop 172.31.14.1
Jan 12 07:17:42.463: DUAL: RT installed 172.31.151.0/24 via 172.31.24.2
Jan 12 07:17:42.467: DUAL: Send update about 172.31.151.0/24.  Reason: metric chg
Jan 12 07:17:42.467: DUAL: Send update about 172.31.151.0/24.  Reason: new if
! Next, note that because R4 actively queried for the route, the FD could change.
R4# show ip eigrp topo
IP-EIGRP Topology Table for AS(1)/ID(172.31.104.4)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status

P 172.31.151.0/24, 1 successors, FD is 2560
via 172.31.24.2 (2560/1792), Serial0/0.2
```

Of particular note in this example, look for the **debug** message starting with "Dual: rcvreply:" (highlighted). This message means that the router received an EIGRP Reply message, in this case from R2. The message includes R2's valid routing information for 172.31.151.0/24. Also note that the FD was recomputed, whereas it was not in Example 9-4 when an FS route was found.

NOTE Query messages use reliable transmission via RTP and are multicasts; Reply messages are reliable and are unicasts. Both are acknowledged using Ack messages.

NOTE The EIGRP term *Diffusing Update Algorithm (DUAL)* refers to the totality of the logic used by EIGRP to calculate new routes. The term is based on the logic used as Query messages go outward from a router, with the outward movement stopped when routers Reply.

Stuck-in-Active

Any router in active state for a route must wait for a Reply to each of its Query messages. It is possible for a router to wait several minutes for all the replies, because neighboring routers might also need to go active, and then their neighbors might need to go active, and so on—each withholding its Reply message until it in turn receives all of its Reply messages. In normal

operation, the process should complete; to handle exception cases, EIGRP includes a timer called the *Active timer*, which limits the amount of time in which a route can stay active. If the Active timer expires before a router receives all of its Reply messages, the router places the route in a *stuck-in-active state*. The router also brings down any neighbors from which no corresponding Reply was received, thinking that any neighbors that did not send a Reply are having problems.

In some conditions—large, redundant networks, flapping interfaces, or networks with lots of packet loss, to name a few—neighbors might be working fine, but their Reply messages may not complete within the Active timer. To avoid the downside of having the route become stuck-in-active, and losing all routes through a possibly still-working neighbor, you can disable the Active timer by using the **timers active-time disabled** subcommand under **router eigrp**.

Limiting Query Scope

Although disabling the Active timer can prevent stuck-in-active routes, a better solution to the prolonged wait for Reply messages is to limit the scope of Query messages. By reducing the number of neighbors that receive the messages, and by limiting the number of hops away the queries flow, you can greatly reduce the time required to receive all Reply messages.

Two methods can be used to limit query scope. The first is route summarization. When a Query reaches a router that has a summarized route, but not the specific route in the query, the router immediately replies that it does not have that route. For instance, a router with the route 172.31.0.0/16 in its topology table, upon receiving a query for 172.31.151.0/24, immediately sends a Reply, stating it does not have a route to 172.31.151.0/24. With well-designed route summarization, EIGRP queries can be limited to a few hops. (Chapter 11, "IGP Redistribution, Route Summarization, and Default Routing," covers route summarization details.)

The use of EIGRP *stub routers* also limits the query scope. Stub routers, by definition, should not be used as transit routers for traffic. In Figure 9-4, R5 would be a classic candidate to be a stub router. Also, if R4 should not be used to forward traffic from R1 over to R2, or vice versa, R4 could be a stub as well. In either case, non-stub routers do not send Query messages to the stub routers, knowing that the stub routers should not be transit routers. (Stub router configuration is covered in the next section.)

EIGRP Configuration

This section explains the majority of the options for EIGRP configuration. The "Foundation Summary" section includes the full syntax of the commands, along with some comments, in Table 9-6.

EIGRP Configuration Example

Example 9-6 lists the configuration for R1, R2, R4, and R5 from Figure 9-4. The routers were configured based on the following design goals:

- Enable EIGRP on all interfaces.

- Configure K values to ignore bandwidth.
- Configure R5 as an EIGRP stub router.
- Ensure that R2's LAN interface uses a Hello and Hold time of 2 and 6, respectively.
- Configure R4 to allow 75 percent of interface bandwidth for EIGRP updates.
- Advertise R4's LAN subnet, but do not attempt to send or receive EIGRP updates on the LAN.

Example 9-6 *Basic EIGRP Configuration on R1, R2, R4, and R5*

```
! Below, R1 EIGRP-related configuration
! The default metric weights are "0 1 0 1 0 0".
router eigrp 1
 network 172.31.0.0
 metric weights 0 0 0 1 0 0
```

```
! R2 EIGRP-related configuration
! Note the commands used to change the Hello and Hold Time values per interface.
! R2's Hellos advertise the timer values, and other routers on the LAN use these
! values on their neighbor relationship with R2. Also below, note the use of the
! inverse mask to match a subset of interfaces on a single network command.
interface FastEthernet0/0
 ip hello-interval eigrp 1 2
 ip hold-time eigrp 1 6
!
router eigrp 1
 network 10.0.0.0
 network 172.31.11.2 0.0.0.0
 network 172.31.24.0 0.0.1.255
 metric weights 0 0 0 1 0 0
```

```
! R4 EIGRP-related configuration
! Below, the percentage of the interface bandwidth used for EIGRP is changed. The
! value can go over 100% to allow for cases in which the bandwidth has
! been artificially lowered to impact the EIGRP metric. Also note that R4 makes
! its e0/0 interface passive, meaning no routes learned or advertised on E0/0.
interface Serial0/0.1 point-to-point
 bandwidth 64
 ip bandwidth-percent eigrp 1 150
!
router eigrp 1
 passive-interface Ethernet0/0
 network 172.31.0.0
 metric weights 0 0 0 1 0 0
```

```
! R5 EIGRP-related configuration
! Below, note R5's configuration as a stub area.
```

continues

Example 9-6 *Basic EIGRP Configuration on R1, R2, R4, and R5 (Continued)*

```
router eigrp 1
 network 172.31.0.0
 metric weights 0 0 0 1 0 0
 eigrp stub connected summary
```

EIGRP allows for better control of the three functions enabled on an interface by the EIGRP **network** command. (The three functions are advertising the connected subnet, sending routing updates, and receiving routing updates.) Unlike RIP, but like OSPF, the EIGRP **network** command supports configuration of an optional wildcard mask (as seen on R4 in Example 9-6), allowing each interface to be matched individually—and making it simple to enable EIGRP on a subset of interfaces. Also, a LAN subnet might have a single router attached to it, so there is no need to attempt to send or receive updates on those interfaces. By enabling EIGRP on the interface with a **network** command, and then configuring the **passive-interface** command, you can stop the router from sending Hellos. If a router does not send Hellos, it forms no neighbor adjacencies, and it then neither sends nor receives updates on that LAN.

Example 9-6 also shows R5 configured as an EIGRP stub router. R5 announces itself as a stub router via its EIGRP Hellos. As a result, R2 will not send Query messages to R5, limiting the scope of Query messages.

The **eigrp stub** command has several options, with the default options (**connected** and **summary**) shown on the last line of Example 9-6. (Note that the **eigrp stub** command was typed, and IOS added the **connected** and **summary** options in the configuration.) Table 9-4 lists the **eigrp stub** command options, and explains some of the logic behind using them.

Table 9-4 *EIGRP Features Related to Convergence*

KEY POINT

Option	This Router Is Allowed To…
connected	Advertise connected routes, but only for interfaces matched with a **network** command.
summary	Advertise auto-summarized or statically configured summary routes.
static	Advertise static routes, assuming the **redistribute static** command is configured.
redistributed	Advertise redistributed routes, assuming redistribution is configured.
receive-only	Not advertise any routes. This option cannot be used with any other option.

Note that the stub option still requires the stub router to form neighbor relationships, even in receive-only mode. The stub router simply performs less work and reduces the query scope.

Example 9-6 also shows the EIGRP hello interval and hold time being set. These parameters can be set per interface using the interface subcommands **ip hello-interval eigrp** *asn seconds* and

ip hold-time eigrp *asn seconds*, respectively. The default EIGRP hello interval defaults to 5 seconds on most interfaces, with NBMA interfaces whose bandwidth is T1 or slower using a hello interval of 60 seconds. The hold time defaults to 15 and 180 seconds, respectively—three times the default hello interval. However, if you change the hello interval, the hold time default does not automatically change to three times the new hello interval; instead, it remains at 15 or 180 seconds.

EIGRP Load Balancing

EIGRP allows for up to six equal-metric routes to be installed into the IP routing table at the same time. However, because of the complex EIGRP metric calculation, metrics may often be close to each other, but not exactly equal. To allow for metrics that are somewhat close in value to be considered equal, and added to the IP routing table, you can use the **variance** *multiplier* command. The *multiplier* defines a value that is multiplied by the lowest metric (in other words, the FD, which is the metric of the successor route). If any other routes have a better metric than that product of variance * FD, those other routes are considered equal, and added to the routing table.

NOTE EIGRP allows only FS routes to be considered for addition as a result of using the **variance** command. Otherwise, routing loops could occur.

Once the multiple routes for the same destination are in the routing table, EIGRP allows several options for balancing traffic across the routes. Table 9-5 summarizes the commands that impact how load balancing is done with EIGRP, plus the other commands related to installing multiple EIGRP routes into the same subnet. Note that these commands are all subcommands under **router eigrp**.

Table 9-5 *EIGRP Route Load-Balancing Commands*

Router EIGRP Subcommand	Meaning
variance	Any FS route whose metric is less than the variance value multiplied by the FD is added to the routing table (within the restrictions of the **maximum-paths** command).
maximum-paths {1..6}	The maximum number of routes to the same destination allowed in the routing table. Defaults to 4.
traffic-share balanced	The router balances across the routes, giving more packets to lower-metric routes.
traffic-share min	Although multiple routes are installed, sends traffic using only the lowest-metric route.
traffic-share balanced across-interfaces	If more routes exist than are allowed with the **maximum-paths** setting, the router chooses routes with different outgoing interfaces, for better balancing.
No **traffic-share** command configured	Balances evenly across routes, ignoring EIGRP metrics.

EIGRP Configuration Options That Are Similar to RIP

Although EIGRP and RIPv2 differ quite a bit in their underlying operation, several of their features are configured almost identically. This section details these features. You can refer to Chapter 8, "RIP Version 2," for more information on the configuration syntax for these features.

- **Authentication**—EIGRP configures authentication almost exactly like RIP. EIGRP authentication commands use a keyword of **eigrp** *asn* instead of **rip**, using the ASN configured by the **router eigrp** command. For example, the interface subcommand **ip eigrp 1 authentication key-chain carkeys** enables EIGRP MD5 authentication on the interface, pointing to a key chain called "carkeys." Also, EIGRP does not support simple-text authentication, instead defaulting to MD5. Therefore, it does not use an equivalent of the **ip rip authentication mode** {**text** | **md5**} command.

- **Route filtering**—Configured with the **distribute-list** command, EIGRP route filtering is configured identically to RIP route filtering—with one important difference in the underlying operation. With RIP, incoming filters prevent the information from getting into the IP routing table; with EIGRP, an incoming filter prevents topology information from getting into the topology table.

- **Offset lists**—EIGRP uses the same syntax for the **offset-list** command as RIP, but with an interesting underlying bit of logic. EIGRP does not advertise an integer metric, but rather advertises the components of the metric, including constraining bandwidth and cumulative delay. An EIGRP offset list increments only the delay value in the EIGRP metric. For instance, an inbound offset list adding an offset of 1 results in a net increase in the metric of 256, because it increases delay by 1, and EIGRP multiplies the delay setting by 256.

- **Autosummarization**—EIGRP, like RIP, defaults to use auto-summarization; like RIPv2, autosummarization can be disabled with the **no auto-summary** command under **router eigrp**.

- **Split horizon**—EIGRP bounds its updates using split horizon logic, like RIP. Split horizon can be disabled per interface by using the **no ip split-horizon eigrp** *asn* interface subcommand. Note that, like RIP, most interfaces default to split horizon, with the notable exception of a physical serial interface configured for Frame Relay.

- **Clearing IP routing tables**—The **clear ip route *** command clears the IP routing table. However, because EIGRP keeps all possible routes in its topology table, a **clear ip route *** does not cause EIGRP to send any messages or learn any new topology information; the router simply refills the IP routing table with the best routes from the existing topology table. The **clear ip eigrp neighbor** command clears all neighbor relationships, which clears the entire topology table on the router. The neighbors then come back up, send new updates, and repopulate the topology and routing tables. The **clear** command also allows for clearing all neighbors that are reachable out an interface, or based on the neighbor's IP address. (The generic syntax is **clear ip eigrp neighbors** [*ip-address* | *interface-type interface-number*].)

Foundation Summary

This section lists additional details and facts to round out coverage of the topics in this chapter. Unlike most of the Cisco Press *Exam Certification Guides*, this book does not repeat information presented in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 9-6 lists some of the most popular Cisco IOS commands related to the topics in this chapter. Also refer to Table 9-4 for a few additional commands related to load balancing.

Table 9-6 *Command Reference for Chapter 9*

Command	Command Mode and Description
router eigrp *as-number*	Global config; puts user in EIGRP configuration mode for that AS
network *ip-address* [*wildcard-mask*]	EIGRP config mode; defines matching parameters, compared to interface IP addresses, to pick interfaces on which to enable EIGRP
distribute-list [*access-list-number* | *name*] {**in** | **out**} [*interface-type* | *interface-number*]	EIGRP config mode; defines ACL or prefix list to use for filtering EIGRP updates
ip split-horizon eigrp *asn*	Interface subcommand; enables or disables split horizon
passive-interface [**default**] {*interface-type interface-number*}	EIGRP config mode; causes EIGRP to stop sending Hellos on the specified interface, and thereby to also stop receiving and/or sending updates
ip hello-interval eigrp *asn seconds*	Interface subcommand; sets the interval for periodic Hellos sent by this interface
ip hold-time eigrp *asn seconds*	Interface subcommand; sets the countdown timer to be used by a router's neighbor when monitoring for incoming EIGRP messages from this interface
auto-summary	EIGRP config mode; enables automatic summarization at classful network boundaries
metric weights *tos k1 k2 k3 k4 k5*	EIGRP config mode; defines the per-ToS K values to be used in EIGRP metric calculations
ip bandwidth-percent eigrp *asn percent*	Interface subcommand; defines the maximum percentage of interface bandwidth to be used for EIGRP messages

continues

Table 9-6 *Command Reference for Chapter 9 (Continued)*

Command	Command Mode and Description
timers active-time [*time-limit* | **disabled**]	EIGRP config mode; sets the time limit for how long a route is in active state before becoming stuck-in-active
show ip route eigrp *asn*	User mode; displays all EIGRP routes in the IP routing table
show ip eigrp topology [*as-number* | [[*ip-address*] *mask*]] [**active** | **all-links** | **pending** | **summary** | **zero-successors**]	User mode; lists different parts of the EIGRP topology table, depending on the options used
show ip eigrp interfaces [*interface-type interface-number*] [*as-number*]	User mode; lists EIGRP protocol timers and statistics per interface
show ip eigrp traffic [*as-number*]	User mode; displays EIGRP traffic statistics
show ip protocols	User mode; lists EIGRP timer settings, current protocol status, automatic summarization actions, and update sources
show ip eigrp *asn* **neighbors**	User mode; lists EIGRP neighbors
clear ip eigrp neighbors [*ip-address* | *interface-type interface-number*]	Enable mode; disables current neighbor relationships, removing topology table entries associated with each neighbor
clear ip route {*network* [*mask*] | *****}	Enable mode; clears the routing table entries, which are then refilled based on the current topology table
show ip interface [*type number*]	User mode; lists many interface settings, including split horizon
eigrp log-neighbor-changes	EIGRP subcommand; displays log messages when neighbor status changes; enabled by default

Table 9-7 summarizes the types of EIGRP packets and their purposes.

Table 9-7 *EIGRP Message Summary*

KEY POINT

EIGRP Packet	Purpose
Hello	Identifies neighbors, exchanges parameters, and is sent periodically as a keepalive function
Update	Informs neighbors about routing information
Ack	Acknowledges Update, Query, and Response packets
Query	Asks neighboring routers to verify their route to a particular subnet
Reply	Sent by neighbors to reply to a Query
Goodbye	Used by a router to notify its neighbors when the router is gracefully shutting down

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

hello interval, full update, partial update, Route Tag field, Next Hop field, MD5, DUAL, Hold timer, K value, neighbor, adjacency, RTP, SRTT, RTO, Update, Ack, query, Reply, Hello, Goodbye, RD, FD, feasibility condition, successor route, feasible successor, input event, local computation, active, passive, going active, stuck-in-active, query scope, EIGRP stub router, limiting query scope, variance

Refer to the CD-based glossary to check your answers.

Further Reading

Jeff Doyle's *Routing TCP/IP*, Volume I, Second Edition, (Cisco Press) has several excellent examples of configuration, as well as several examples of the DUAL algorithm and the Active Query process.

EIGRP Network Design Solutions, by Ivan Pepelnjak, contains wonderfully complete coverage of EIGRP, at least as it existed at the latest publication date. It also has great, detailed examples of the Query process.

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- IP Routing
  - — OSPF
  - — The use of **show** and **debug** commands

CHAPTER 10

OSPF

This chapter covers OSPF, the only link-state routing protocol covered by the CCIE Routing and Switching exam blueprint. As with the other routing protocol chapters, this chapter includes most of the features, concepts, and commands related to OSPF. Chapter 11 "IGP Route Redistribution, Route Summarization, and Default Routing," covers a few other details of OSPF, in particular, route redistribution, route filtering in redistribution, and route summarization.

"Do I Know This Already?" Quiz

Table 10-1 outlines the major sections in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 10-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
OSPF Database Exchange	1–5	
OSPF Design and LSAs	6–9	
OSPF Configuration	10–12	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. R1 has received an OSPF LSU from R2. Which of the following methods may be used by R1 to acknowledge receipt of the LSU from R2?
   a. TCP on R1 acknowledges using the TCP Acknowledgement field.
   b. R1 sends back an identical copy of the LSU.
   c. R1 sends back an LSAck to R2.
   d. R1 sends back a DD packet with LSA headers whose sequence numbers match the sequence numbers in the LSU.

2. Fredsco has an enterprise network with one core Frame Relay connected router, with a hub-and-spoke network of PVCs connecting to ten remote offices. The network uses OSPF exclusively. The core router (R-core) has all ten PVCs defined under multipoint subinterface s0/0.1. Each remote router also uses a multipoint subinterface. Fred, the engineer, configures an **ip ospf network non-broadcast** command under the subinterface on R-core and on the subinterfaces of the ten remote routers. Fred also assigns an IP address to each router from subnet 10.3.4.0/24, with R-core using the .100 address, and the remote offices using .1 through .10. Assuming all other related options are using defaults, which of the following would be true about this network?

   a. The OSPF hello interval would be 30 seconds.

   b. The OSPF dead interval would be 40 seconds.

   c. The remote routers could learn all routes to other remote routers' subnets, but only if R-core became the designated router.

   d. No designated router will be elected in subnet 10.3.4.0/24.

3. Which of the following interface subcommands, used on a multipoint Frame Relay subinterface, creates a requirement for a DR to be elected for the attached subnet?

   a. **ip ospf network point-to-multipoint**

   b. **ip ospf network point-to-multipoint non-broadcast**

   c. **ip ospf network non-broadcast**

   d. None of these answers is correct.

4. The following routers share the same LAN segment and have the stated OSPF settings: R1: RID 1.1.1.1, hello 10, priority 3; R2: RID 2.2.2.2, hello 9, priority 4; R3, RID 3.3.3.3, priority 3; and R4: RID 4.4.4.4, hello 10, priority 2. The LAN switch fails, recovers, and all routers attempt to elect an OSPF DR and form neighbor relationships at the same time. No other OSPF-related parameters were specifically set. Which of the following is true about negotiations and elections on this LAN?

   a. R1, R3, and R4 will expect Hellos from R2 every 9 seconds.

   b. R2 will become the DR but have no neighbors.

   c. R3 will become the BDR.

   d. R4's dead interval will be 40 seconds.

   e. All routers will use R2's hello interval of 9 once R2 becomes the designated router.

5. Which of the following must be true in order for two OSPF routers that share the same LAN data link to be able to become OSPF neighbors?

   a. Must be in the same area

   b. Must have the same LSRefresh setting

   c. Must have differing OSPF priorities

   d. Must have the same Hello timer, but can have different dead intervals

6. R1 is an OSPF ASBR that injects an E1 route for network 200.1.1.0/24 into the OSPF backbone area. R2 is an ABR connected to area 0 and to area 1. R2 also has an Ethernet interface in area 0, IP address 10.1.1.1/24, for which it is the designated router. R3 is a router internal to area 1. Enough links are up and working for the OSPF design to be working properly. Which of the following is true regarding this topology? (Assume no other routing protocols are running, and that area 1 is not a stub area.)

   a. R1 creates a type 7 LSA and floods it throughout area 0.

   b. R3 will not have a specific route to 200.1.1.0/24.

   c. R2 forwards the LSA that R1 created for 200.1.1.0/24 into area 1.

   d. R2 will create a type 2 LSA for subnet 10.1.1.0/24 and flood it throughout area 0.

7. R1 is an OSPF ASBR that injects an E1 route for network 200.1.1.0/24 into the OSPF backbone area. R2 is an ABR connected to area 0 and to area 1. R2 also has an Ethernet interface in area 0, IP address 10.1.1.1/24, for which it is the designated router. R3 is a router internal to area 1. Enough links are up and working for the OSPF design to be working properly. Which of the following are true regarding this topology? (Assume no other routing protocols are running, and that area 1 is a totally NSSA area.)

   a. R3 could inject internal routes into the OSPF domain.

   b. R3 will not have a specific route to 200.1.1.0/24.

   c. R2 forwards the LSA that R1 created for 200.1.1.0/24 into area 1.

   d. R2 will create a type 2 LSA for subnet 10.1.1.0/24 and flood it throughout area 0.

8. The routers in area 55 all have the **area 55 stub no-summary** command configured under the **router ospf** command. OSPF has converged, with all routers in area 55 holding an identical link-state database for area 55. All IP addresses inside the area come from the range 10.55.0.0/16; no other links outside area 55 use addresses in this range. R11 is the only ABR for the area. Which of the following is true about this design?

   a. The area is a stubby area.

   b. The area is a totally stubby area.

   c. The area is an NSSA.

   d. ABR R11 is not allowed to summarize the type 1 and 2 LSAs in area 55 into the 10.55.0.0/16 prefix due to the **no-summary** keyword.

   e. Routers internal to area 55 can have routes to specific subnets inside area 0.

   f. Routers internal to area 55 can have routes to E1, but not E2, OSPF routes.

9. R1 is an OSPF ASBR that injects an E1 route for network 200.1.1.0/24 into the OSPF backbone area. R2 is an ABR connected to area 0 and to area 1. R2 also has an Ethernet interface in area 0, IP address 10.1.1.1/24, for which it is the designated router. R3 is a router internal to area 1. Enough links are up and working for the OSPF design to be working properly. Which of the following are true regarding this topology? (Assume no other routing protocols are running, and that area 1 is not a stubby area.)

   a. R3's cost for the route to 200.1.1.0 will be the cost of the route as it was injected into the OSPF domain by R1, without considering any internal cost.

   b. R3's cost for the route to 200.1.1.0 will include the addition of R3's cost to reach R1, plus the external cost listed in the LSA.

   c. R3's cost for the route to 10.1.1.0/24 will be the same as its cost to reach ABR R2.

   d. R3's cost for the route to 10.1.1.0/24 will be the sum of its cost to reach ABR R2 plus the cost listed in the type 3 LSA created for 10.1.1.0/24 by ABR R2.

   e. It is impossible to characterize R3's cost to 10.1.1.0/24 because R3 uses a summary type 3 LSA, which hides some of the costs.

10. R1 and R2 each connect via Fast Ethernet interfaces to the same LAN, which should be in area 0. R1's IP address is 10.1.1.1/24, and R2's is 10.1.1.2/24. The only OSPF-related configuration is as follows:

```
hostname R1
router ospf 1
 network 0.0.0.0 255.255.255.255 area 0
 auto-cost reference-bandwidth 1000
!
hostname R2
router ospf 2
 network 10.0.0.0 0.0.0.255 area 0
```

Which of the following statements are true about the configuration?

a. The **network** command on R2 does not match IP address 10.1.1.2, so R2 will not attempt to send Hellos or discover neighbors on the LAN.

b. The different process IDs in the **router ospf** command prevent the two routers from becoming neighbors on the LAN.

c. R2 will become the DR as a result of having a cost of 1 associated with its Fast Ethernet interface.

d. R1 and R2 could never become neighbors due to the difference in cost values.

e. R1's OSPF cost for its Fast Ethernet interface would be 10.

11. Which of the following are true about setting timers with OSPF?

a. The **ip ospf dead-interval minimal hello-multiplier 4** interface subcommand sets the hello interval to 4 ms.

b. The **ip ospf dead-interval minimal hello-multiplier 4** interface subcommand sets the dead interval to 4 seconds.

c. The **ip ospf dead-interval minimal hello-multiplier 4** interface subcommand sets the hello interval to 250 ms.

d. On all interfaces, the **ip ospf hello-interval 30** interface subcommand changes the hello interval from 10 to 30.

e. The **ip ospf hello-multiplier 5** interface subcommand sets the dead interval to five times the then-current hello interval.

f. Cisco IOS defaults the hello and dead intervals to 30/120 on interfaces using the OSPF nonbroadcast network type.

12. R1 has been configured for OSPF authentication on its fa0/0 interface as shown below. Which of the following is true about the configuration?

```
interface fa0/0
 ip ospf authentication-key hannah
 ip ospf authentication
 ip ospf message-digest-key 2 md5 jessie
router ospf 2
 area 0 authentication message-digest
```

a. R1 will attempt simple-text authentication on the LAN with key **hannah**.

b. R1 will attempt MD5 authentication on the LAN with key **jessie**.

c. R2 will attempt OSPF type 2 authentication on fa0/0.

d. R2 will attempt OSPF type 3 authentication on fa0/0.

Foundation Topics

Link-state routing protocols define the content and structure of data that describes network topology, and define the processes by which routers exchange that detailed topology information. The name "link state" refers to the fact that the topology information includes information about each data *link*, along with each link's current operational *state*. All the topological data together comprises the *link-state database (LSDB)*. Each link-state router applies the Dijkstra algorithm to the database to calculate the current-best routes to each subnet.

This chapter breaks down the OSPF coverage into three major sections. The first section details how the topology data is exchanged. The second section covers OSPF design and the contents of the LSDB, which comprises different types of *link-state advertisements (LSAs)*. (The second section covers both design and the LSDB because the design choices directly impact which types of LSAs are forwarded into the differing parts of an OSPF network.) The third section covers the majority of the OSPF configuration details of OSPF for this chapter, although a few configuration topics are interspersed in the first two sections.

NOTE This chapter addresses the functions of OSPF Version 2. It ignores OSPF Version 3 (RFC 2740), which was introduced primarily to support IPv6.

OSPF Database Exchange

OSPF defines five different messages that routers can use to exchange LSAs. The process by which LSAs are exchanged does not change whether a single area or multiple areas are used, so this section will use a single OSPF area (area 0).

OSPF Router IDs

Before an OSPF router can send any OSPF messages, it must choose a unique 32-bit dotted-decimal identifier called the OSPF *router identifier (RID)*. Cisco routers use the following sequence to choose their OSPF RID, only moving on to the next step in this list if the previous step did not supply the OSPF RID:

KEY POINT

1. Use the router ID configured in the **router-id** *id* subcommand under **router ospf**.
2. Use the highest numeric IP address on any currently "up and up" loopback interface.
3. Use the highest numeric IP address on any currently "up and up" non-loopback interface.

The sequence and logic are very simple, but some details are hidden in the sequence:

- The interface from which the RID is taken does not have to be matched by an OSPF **network** command.
- OSPF does not have to advertise a route to reach the RID's subnet.
- The RID does not have to be reachable per the IP routing table.
- Steps 2 and 3 look at the then-current interface state to choose the RID when the OSPF process is started.
- Routers consider changing the OSPF RID when the OSPF process is restarted, or when the RID is changed via configuration.
- If a router's RID changes, the rest of the routers in the same area will have to perform a new SPF calculation.
- If the RID is configured with the **router-id** command, and the command remains unchanged, that router's RID will never change.

For these reasons, many people set their RIDs with the **router-id** command and use an obvious numbering scheme to make it easy to identify a router by its RID.

Becoming Neighbors, Exchanging Databases, and Becoming Adjacent

OSPF directly encapsulates the five different types of OSPF messages inside IP packets, using IP protocol 89, as listed in Table 10-2.

Table 10-2 *OSPF Messages*

KEY POINT

Message	Description
Hello	Used to discover neighbors, bring a neighbor relationship to a 2-way state, and monitor a neighbor's responsiveness in case it fails
Database Description (DD or DBD)	Used to exchange brief versions of each LSA, typically on initial topology exchange, so that a router knows a list of that neighbor's LSAs
Link-State Request (LSR)	A packet that identifies one or more LSAs about which the sending router would like the neighbor to supply full details about the LSAs
Link-State Update (LSU)	A packet that contains fully detailed LSAs, typically sent in response to an LSR message
Link-State Acknowledgement (LSAck)	Sent to confirm receipt of an LSU message

These messages together allow routers to discover each other's presence (Hello), learn which LSAs are missing from their LSDBs (DD), request and reliably exchange the LSAs (LSR/LSU), and monitor their neighbors for any changes in the topology (Hello). Note that the LSAs themselves are not OSPF messages—an LSA is a data structure, held inside a router's LSDB, and exchanged inside LSU messages.

When a particular data link first comes up, OSPF routers first become neighbors using the Hello message. At that point, they exchange topology information using the other four OSPF messages. Figure 10-1 outlines the overall process between two routers.

Figure 10-1 *Overview of OSPF LSDB Exchange*

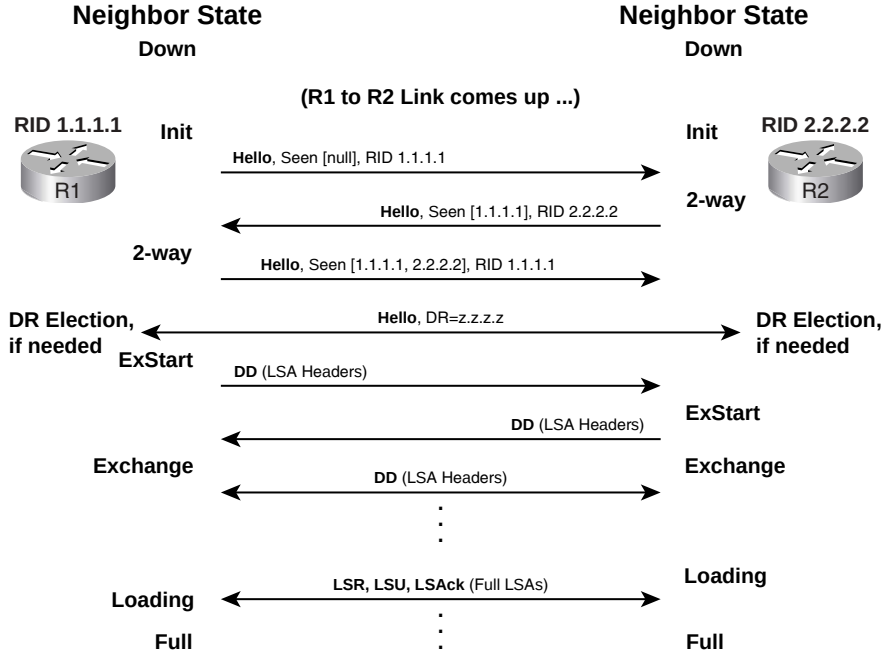


Figure 10-1 shows the overall message flow, along with the *neighbor state* on each router. An OSPF router keeps a state machine for each neighbor, listing the current neighbor state in the output of the **show ip ospf neighbor** command. These neighbor states change as the neighbors progress through their messaging; in this example, the neighbors settle into a *full state*, meaning *fully adjacent*, once the process is completed.

The "Foundation Summary" section at the end of this chapter includes a reference table (Table 10-13) listing the neighbor states and their meanings. The next few sections explain the details behind the process shown in Figure 10-1.

Becoming Neighbors: The Hello Process

Hello messages perform three major functions:

- Discover other OSPF-speaking routers on common subnets
- Check for agreement on some configuration parameters
- Monitor health of the neighbors to react if the neighbor fails

To discover neighbors, Cisco OSPF routers listen for multicast Hello messages sent to 224.0.0.5—the *All OSPF Routers* multicast address—on any interfaces that have been enabled for OSPF. The Hellos are sourced from that router's primary IP address on the interface—in other words, Hellos are not sourced from secondary IP addresses. (OSPF routers will advertise secondary IP addresses, but they will not send Hellos from those IP addresses, and never form neighbor relationships using secondary addresses.)

Once two routers discover each other by receiving Hellos from the other router, the routers perform the following parameter checks based on the receive Hellos:

KEY POINT

- Must pass the authentication process
- Must be in the same primary subnet, including same subnet mask
- Must be in the same OSPF area
- Must be of the same area type (stub, NSSA, and so on)
- Must not have duplicate RIDs
- OSPF Hello and Dead timers must be equal

If any of these items do not match, the two routers simply do not form a neighbor relationship. Also of note is one important item that does not have to match: the OSPF process ID (PID), as configured in the **router ospf** *process-id* command. (Also, the MTU must be equal for the DD packets to be successfully sent between neighbors, but this parameter check is technically not part of the Hello process.)

The third important function for a Hello is to maintain a heartbeat function between neighbors. The neighbors send Hellos every *hello interval*; failure to receive a Hello within the longer *dead interval* causes a router to believe that its neighbor has failed. The hello interval defaults to 10 seconds on LAN interfaces and 30 seconds on T1 and slower WAN interfaces; the dead interval defaults to four times the hello interval.

Example 10-1 lists some basic OSPF command output related to the neighbor establishment with Hellos, and the hello and dead intervals.

Example 10-1 *Hello Mismatches and Basic Neighbor Parameters*

```
! Below, debug messages show that this router disagrees with the hello and dead
! intervals on router 10.1.111.4; The "C" and "R" mean "configured" and "received,"
! respectively, meaning that this router uses 30/120 for hello/dead, and the other
! router is trying to use 10/40.
R1# debug ip ospf hello
OSPF hello events debugging is on
Jan 12 06:41:20.940: OSPF: Mismatched hello parameters from 10.1.111.4
Jan 12 06:41:20.940: OSPF: Dead R 40 C 120, Hello R 10 C 30  Mask R 255.255.255.0 C
  255.255.255.0
! Below, R1's hello and dead intervals are listed for the same interface.
R1# show ip ospf int s 0/0.100
Serial0/0.100 is up, line protocol is up
  Internet Address 10.1.111.1/24, Area 0
  Process ID 1, Router ID 1.1.1.1, Network Type NON_BROADCAST, Cost: 64
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 1.1.1.1, Interface address 10.1.111.1
  No backup designated router on this network
  Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit 5
! Lines omitted for brevity
! Below, R1 shows a neighbor on S0/0.100, in the full state, meaning the routers
! have completed LSDB exchange. Note the current Dead timer counts down, in this
! case from 2 minutes; the value of 1:58 means R1 last received a Hello from
! neighbor 10.1.111.6 two seconds ago.
R1# sh ip ospf neighbor 6.6.6.6
 Neighbor 6.6.6.6, interface address 10.1.111.6
    In the area 0 via interface Serial0/0.100
    Neighbor priority is 0, State is FULL, 8 state changes
    DR is 10.1.111.1 BDR is 0.0.0.0
    Poll interval 120
    Options is 0x42
    Dead timer due in 00:01:58
    Neighbor is up for 00:17:22
! Lines omitted for brevity
```

Flooding LSA Headers to Neighbors

Once two routers hear Hellos, and the parameter check passes, they do not immediately send packets holding the LSAs. Instead, each router creates and sends *Database Description* (*DD*, or sometimes called *DBD*) packets, which contain the headers of each LSA. The headers include enough information to uniquely identify each LSA. Essentially, the routers exchange a list of all the LSAs they each know about; the next step in the process is letting a router request a new copy of any old or unknown LSAs.

The DD messages use an OSPF-defined simple error-recovery process. Each DD packet, which may contain several LSA headers, has an assigned sequence number. The receiver acknowledges

a received DD packet by sending an identical DD packet back to the sender. The sender uses a window size of one packet, then waits for the acknowledgement before sending the next DD packet.

Requesting, Getting, and Acknowledging LSAs

Once all LSA headers have been exchanged using DD packets, each neighboring router has a list of LSAs known by the neighbor. Using that knowledge, a router needs to request a full copy of each LSA that is missing from its LSDB.

To know if a neighbor has a more recent copy of a particular LSA, a router looks at the sequence number of the LSA in its LSDB and compares it to the sequence number of that same LSA learned from the DD packet. Each LSA's sequence number is incremented every time the LSA changes. So, if a router received (via a DD packet) an LSA header with a later sequence number for a particular LSA (as compared with the LSA in the LSDB), that router knows that the neighbor has a more recent LSA. For example, R1 sent R2 an LSA header for the type 1 LSA that describes R1 itself, with sequence number 0x80000004. If R2's database already held that LSA, but with a sequence number of 0x80000003, then R2 would know that it needs to ask R1 to send the latest copy (sequence number 0x80000004) of that LSA.

NOTE New LSAs begin with sequence number 0x80000001, increase, and then wrap back to 0x7FFFFFFF. If the LSA made it to sequence number 0x80000000, the LSA must be reflooded throughout the network.

Routers use *Link-State Request (LSR)* packets to request one or more LSAs from a neighbor. The neighboring router replies with *Link-State Update (LSU)* packets, which hold one or more full LSAs. As shown in Figure 10-1, both routers sit in a loading state while the LSR/LSA process continues. Once the process is complete, they settle into a *full* state, which means that the two routers should have fully exchanged their databases, resulting in identical copies of the LSDB entries for that area on both routers.

The LSR/LSA process uses a reliable protocol that has two options for acknowledging packets. First, an LSU can be acknowledged by the receiver of the LSU simply repeating the exact same LSU back to the sender. Alternatively, a router can send back an *LSAck* packet to acknowledge the packet, which contains a list of acknowledged LSA headers.

At the end of the process outlined in Figure 10-1, two neighbors have exchanged their LSDBs. As a result, their LSDBs should be identical. At this point, they can each independently run the Dijkstra Shortest Path First (SPF) algorithm to calculate the best routes from their own perspectives.

Designated Routers on LANs

OSPF optimizes the LSA flooding process on multiaccess data links by using the concept of a *designated router (DR)*. Without the concept of a DR, each pair of routers that share a data link would become fully adjacent neighbors. Each pair of routers would directly exchange their LSDBs with each other as shown in Figure 10-1. On a LAN with only six routers, without a DR, 15 different pairs of routers would exist, and 15 different instances of full database flooding would occur. OSPF uses a DR (and *backup DR*, or *BDR*) on a LAN or other multiaccess network. The flooding occurs through the DR, significantly reducing the unnecessary exchange of redundant LSAs.

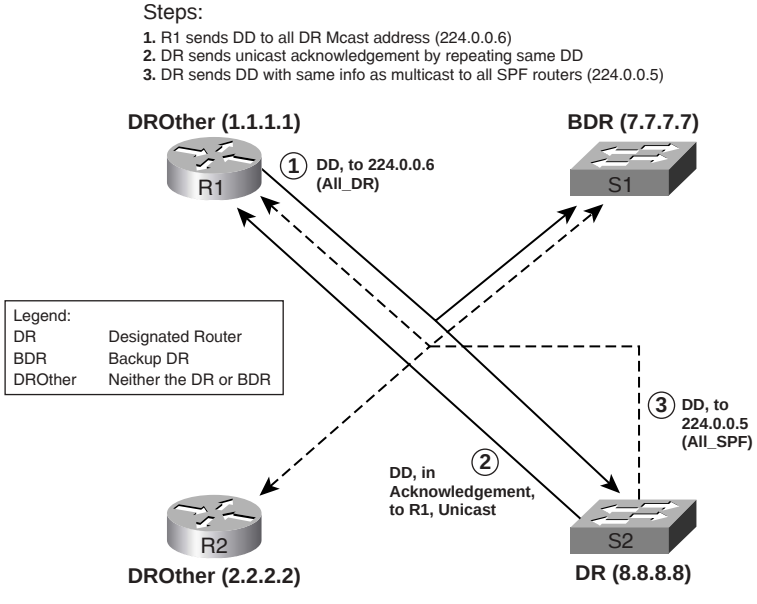
NOTE DRs have one other major function besides improving the efficiency of LSA flooding process. They also create a type 2 LSA that represents the subnet. LSA types are covered in the next major section, "OSPF Design and LSAs."

The next section goes through the basics of the DR/BDR process on LANs, which is followed by coverage of options of OSPF network types and how they impact OSPF flooding on Frame Relay links.

Designated Router Optimization on LANs

Figure 10-2 depicts the DR flooding optimization that occurs with sending DD packets over a LAN.

Figure 10-2 *DR Optimization on a LAN*



Routers that are not the DR (including the BDR) send DDs to the DR by sending them to multicast address 224.0.0.6, the All OSPF DR Routers multicast address. The DR then acknowledges the DDs with a unicast DD (Step 2 in Figure 10-2). The DR then floods a new DD packet to all OSPF routers (multicast address 224.0.0.5).

Figure 10-2 shows the three main steps, but the non-DR routers also need to acknowledge the DD packet sent in Step 3. Typically, the acknowledgment occurs by the other routers each replying with a unicast DD packet.

NOTE In topologies without a DR, the DD and LSU packets are typically sent to the 224.0.0.5. All OSPF Routers multicast IP address.

Example 10-2 shows the output of a **show ip ospf neighbor** command on R1 from Figure 10-2. Note that R1 is in a full state with S2, which is the DR, with OSPF RID 8.8.8.8. R1 is also in a full state with S1, the BDR, OSPF RID 7.7.7.7. However, R1 is in a 2WAY state with R2, RID 2.2.2.2.

Example 10-2 *The* **show ip ospf neighbor** *Command*

```
R1# sh ip ospf neighbor fa 0/0
Neighbor ID     Pri   State           Dead Time   Address         Interface
2.2.2.2           1   2WAY/DROTHER    00:00:35    10.1.1.2        FastEthernet0/0
7.7.7.7           1   FULL/BDR        00:00:38    10.1.1.3        FastEthernet0/0
8.8.8.8           1   FULL/DR         00:00:34    10.1.1.4        FastEthernet0/0
```

When a DR is used on a link, routers end up as DR, BDR, or neither; a router that is neither DR or BDR is called a *DROther* router. The DR and BDR form full adjacencies with all other neighbors on the link, so they reach a full state once the database exchange process is complete. However, two neighbors that are both DROthers do not become fully adjacent—they stop at the 2WAY state, as shown in Example 10-2. Stopping at the 2WAY state between two DROther routers is normal; it simply means that the Hello parameter-match check worked, but the neighbors do not need to proceed to the point of exchanging DD packets, because they do not need to when a DR is present.

To describe the fact that some neighbors do not directly exchange DD and LSU packets, OSPF makes a distinction between the terms *neighbors* and *adjacent*, as follows:

- **Neighbors**—Two routers that share a common data link, that exchange Hello messages, and the Hellos must match for certain parameters.
- **Adjacent (fully adjacent)**—Two neighbors that have completed the process of fully exchanging DD and LSU packets directly between each other.

Note that although DROther routers do not exchange DD and LSU packets directly with each other, like R1 and R2 in Figure 10-2, the DROther routers do end up with an identical copy of the LSDB entries by exchanging them with the DR.

DR Election on LANs

As noted in Figure 10-1, if a DR is elected, the election occurs after the routers have become neighbors, but before they send DD packets and reach the ExStart neighbor state. When an OSPF router reaches the 2-way state with the first neighbor on an interface, it has already received at least one Hello from that neighbor. If the Hello messages state a DR of 0.0.0.0—meaning none has been elected—the router waits before attempting to elect a DR. This typically occurs after a failure on the LAN. OSPF routers wait with the goal of giving all the routers on that subnet a chance to finish initializing after a failure so that all the routers can participate in the DR election—otherwise, the first router to become active would always become the DR. (The time period is called the OSPF *wait time*, which is set to the same value as the Dead timer.)

However, if the received Hellos already list the DR's RID, the router does not have to wait before beginning the election process. This typically occurs when one router lost its connection to the LAN, but other routers remained and continued to work. In this case, the newly-connected router does not attempt to elect a new DR, assuming the DR listed in the received Hello is indeed the current DR.

The election process allows for the possibility of many different scenarios for which routers may and may not become the DR or BDR. Generally speaking, the following rules govern the DR/BDR election process:

KEY POINT

- Any router with its OSPF priority set to between 1–255 inclusive can try to become DR by putting its own RID into the DR field of its sent Hellos.
- Routers examine received Hellos, looking at other routers' priority settings, RIDs, and whether each neighbor claims to want to become the DR.
- If a received Hello implies a "better" potential DR, the router stops claiming to want to be DR and asserts that the better candidate should be the DR.
- The first criteria for "better" is the router with the highest priority.
- If the priorities tie, the router with the higher RID is better.
- The router not claiming to be the DR, but with the higher priority (or higher RID, in case priority is a tie) becomes the BDR.

- If a new router arrives after the election, or an existing router improves its priority, it cannot preempt the existing DR and take over as DR (or as BDR).
- Once a DR is elected, and the DR fails, the BDR becomes DR, and a new election is held for a new BDR.

Once the DR is elected, LSA flooding continues as illustrated previously in Figure 10-2.

Designated Routers on WANs and OSPF Network Types

Using a DR makes good sense on a LAN because it improves LSA flooding efficiency. Likewise, not using a DR on a point-to-point WAN link also makes sense, because with only two routers on the subnet, there is no inefficiency upon which to improve. However, on nonbroadcast multiaccess (NBMA) networks, arguments can be made regarding whether a DR is helpful. So, OSPF includes several options that include a choice of whether to use a DR on WAN interfaces.

Cisco router interfaces can be configured to use, or not use, a DR, plus a couple of other key behaviors, based on the *OSPF network type* for each interface. The OSPF network type determines that router's behavior regarding the following:

KEY POINT

- Whether the router tries to elect a DR on that interface
- Whether the router must statically configure a neighbor (with the **neighbor** command), or find neighbors using the typical multicast Hello packets
- Whether more than two neighbors should be allowed on the same subnet

For instance, LAN interfaces default to use an OSPF network type of *broadcast*. OSPF broadcast networks elect a DR, use Hellos to dynamically find neighbors, and allow more than two routers to be in the same subnet on that LAN. For HDLC and PPP links, OSPF uses a network type of *point-to-point*, meaning that no DR is elected, only two IP addresses are in the subnet, and neighbors can be found through Hellos.

Table 10-3 summarizes the OSPF interface types and their meanings. Note that the interface type values can be set with the **ip ospf network** *type* interface subcommand; the first column in the table lists the exact keyword according to this command. Also, for cases in which a DR is not elected, all routers that become neighbors also attempt to become adjacent by the direct exchange of DD, LSR, and LSU packets.

Table 10-3 *OSPF Network Types*

KEY POINT

Interface Type	Uses DR/ BDR?	Default Hello Interval	Requires a neighbor Command?	More than 2 Hosts Allowed in the Subnet?
Broadcast	Yes	10	No	Yes
Point-to-point[1]	No	10	No	No
Nonbroadcast[2] (NBMA)	Yes	30	Yes	Yes
Point-to-multipoint	No	30	No	Yes
Point-to-multipoint nonbroadcast	No	30	Yes	Yes
Loopback	No	—	—	No

[1] Default on Frame Relay point-to-point subinterfaces.

[2] Default on Frame Relay physical and multipoint subinterfaces.

Caveats Regarding OSPF Network Types over NBMA Networks

When configuring OSPF over Frame Relay, the OSPF network type concept can become a bit troublesome. In fact, many CCIE Routing and Switching lab preparation texts and lab books focus on the variety of combinations of OSPF network types used with Frame Relay for various interfaces/subinterfaces. The following list contains many of the key items you should check when looking at an OSPF configuration over Frame Relay, when the OSPF network types used on the various routers do not match:

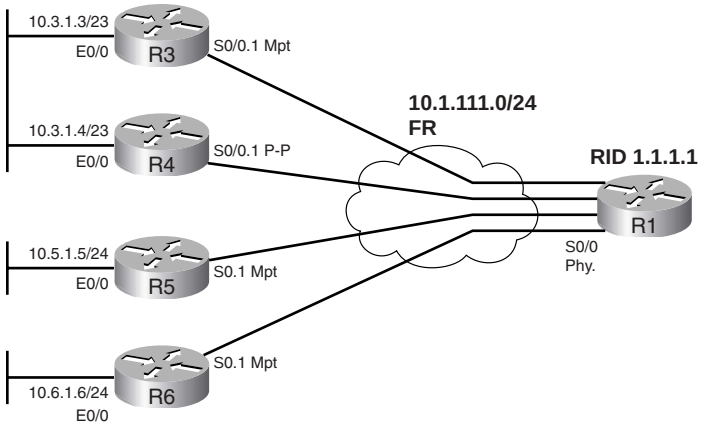
- Make sure the default Hello/Dead timers do not cause the Hello parameter check to fail. (See Table 10-3 for the defaults for each OSPF network type.)
- If one router expects a DR to be elected, and the other does not, the neighbors may come up, and full LSAs be communicated. However, **show** command output may show odd information, and next-hop routers may not be reachable. So, make sure all routers in the same NBMA subnet use an OSPF network type that either does use a DR or does not.
- If a DR is used, the DR and BDR must have a permanent virtual circuit (PVC) to each other router in the subnet. If not, not all routers will be able to learn routes, because the DR must forward the DD and LSU packets to each of the other routers. Routers without a PVC to each other router should be prevented from becoming a DR/BDR.
- If one router requires a static **neighbor** command, typically the other router on the other end of the PVC does not require a **neighbor** command. For clarity, however, it is better to configure **neighbor** commands on both routers.

Two very simple options exist for making OSPF work over Frame Relay—both of which do not require a DR and do not require **neighbor** commands. If the design allows for the use of point-to-point subinterfaces, use those, take the default OSPF network type of point-to-point, and no additional work is required. If multipoint subinterfaces are needed, or if the configuration must not use subinterfaces, adding the **ip ospf network point-to-multipoint** command on all the routers works, without requiring additional effort to manually define neighbors or worry about which router becomes the DR.

Example of OSPF Network Types and NBMA

On NBMA networks with an OSPF network type that requires that a DR be elected, you must take care to make sure the correct DR is elected. The reason is that the DR and BDR must each have a PVC connecting it to all the DROther routers—otherwise, LSA flooding will not be possible. So, with partial meshes, the election should be influenced by configuring the routers' priority and RIDs such that the hub site of a hub-and-spoke partial mesh becomes the DR. Figure 10-3 shows an example network for which R1 should be the only router allowed to become DR or BDR.

Figure 10-3 *Network Used in the Frame Relay Priority and Network Type Example*



Example 10-3 depicts the following scenarios relating to DR election in Figure 10-3:

- The R1, R3, and R5 configuration is correct for operating with default OSPF network type nonbroadcast in a partial mesh.
- R6 has omitted the **ip ospf priority** interface subcommand, causing it to inadvisably become the DR.
- R4 will be used as an example of what not to do, in part to point out some interesting facts about OSPF **show** commands.

NOTE Figure 10-3 and Example 10-3 do not depict a suggested design for Frame Relay and OSPF. With this topology, using point-to-point subinterfaces in all cases, using four small (/30) subnets, and defaulting to OSPF network type point-to-point would work well. Such a design, however, would not require any thought regarding the OSPF network type. So, this example is purposefully designed to provide a backdrop from which to show how the OSPF network types work.

Example 10-3 shows only the nondefault OSPF configuration settings; also, the routers have an obvious RID numbering scheme (1.1.1.1 for R1, 2.2.2.2 for R2, and so on).

Example 10-3 *Setting Priority on NBMA Networks*

```
! R1 configuration—the neighbor commands default to a priority value of 0,
! meaning R1's perception of that neighbor is priority 0.
router ospf 1
 log-adjacency-changes detail
 network 0.0.0.0 255.255.255.255 area 0
 neighbor 10.1.111.3
 neighbor 10.1.111.4
 neighbor 10.1.111.5
 neighbor 10.1.111.6
```

```
! R3 configuration—R3's interface priority is set to 0; R1 will use the higher
! of R3's announced priority 0 (based on R3's ip ospf priority interface
! subcommand) and the priority value on R1's neighbor command, which defaulted
! to 0. So, R3 will not ever become a DR/BDR.
interface Serial0/0.1 multipoint
 ip address 10.1.111.3 255.255.255.0
 ip ospf priority 0
 frame-relay interface-dlci 100
```

```
! R4 configuration—note from Figure 10-3 that R4 is using a point-to-point
! subinterface, with all defaults. This is not a typical use of a point-to-point
! subinterface, and is shown to make a few points later in the example.
router ospf 1
 network 0.0.0.0 255.255.255.255 area 0
```

```
! R5's configuration is equivalent to R3 in relation to the OSPF network type
! and its implications.
interface Serial0.1 multipoint
 ip address 10.1.111.5 255.255.255.0
 ip ospf priority 0
 frame-relay interface-dlci 100
!
router ospf 1
 network 0.0.0.0 255.255.255.255 area 0
```

```
! R6 configuration—R6 forgot to set the interface priority with the ip ospf
! priority 0 command, defaulting to priority 1.
router ospf 1
 network 0.0.0.0 255.255.255.255 area 0
```

Example 10-3 *Setting Priority on NBMA Networks (Continued)*

```
! Below, the results of R6's default interface priority of 1—R6, with RID
! 6.6.6.6, and an announced priority of 1, wins the DR election. Note that the
! command is issued on R1.
R1# show ip ospf neighbor
Neighbor ID     Pri   State           Dead Time   Address         Interface
6.6.6.6           1   FULL/DR         00:01:52    10.1.111.6      Serial0/0
3.3.3.3           0   FULL/DROTHER    00:01:46    10.1.111.3      Serial0/0
N/A               0   ATTEMPT/DROTHER   —         10.1.111.4       Serial0/0
5.5.5.5           0   FULL/DROTHER    00:01:47    10.1.111.5      Serial0/0
! Next, R1's neighbor command was automatically changed to "priority 1" based on
! the Hello, with priority 1, that R1 received from R6. To prevent this dynamic
! reconfiguration, you could add an ip ospf priority 0 command under R6's s0/0.1
! interface.
R1# show run | beg router ospf 1
router ospf 1
 network 0.0.0.0 255.255.255.255 area 0
 neighbor 10.1.111.6 priority 1
 neighbor 10.1.111.3
 neighbor 10.1.111.4
 neighbor 10.1.111.5
! lines omitted for brevity
```

```
! Below, R4 is OSPF network type "point to point," with Hello/dead of 10/40.
! R1's settings, based on Table 10-3, would be nonbroadcast, 30/120.
R4# show ip ospf int s 0/0.1
Serial0/0.1 is up, line protocol is up
  Internet Address 10.1.111.4/24, Area 0
  Process ID 1, Router ID 4.4.4.4, Network Type POINT_TO_POINT, Cost: 1562
  Transmit Delay is 1 sec, State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
! lines omitted for brevity
! Below, R4 changes its network type to yet a different value, one that expects
! neighbor commands, but does not expect a DR to be used.
R4# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R4(config)# int s 0/0.1
R4(config-subif)# ip ospf network point-to-multipoint non-broadcast
```

```
! Next, R1 and R4 become neighbors now that the Hello parameters match. Note that
! R1 believes that R4 is DROther.
R1# show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
! lines omitted for brevity
4.4.4.4           1   FULL/DROTHER    00:01:56    10.1.111.4      Serial0/0
! Below, R4 agrees it is in a full state with R1, but does not list R1 as DR,
! because R4 is not using the concept of a DR at all due to R4's network type.
R4# sh ip ospf neigh
Neighbor ID     Pri   State           Dead Time   Address         Interface
1.1.1.1           0   FULL/ —       00:01:42    10.1.111.1      Serial0/0.1
```

The first and most important point from Example 10-3 is the actual behavior of the two ways to set the priority in the example. The *Cisco IOS Configuration Guide* at Cisco.com states that the OSPF **neighbor** command defines the priority of the neighbor. However, in practice, a router's **neighbor priority** setting is compared with the priority inside the Hello it receives from that neighbor—and the larger of the two values is used. In this example, R1's **neighbor 10.1.111.6** command (with default priority of 0) was overridden by R6's Hello, which was based on R6's default OSPF interface priority of 1. So, during DR election, R1 and R6 tied on OSPF priority, and R6 won due to its larger (6.6.6.6 versus 1.1.1.1) RID. R1 even automatically changed its **neighbor** command dynamically to **neighbor 10.1.111.6 priority 1** to reflect the correct priority for R6.

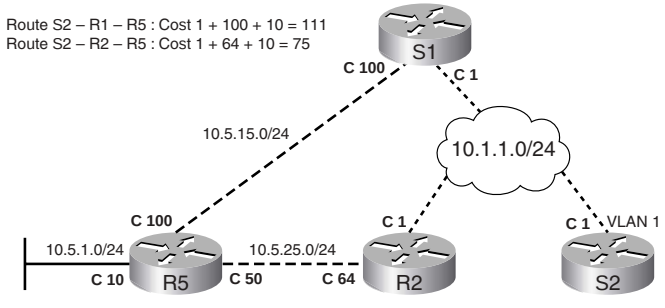
Also note that, although neighbors must be statically configured for some network types, the **neighbor** command needs to be configured on only one router. R3 and R5, with correct working configurations, did not actually need a **neighbor** command.

Finally, it might seem that all is now fine between R1 and R4 by the end of the example, but even though the neighbors are fully adjacent, R4 cannot route packets to R3, R5, or R6 over the Frame Relay network. For instance, R5 could have some routes that point to 10.1.111.4 (R4's Frame Relay IP address) as the next hop. However, because R5 is using a multipoint subinterface, R5 will not know what PVC to use to reach 10.1.111.4. (Chapter 7, "IP Forwarding (Routing)," covers how Frame Relay mapping occurs, and the logic used on multipoint and point-to-point subinterfaces.) In this case, the routers with multipoint subinterfaces would need to add **frame-relay map** commands; for example, R5 would need a **frame-relay map ip 10.1.111.4 100 broadcast** command, causing packets to next-hop 10.1.111.4 to go over DLCI 100 to R1, which would then route the packet on to R4. Keep in mind that R4's configuration is not a recommended configuration.

SPF Calculation

So far, this chapter has covered a lot of ground related to the exchange of LSAs. Regardless of the OSPF network type and whether DRs are used, once a router has new or different information in its LSDB, it uses the Dijkstra SPF algorithm to examine the LSAs in the LSDB and derive the math-equivalent of a figure of a network. This mathematical model has routers, links, costs for each link, and the current (up/down) status of each link. Figure 10-4 represents the SPF model of a sample network.

Figure 10-4 *Single-Area SPF Calculation: Conceptual View*



Humans can easily see the conclusion that the SPF algorithm will reach, even though the algorithm itself is fairly complicated. SPF on a router finds all possible routes to each subnet, adds the cost for each *outgoing* interface in that route, and then picks the path with the least cost. OSPF then places those least (shortest) cost routes into the routing table. For example, S2 calculates two possible routes to subnet 10.5.1.0/24, with the better route being out S2's VLAN 1 interface, with R2 as the next-hop router. Also note in Figure 10-4 that the cost values are per interface, and it is each outgoing interface's cost that SPF adds to come up with the total cost of the route.

Steady-State Operation

Even after a network has stabilized, all routers in the same area have the exact same LSAs, and each router has chosen its best routes using SPF, the following is still true of routers running OSPF:

- Each router sends Hellos, based on per-interface hello intervals.
- Each router expects to receive Hellos from neighbors within the dead interval on each interface; if not, the neighbor is considered to have failed.
- Each router originally advertising an LSA refloods each LSA (after incrementing its sequence number by 1) based on a per-LSA Link-State Refresh (LSRefresh) interval (default 30 minutes).
- Each router expects to have its LSA refreshed within each LSA's MaxAge timer (default 60 minutes).

OSPF Design and LSAs

This section covers two major topics:

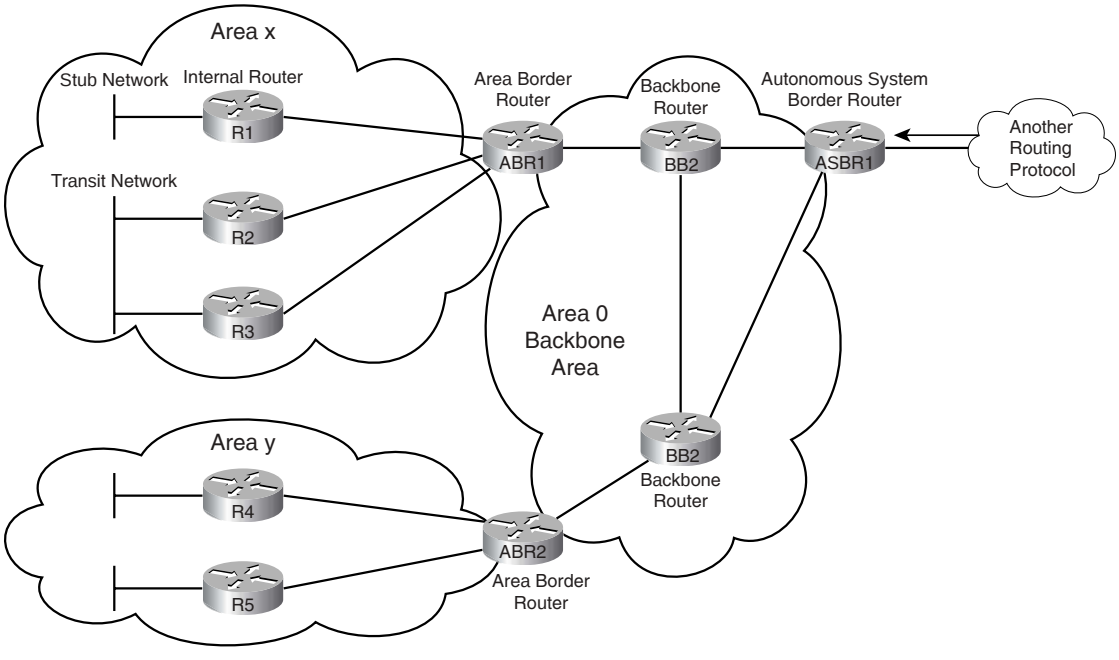
- OSPF design
- OSPF LSA types

Although these might seem to be separate concepts, most OSPF design choices directly impact the LSA types in a network and impose restrictions on which neighbors may exchange those LSAs. This section starts with an OSPF design and terminology review, and then moves on to LSA types. Toward the end of the section, OSPF area types are covered, including how each variation changes how LSAs flow through the different types of OSPF stubby areas.

OSPF Design Terms

OSPF design calls for grouping links into contiguous areas. Routers that connect to links in different areas are *Area Border Routers (ABRs)*. ABRs must connect to area 0, the *backbone area*, and one or more other areas as well. *Autonomous System Boundary Routers (ASBRs)* inject routes external to OSPF into the OSPF domain, having learned those routes from wide-ranging sources from the Border Gateway Protocol (BGP) on down to simple redistribution of static routes. Figure 10-5 shows the terms in the context of a simple OSPF design.

Figure 10-5 *OSPF Design Terminology*



Networks can use a single OSPF area, but using OSPF areas helps speed convergence and reduce overhead in an OSPF network. Using areas provides the following benefits:

KEY POINT

- Generally smaller per-area LSDBs, requiring less memory.
- Faster SPF computation due to the sparser LSDB.
- A link failure in one area only requires a partial SPF computation in other areas.
- Routes may only be summarized at ABRs (and ASBRs); having areas allows summarization, again shrinking the LSDB and improving SPF calculation performance.

When comparing the use of one area versus using many areas, the number of routers or subnets does not shrink, but the size of the LSDB on most routers should shrink. The LSDB shrinks because an ABR does not pass denser and more detailed type 1 and 2 LSAs from one area to another—instead, it passes type 3 summary LSAs. LSA types 1 and 2 can be thought of as the detailed topology information that causes most of the computing-intensive parts of the SPF algorithm; by representing these detailed type 1 and 2 LSAs in a different way in other areas, OSPF achieves its goal of reducing the effects of SPF.

LSA Types and Network Types

Table 10-4 lists the LSA types and their descriptions for reference; following the table, each type is explained in more detail, in the context of a working network.

Table 10-4 *OSPF LSA Types*

KEY POINT

LSA Type	Common Name	Description
1	Router	One per router, listing RID and all interface IP addresses. Represents stub networks as well.
2	Network	One per transit network. Created by the DR on the subnet, and represents the subnet and the router interfaces connected to the subnet.
3	Net Summary	Created by ABRs to represent one area's type 1 and 2 LSAs when being advertised into another area. Defines the links (subnets) in the origin area, and cost, but no topology data.
4	ASBR Summary	Like a type 3 LSA, except it advertises a host route used to reach an ASBR.
5	AS External	Created by ASBRs for external routes injected into OSPF.
6	Group Membership	Defined for MOSPF; not supported by Cisco IOS.
7	NSSA External	Created by ASBRs inside an NSSA area, instead of a type 5 LSA.
8	External Attributes	Not implemented in Cisco routers.
9–11	Opaque	Used as generic LSAs to allow for easy future extension of OSPF; for example, type 10 has been adapted for MPLS traffic engineering.

Before diving into the coverage of LSA types, two more definitions are needed:

- **Transit network**—A network over which two or more OSPF routers have become neighbors, so traffic can transit from one to the other.
- **Stub network**—A subnet on which a router has not formed any neighbor relationships.

Now on to the LSA types!

LSA Types 1 and 2

Each router creates and floods a type 1 LSA for itself. These LSAs describe the router, its interfaces (in that area), and a list of neighboring routers (in that area) on each interface. The LSA itself is identified by a *link-state ID (LSID)* equal to that router's RID.

Type 2 LSAs represent a transit subnet for which a DR has been elected. The LSID is the RID of the DR on that subnet. Note that type 2 LSAs are not created for subnets on which no DR has been elected.

Armed with an LSDB with all the type 1 and 2 LSAs inside an area, a router's SPF algorithm should be able to create a topological graph of the network, calculate the possible routes, and finally choose the best routes. For example, Figure 10-6 shows a sample internetwork that is used in several upcoming examples. Figure 10-7 shows a graphical view of the type 1 and type 2 LSAs created in area 3.

Figure 10-6 *Network Used in LSA Examples*

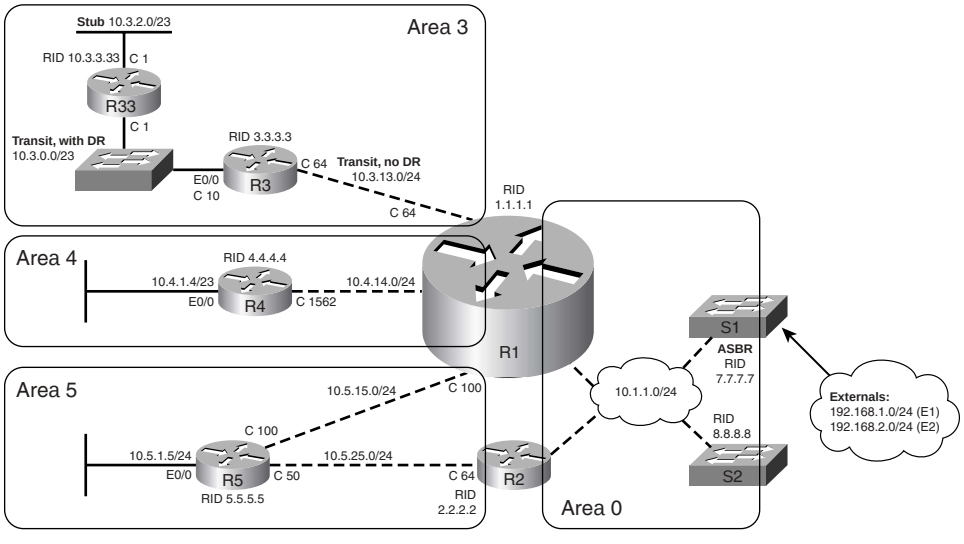
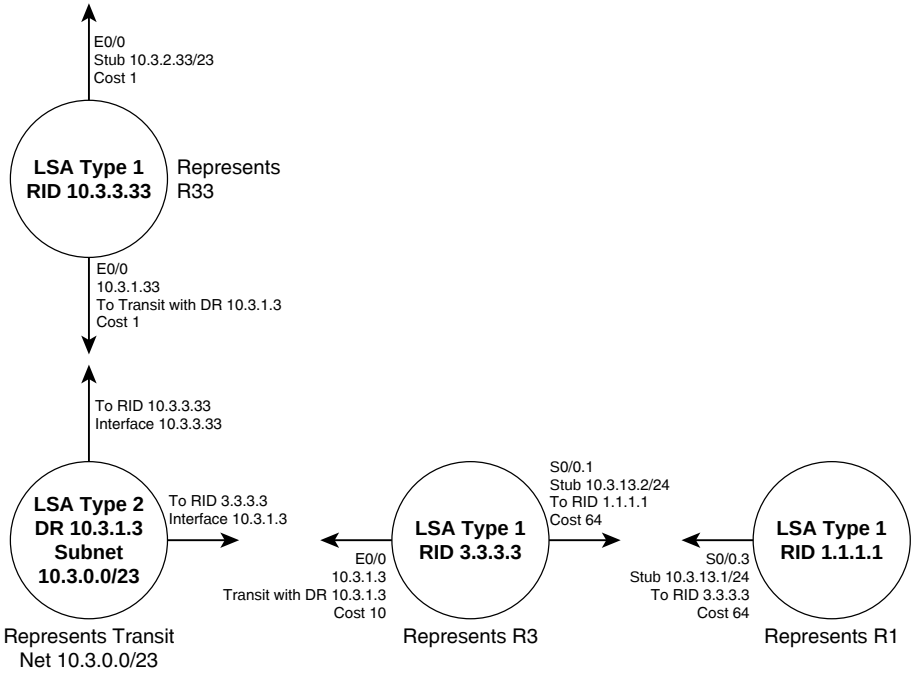


Figure 10-7 *Graph of Type 1 and 2 LSAs for Area 3*



For subnets without a DR, the type 1 LSAs hold enough information for the SPF algorithm to create the math model of the topology. For example, R1 and R3 use point-to-point subinterfaces, and the OSPF point-to-point network type. SPF can match up the information shown in the type 1 LSAs for R1 and R3 in Figure 10-7 to know that the two routers are connected.

For transit networks with DRs, OSPF uses a type 2 LSA to model the subnet as a node in the SPF mathematical model. Because the SPF process treats the type 2 LSA as a node in the graph, this LSA is sometimes called a *pseudonode*. The type 2 LSA includes references to the RIDs of all routers that are currently neighbors of the DR on that subnet. That information, combined with the type 1 LSAs for each router connected to the subnet represented by the type 2 LSA, allows SPF to construct an accurate picture of the network.

Example 10-4 shows the LSAs in area 3 (Figures 10-6 and 10-7) via **show** commands.

Example 10-4 *LSA Types 1 and 2 in Area 3*

```
! R3's LSDB is shown, with type 1 LSAs listed as "Router Link States" and
! type 2 LSAs as "Net Link States." The command output shows a section for each LSA
! type, in sequential order.
R3# show ip ospf database
            OSPF Router with ID (3.3.3.3) (Process ID 1)
            Router Link States (Area 3)
```

continues

Example 10-4 *LSA Types 1 and 2 in Area 3 (Continued)*

```
Link ID         ADV Router      Age         Seq#       Checksum Link count
1.1.1.1         1.1.1.1         1203        0x80000025 0x0072C3 2
3.3.3.3         3.3.3.3         779         0x80000027 0x003FB0 3
10.3.3.33       10.3.3.33       899         0x80000020 0x002929 2

                Net Link States (Area 3)
Link ID         ADV Router      Age         Seq#       Checksum
10.3.1.3        3.3.3.3         1290        0x8000001F 0x00249E
! Lines omitted for brevity
! Next, the specific LSA's link ID is included in the show command, listing detail
! for the one LSA type 2 inside area 3. Note that the "Link ID" is the DR's
! interface address on the subnet. The network keyword refers to the network LSAs (type 2 LSAs).
R3# show ip ospf database network 10.3.1.3
            OSPF Router with ID (3.3.3.3) (Process ID 1)
                Net Link States (Area 3)

Routing Bit Set on this LSA
LS age: 1304
Options: (No TOS-capability, DC)
LS Type: Network Links
Link State ID: 10.3.1.3 (address of Designated Router)
Advertising Router: 3.3.3.3
LS Seq Number: 8000001F
Checksum: 0x249E
Length: 32
Network Mask: /23
      Attached Router: 3.3.3.3
      Attached Router: 10.3.3.33
! Next, the type 1 LSA for R3 is listed. The link ID is the RID of R3. Note that
! the LSA includes reference to each stub and transit link connected to R3. The router
! keyword refers to the router LSAs (type 1 LSAs).
R3# show ip ospf database router 3.3.3.3
            OSPF Router with ID (3.3.3.3) (Process ID 1)
                 Router Link States (Area 3)

  LS age: 804
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 3.3.3.3
  Advertising Router: 3.3.3.3
  LS Seq Number: 80000027
  Checksum: 0x3FB0
  Length: 60
  Number of Links: 3

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 1.1.1.1
     (Link Data) Router Interface address: 10.3.13.3
     Number of TOS metrics: 0
     TOS 0 Metrics: 64
```

Example 10-4 *LSA Types 1 and 2 in Area 3 (Continued)*

```
    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 10.3.13.0
     (Link Data) Network Mask: 255.255.255.0
     Number of TOS metrics: 0
     TOS 0 Metrics: 64

! Note that R3's LSA refers to a transit network next, based on its DR RID –
! these lines allow OSPF to know that this router (R3) connects to the transit
! network whose type 2 LSA has LSID 10.3.1.3.

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.3.1.3
     (Link Data) Router Interface address: 10.3.1.3
     Number of TOS metrics: 0
      TOS 0 Metrics: 10
! Below, the routes from R3 and R1 to 10.3.2.0/23 are shown. Note the cost values
! for each reflect the cumulative costs of the outgoing interfaces used to reach
! the subnet—for instance, R3's cost is the sum of its outgoing interface cost
! (10) plus R33's outgoing interface cost (1). R1's cost is based on three outgoing
! links: R1 (cost 64), R3 (cost 10), and R33 (cost 1), for a total of 75. Also
! note that the time listed in the route is the time since this LSA first arrived
! at the router, even if the LSA has been refreshed due to the LSRefresh interval.
R3# show ip route ospf 1 | include 10.3.2.0
O       10.3.2.0/23 [110/11] via 10.3.1.33, 17:08:33, Ethernet0/0
R1# show ip route ospf | include 10.3.2.0
O       10.3.2.0/23 [110/75] via 10.3.13.3, 17:10:15, Serial0/0.3
```

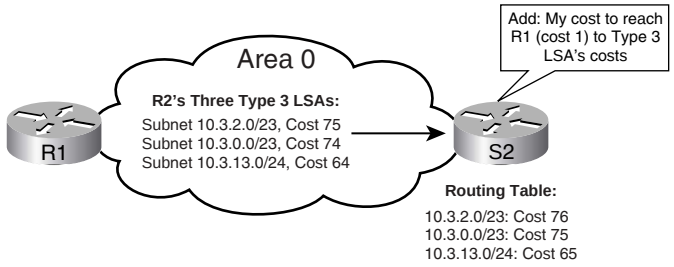
The **show ip ospf database** command lists the LSAs in that router's LSDB, with LSA type 1 LSAs (router LSAs) first, then type 2 (network link states), continuing sequentially through the LSA types. Also note that the LSDB for area 3 should be identical on R33, R3, and R1. However, on R1, the **show ip ospf database** command lists all of R1's LSDB entries, including LSAs from other areas, so using an internal router to look at the LSDB may be the best place to begin troubleshooting a problem. Also note the costs for the routes on R3 and R1 at the end of the example—the SPF algorithm simply added the outgoing costs along the routes, from each router's perspective.

NOTE To signify a network that is down, the appropriate type 1 or 2 LSA is changed to show a metric of 16,777,215 ($2^{24} - 1$), which is considered to be an infinite metric to OSPF.

LSA Type 3 and Inter-Area Costs

ABRs do not forward type 1 and 2 LSAs from one area to another. Instead, ABRs advertise type 3 LSAs into one area in order to represent subnets described in both the type 1 and 2 LSAs in another area. Each type 3 summary LSA describes a simple vector—the subnet, mask, and the ABR's cost to reach that subnet, as shown in Figure 10-8.

Figure 10-8 *Representation of Area 3 Subnets as Type 3 LSAs in Area 0*



Example 10-5 focuses on the three subnets inside area 3, looking at the type 3 summary LSAs created for those subnets by ABR R1. Note that the example shows commands on S2; S2 has identical area 0 LSDB entries as compared with R1.

Example 10-5 *LSA Type 3 Created by R1 for Area 3's Subnets*

```
! S2, internal to area 0, does not have the type 1 and 2 LSAs seen by R3 back in
! Example 10-4. However, type 3 LSAs (listed as "Summary Net Links") show all
! three subnets inside area 3. R1 is listed as the advertising router because it
! created the type 3 LSAs.
S2# show ip ospf database
! Lines omitted for brevity
                Summary Net Link States (Area 0)
Link ID         ADV Router      Age         Seq#       Checksum
10.3.0.0        1.1.1.1         257         0x80000001 0x00A63C
10.3.2.0        1.1.1.1         257         0x80000001 0x009A45
10.3.13.0       1.1.1.1         261         0x80000021 0x007747
! Lines omitted for brevity
! Below, note that the summary keyword is used to view type 3 LSAs. The metric
! reflects R1's cost to reach the subnet inside area 3.
S2# show ip ospf database summary 10.3.0.0
            OSPF Router with ID (8.8.8.8) (Process ID 1)
               Summary Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 341
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 10.3.0.0 (summary Network Number)
  Advertising Router: 1.1.1.1
  LS Seq Number: 80000001
  Checksum: 0xA63C
  Length: 28
  Network Mask: /23
         TOS: 0   Metric: 74
! Next, S2's routes to all three subnets are listed. S2 calculates its cost
! based on its cost to reach R1, plus the cost listed in the type 3 LSA. For
```

Example 10-5 *LSA Type 3 Created by R1 for Area 3's Subnets (Continued)*

```
! example, the cost (above) in the type 3 LSA for 10.3.0.0/23 is 74; S2 adds
! that to S2's cost to reach ABR R1 (cost 1), for a metric of 75.
S2# show ip route ospf ¦ include 10.3
O IA    10.3.13.0/24 [110/65] via 10.1.1.1, 00:16:04, Vlan1
O IA    10.3.0.0/23 [110/75] via 10.1.1.1, 00:05:08, Vlan1
O IA    10.3.2.0/23 [110/76] via 10.1.1.1, 00:05:12, Vlan1
! Next, S2's cost to reach RID 1.1.1.1 is listed as cost 1.
S2# show ip ospf border-routers
OSPF Process 1 internal Routing Table
Codes: i — Intra-area route, I — Inter-area route

i 1.1.1.1 [1] via 10.1.1.1, Vlan1, ABR, Area 0, SPF 18
i 2.2.2.2 [1] via 10.1.1.2, Vlan1, ABR, Area 0, SPF 18
i 7.7.7.7 [1] via 10.1.1.3, Vlan1, ASBR, Area 0, SPF 18
```

```
! Below, the show ip ospf statistics command lists the number of SPF calculations.
R1# show ip ospf stat
OSPF process ID 1
----------------------------------------
  Area 0: SPF algorithm executed 6 times
  Area 3: SPF algorithm executed 15 times
  Area 4: SPF algorithm executed 6 times
  Area 5: SPF algorithm executed 5 times
! Lines omitted for brevity
```

Example 10-5 shows how S2 calculated its cost to the area 3 subnets. Routers calculate the cost for a route to a subnet defined in a type 3 LSA by adding the following items:

KEY POINT

1. The calculated cost to reach the ABR that created and advertised the type 3 LSA.
2. The cost as listed in the type 3 LSA.

You can see the cost of the type 3 LSA with the **show ip ospf database summary** *link-id* command, and the cost to reach the advertising ABR with the **show ip ospf border-routers** command, as shown in Example 10-5.

The beauty of this two-step cost calculation process is that it allows a significant reduction in the number of SPF calculations. When a type 1 or 2 LSA changes in some way that affects the underlying routes—for instance, a link failure—each router in the area runs SPF, but routers inside other areas do not. For instance, if R3's E0/0 is shut down, all three routers in area 3 run SPF inside that area, and the counter for area 3 in the **show ip ospf statistics** command increments. However, routers not inside area 0 do not run SPF, even though they update their routing tables—a process called a *partial run*, *partial SPF*, or *partial calculation*.

For example, imagine that R3's LAN interface fails. R33 then updates its type 2 LSA, listing a metric of 16,777,215. R1 in turn updates its type 3 LSA for 10.3.0.0/23, flooding that throughout area 0. The next step shows the computational savings: S2, using the two-step calculation, simply adds its cost to R1 (still 1) to 16,777,215, finds the number out of range, and removes the route from the IP routing table. S2 did not have to actually run the SPF algorithm to discover a new SPF tree.

Of particular importance is that partial calculations happen without any route summarization. With OSPF, route summarization does help reduce the overall number of routes that require SPF calculations, but route summarization is not required for partial calculations to occur.

LSA Types 4 and 5, and External Route Types 1 and 2

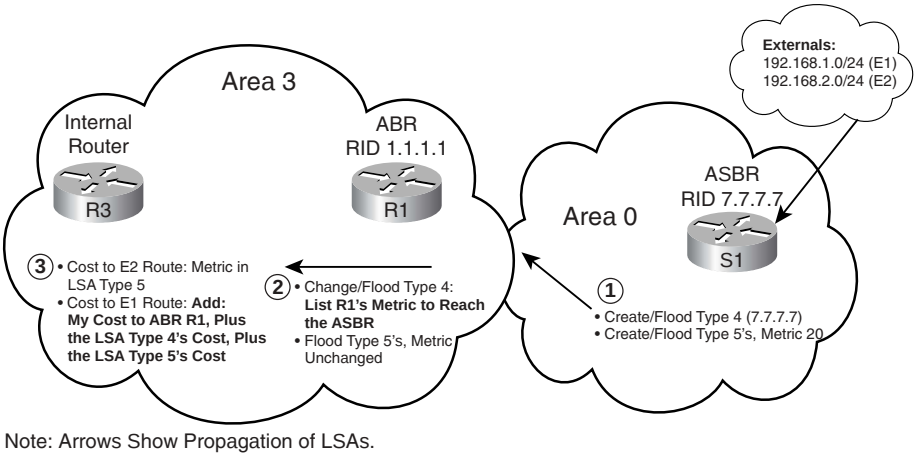
KEY POINT OSPF allows for two types of external routes, aptly named types 1 and 2. The type determines whether only the external metric is considered by SPF when picking the best routes (external type 2, or E2), or whether both the external and internal metrics are added together to compute the metric (external type 1, or E1).

When an ASBR injects an E2 route, it creates a type 5 LSA for the subnet. The LSA lists the metric. The ASBR then floods the type 5 LSA throughout all areas. The other routers simply use the metric listed in the LSA; no need exists to add any cost on any links internal to the OSPF domain.

To support E1 routes, the ASBR creates both an LSA type 4 for itself and a type 5 LSA. Both types of LSAs are flooded throughout the OSPF domain, including being forwarded by ABRs into other areas. Other routers calculate their costs to reach E1 routes in a manner similar to how metrics for LSA type 3 routes are calculated—by calculating the cost to reach the ASBR, and then adding the cost listed in the type 5 LSA. Figure 10-9 outlines the mechanics of how the LSAs are propagated, and how the metrics are calculated.

Figure 10-9 *LSA Types 4 and 5 Propagation and the Effect on Type 1 External Routes*

KEY POINT



Note: Arrows Show Propagation of LSAs.

E1 routes by definition include the cost as assigned when the ASBR injected the route into OSPF, plus any cost inside the OSPF domain. To calculate the cost for the E1 route, a router inside a different area must use two steps to calculate the internal cost, and a third step to add the external cost. For example, when R3, internal to area 3, calculates the cost to reach 192.168.2.0/24 (an E1 route), R3 adds the following:

- R3's calculated area 3 cost to reach ABR R1 (RID 1.1.1.1).
- R1's cost to reach the ASBR that advertised the route (S2, RID 7.7.7.7). R1 announces this cost in the forwarded LSA type 4 that describes a host route to reach ASBR 7.7.7.7.
- The external metric for the route, as listed in the type 5 LSA created by the ASBR.

Example 10-6 shows the components of the metrics and LSAs for two external routes: 192.168.1.0/24 E1 with metric 20, and 192.168.2.0/24 E2, also with metric 20.

Example 10-6 *Calculating the Metric for External Types 1 and 2*

```
! R3 has learned the two LSA type 5s.
R3# show ip ospf database | begin Type-5
               Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
192.168.1.0     7.7.7.7         1916        0x8000002B 0x0080EF 0
192.168.2.0     7.7.7.7         1916        0x80000028 0x00FEF2 0
! Next, the detail for E2 192.168.2.0 is listed, with "metric type" referring
! to the external route type E2. (192.168.1.0, not shown, is type 1.)
R3# show ip ospf database external 192.168.2.0
            OSPF Router with ID (3.3.3.3) (Process ID 1)
        Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 1969
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 192.168.2.0 (External Network Number)
  Advertising Router: 7.7.7.7
  LS Seq Number: 80000028
  Checksum: 0xFEF2
  Length: 36
  Network Mask: /24
    Metric Type: 2 (Larger than any link state path)
    TOS: 0
    Metric: 20
    Forward Address: 0.0.0.0
    External Route Tag: 0
! Next, R1's advertised cost of 1 between itself and the ASBR is listed. Note
! that S1's RID (7.7.7.7) is listed, with the ABR that forwarded the LSA into
```

continues

Example 10-6 *Calculating the Metric for External Types 1 and 2 (Continued)*

```
! area 3, R1 (RID 1.1.1.1) also listed.
R3# show ip ospf database asbr-summary
            OSPF Router with ID (3.3.3.3) (Process ID 1)
        Summary ASB Link States (Area 3)

  Routing Bit Set on this LSA
  LS age: 923
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(AS Boundary Router)
  Link State ID: 7.7.7.7 (AS Boundary Router address)
  Advertising Router: 1.1.1.1
  LS Seq Number: 8000000A
  Checksum: 0x12FF
  Length: 28
  Network Mask: /0
    TOS: 0     Metric: 1
! Below, R3's calculated cost to R1 (64) and then to S2 (7.7.7.7) are listed. Note
! that the total of 65 is the cost 64 to reach the ABR, plus the cost 1 for the
! ABR to reach the ASBR.
R3# show ip ospf border-routers
OSPF Process 1 internal Routing Table
Codes: i — Intra-area route, I — Inter-area route

i 1.1.1.1 [64] via 10.3.13.1, Serial0/0.1, ABR, Area 3, SPF 30
I 7.7.7.7 [65] via 10.3.13.1, Serial0/0.1, ASBR, Area 3, SPF 30
! Below, each route is noted as E1 or E2, with the E1 route's metric including
! the external cost (20), plus cost to reach the ASBR (65).
R3# show ip route | include 192.168
O E1 192.168.1.0/24 [110/85] via 10.3.13.1, 00:50:34, Serial0/0.1
O E2 192.168.2.0/24 [110/20] via 10.3.13.1, 00:50:34, Serial0/0.1
```

OSPF Design in Light of LSA Types

OSPF's main design trade-offs consist of choosing links for particular areas, with the goal of speeding convergence, reducing memory and computing resources, and keeping routing tables small through route summarization. For instance, by using a larger number of areas, and the implied conversion of dense types 1 and 2 LSAs into sparser type 3 LSAs, the OSPF LSDBs can be made smaller. Also, link flaps in one area require SPF calculations only in that area, due to the partial calculation feature. Additionally, ABRs and ASBRs can be configured to summarize routes, reducing the number of type 3 LSAs introduced into other areas as well. (Route summarization is covered in Chapter 11, "IGP Route Summarization, Route Redistribution, and Default Routes.")

The OSPF design goals to reduce convergence time, reduce overhead processing, and improve network stability can be reached using the core OSPF protocols and features covered so far. Another key OSPF design tool, stubby areas, will be covered next.

NOTE Before moving on, a comment is in order about the relative use of the word "summary" in OSPF. The typical uses within OSPF include the following:

- Type 3 LSAs are called *summary* LSAs in the OSPF RFCs.
- Type 5 and 7 external LSAs are sometimes called summary LSAs, because the LSAs cannot represent detailed topology information.
- The term *LSA summary* refers to the LSA headers that summarize LSAs and are sent inside DD packets.
- The term *summary* can also be used to refer to summary routes created with the **area range** and **summary-address** commands.

Stubby Areas

OSPF can further reduce overhead by treating each area with one of several variations of rules, based on a concept called a *stubby area*. Stubby areas take advantage of the fact that to reach subnets in other areas, routers in an area must forward the packets to some ABR. Without stubby areas, ABRs must advertise all the subnets into the area, so that the routers know about the subnets. With stubby areas, ABRs quit advertising type 5 (external) LSAs into the stubby area, but instead ABRs create and advertise default routes into the stubby area. As a result, internal routers use default routing to forward packets to the ABR anyway. However, the internal routers now have sparser LSDBs inside the area.

The classic case for a stubby area is an area with one ABR, but stubby areas can work well for areas with multiple ABRs as well. For example, the only way out of area 3 in Figure 10-6 is through the only ABR, R1. So, R1 could advertise a default route into area 3 instead of advertising any external type 5 LSAs.

Also in Figure 10-6, area 5 has two ABRs. If area 5 were a stubby area, both ABRs would inject default routes into the area. This configuration would work, but it may result in suboptimal routing.

OSPF defines several different types of stubby areas. By definition, all stubby areas stop type 5 (external) LSAs from being injected into them by the ABRs. However, depending on the variation, a stubby area may also prevent type 3 LSAs from being injected. The other variation includes whether a router inside the stubby area can redistribute routes into OSPF, thereby injecting an external route. Table 10-5 lists the variations on stubby areas, and their names.

Note in Table 10-5 that all four stub area types stop type 5 LSAs from entering the area. When the name includes "totally," type 3 LSAs are also not passed into the area, significantly reducing the size of the LSDB. If the name includes "NSSA," it means that external routes can be redistributed into OSPF by routers inside the stubby area; note that the LSAs for these external routes would be type 7.

Table 10-5 *OSPF Stubby Area Types*

KEY POINT

Area Type	Stops Injection of Type 5 LSAs?	Stops Injection of Type 3 LSAs?	Allows Creation of Type 7 LSAs Inside the Area?
Stub	Yes	No	No
Totally stubby	Yes	Yes	No
Not-so-stubby area (NSSA)	Yes	No	Yes
Totally NSSA	Yes	Yes	Yes

Configuring a stub area is pretty simple—all routers in the area need the same stub settings, as configured in the **area stub** command. Table 10-6 lists the options.

Table 10-6 *Stub Area Configuration Options*

KEY POINT

Stub Type	Router OSPF Subcommand
NSSA	**area** *area-id* **nssa**
Totally NSSA	**area** *area-id* **nssa no-summary**
Stub	**area** *area-id* **stub**
Totally stubby	**area** *area-id* **stub no-summary**

Example 10-7, based on Figure 10-6, shows the results of the following configuration:

- Area 3 is configured as a totally NSSA area.
- R3 will inject an external route to 192.168.21.0/24 as a type 7 LSA.
- Area 4 is configured as a totally stubby area.
- Area 5 is configured as simply stubby.

Example 10-7 *Stub Area Example*

```
! R3, in a totally NSSA area, knows intra-area routes (denoted with an "IA"
! near the front of the output line from show ip route), but the only
! inter-area route is the default route created and sent by R1, the ABR.
R3# show ip route ospf
     10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
O       10.3.2.0/23 [110/11] via 10.3.1.33, 00:00:00, Ethernet0/0
O*IA 0.0.0.0/0 [110/65] via 10.3.13.1, 00:00:00, Serial0/0.1
! Still on R3, the LSA type 3 summary, created by ABR R1, is shown first.
! Next, the External NSSA LSA type 7 LSA created by R3 is listed.
```

Example 10-7 *Stub Area Example (Continued)*

```
R3# show ip ospf database | begin Summary
        Summary Net Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum
0.0.0.0         1.1.1.1         704         0x80000004 0x00151A

          Type-7 AS External Link States (Area 3)

Link ID         ADV Router      Age         Seq#       Checksum Tag
192.168.21.0    3.3.3.3         17          0x80000003 0x00C12B 0
```

```
! R1, because it is attached to area 3, also has the R3-generated NSSA external
! LSA. Note the advertising router is R3, and it is an E2 external route.
R1# show ip ospf database nssa-external
            OSPF Router with ID (1.1.1.1) (Process ID 1)
        Type-7 AS External Link States (Area 3)

          Routing Bit Set on this LSA
          LS age: 188
          Options: (No TOS-capability, Type 7/5 translation, DC)
          LS Type: AS External Link
          Link State ID: 192.168.21.0 (External Network Number )
          Advertising Router: 3.3.3.3
          LS Seq Number: 80000003
          Checksum: 0xC12B
          Length: 36
          Network Mask: /24
            Metric Type: 2 (Larger than any link state path)
            TOS: 0
            Metric: 20
            Forward Address: 10.3.13.3
            External Route Tag: 0
```

```
! Below, the same command on R2, not in area 3, shows no type 7 LSAs. ABRs
! convert type 7 LSAs to type 5 LSAs before forwarding them into another area.
R2# show ip ospf database nssa-external

            OSPF Router with ID (2.2.2.2) (Process ID 2)
! Next, R2 does have a type 5 LSA for the subnet; R1 converts the type 7 to a type
! 5 before flooding it into other areas.
R2# show ip ospf database | begin Type-5
                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
192.168.1.0     7.7.7.7         521         0x80000050 0x003615 0
192.168.2.0     7.7.7.7         521         0x8000004D 0x00B418 0
192.168.21.0    1.1.1.1         1778        0x80000019 0x006682 0
```

continues

Example 10-7 *Stub Area Example (Continued)*

```
! Below, R4 is in a totally stubby area, with only one inter-area route.
R4# show ip route ospf
O*IA 0.0.0.0/0 [110/1563] via 10.4.14.1, 00:11:59, Serial0/0.1

! R5, in a stubby area, has several inter-area routes, but none of the
! external routes (e.g. 192.168.1.0). R5's default points to R2.
R5# show ip route ospf
10.0.0.0/8 is variably subnetted, 7 subnets, 3 masks
O IA    10.3.13.0/24 [110/115] via 10.5.25.2, 13:45:49, Serial0.2
O IA    10.3.0.0/23 [110/125] via 10.5.25.2, 13:37:55, Serial0.2
O IA    10.1.1.0/24 [110/51] via 10.5.25.2, 13:45:49, Serial0.2
O IA    10.4.0.0/16 [110/1613] via 10.5.25.2, 13:45:49, Serial0.2
O*IA 0.0.0.0/0 [110/51] via 10.5.25.2, 13:45:49, Serial0.2

! Below, R5's costs on its two interfaces to R1 and R2 are highlighted. Note that
! the default route's metric (51) comes from the 50 below, plus an advertised
! cost of 1 in the summary (type 3) for default 0.0.0.0/0 generated by R2. R5
! simply chose to use the default route with the lower metric.
R5# sh ip ospf int brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Se0.1        1     5               10.5.15.5/24       64    P2P   1/1
Se0.2        1     5               10.5.25.5/24       50    P2P   1/1
Et0          1     5               10.5.1.5/24        10    DR    0/0

! Next, R2 changes the cost of its advertised summary from 1 to 15.
R2# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R2(config)# router ospf 2
R2(config-router)# area 5 default-cost 15

! Below, R5's metrics to both R1's and R2's default routes tie,
! so both are now in the routing table.
R5# show ip route ospf
! lines omitted for brevity
O*IA 0.0.0.0/0 [110/65] via 10.5.25.2, 00:00:44, Serial0.2
               [110/65] via 10.5.15.1, 00:00:44, Serial0.1
```

The legend in the top of the output of a **show ip route** command lists several identifiers that pertain to OSPF. For example, the acronym "IA" refers to interarea OSPF routes, E1 refers to external type 1 routes, and E2 refers to external type 2 routes.

OSPF Configuration

This section covers the core OSPF configuration commands, along with the OSPF configuration topics not already covered previously in the chapter. (If you happened to skip the earlier parts of this chapter, planning to review OSPF configuration, make sure to go back and look at the earlier

examples in the chapter. These examples cover OSPF stubby area configuration, OSPF network types, plus OSPF **neighbor** and **priority** commands.)

Example 10-8 shows configuration for the routers in Figure 10-6, with the following design goals in mind:

- Proving that OSPF PIDs do not have to match on separate routers
- Using the **network** command to match interfaces, thereby triggering neighbor discovery inside network 10.0.0.0
- Configuring S1's RID as 7.7.7.7
- Setting priorities on the backbone LAN to favor S1 and S2 to become the DR/BDR
- Configuring a minimal dead interval of 1 second, with hello multiplier of 4, yielding a 250-ms hello interval on the backbone LAN

Example 10-8 *OSPF Configuration Basics and OSPF Costs*

```
! R1 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! R1 has been configured for a (minimal) 1-second dead interval, and 1/4-second
! (250 ms) hello interval based on 4 Hellos per 1-second dead interval.
interface FastEthernet0/0
 ip address 10.1.1.1 255.255.255.0
 ip ospf dead-interval minimal hello-multiplier 4
! R1 uses the same stub area configuration as in Example 10-7, with network
! commands matching based on the first two octets. Note that the network commands
! place each interface into the correct area.
router ospf 1
 area 3 nssa no-summary
 area 4 stub no-summary
 area 5 stub
 network 10.1.0.0 0.0.255.255 area 0
 network 10.3.0.0 0.0.255.255 area 3
 network 10.4.0.0 0.0.255.255 area 4
 network 10.5.0.0 0.0.255.255 area 5
```

```
! R2 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! The R2 configuration also uses the Fast Hello feature, otherwise it
! would not match hello and dead intervals with R1.
interface FastEthernet0/0
 ip address 10.1.1.2 255.255.255.0
 ip ospf dead-interval minimal hello-multiplier 4
! Below, R2 uses a different PID than R1, but the PID is only used locally.
! R1 and R2 will become neighbors. Also, all routers in a stubby area must be
! configured to be that type of stubby area; R2 does that for area 5 below.
```

continues

Example 10-8 *OSPF Configuration Basics and OSPF Costs (Continued)*

```
router ospf 2
 area 5 stub
 network 10.1.0.0 0.0.255.255 area 0
 network 10.5.25.2 0.0.0.0 area 5
```

```
! R3 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!Note that R3's area 2 nssa no-summary command must match the settings
! on R1. Likewise, below, R4's stub settings must match R1's settings for area 4.
.
router ospf 1
 area 3 nssa no-summary
 network 10.0.0.0 0.255.255.255 area 3
```

```
! R4 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
router ospf 1
 area 4 stub no-summary
 network 10.0.0.0 0.255.255.255 area 4
```

```
! S1 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! S1 matches hello and dead intervals on the LAN. Also, it sets its OSPF
! priority to 255, the maximum value, hoping to become the DR.
interface Vlan1
 ip address 10.1.1.3 255.255.255.0
 ip ospf dead-interval minimal hello-multiplier 4
 ip ospf priority 255
! Below, S1 sets its RID manually, removing any reliance on an interface address.
router ospf 1
 router-id 7.7.7.7
 network 10.1.0.0 0.0.255.255 area 0
```

```
! S2 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Below, S2 also matches timers, and sets its priority to 1 less than S1, hoping
! to be the BDR.
interface Vlan1
 ip address 10.1.1.4 255.255.255.0
 ip ospf dead-interval minimal hello-multiplier 4
 ip ospf priority 254
!
router ospf 1
network 10.0.0.0 0.255.255.255 area 0
```

OSPF Costs and Clearing the OSPF Process

Example 10-9 highlights a few details about clearing (restarting) the OSPF process, and looks at changes to OSPF costs. This example shows the following sequence:

1. R3's OSPF process is cleared, causing all neighbors to fail and restart.
2. R3's **log-adjacency-changes detail** configuration command (under **router ospf**) causes more detailed neighbor state change messages to appear.

3. R5 has tuned its cost settings with the **ip ospf cost 50** interface subcommand under S0.2 in order to prefer R2 over R1 for reaching the core.

4. R2 is configured to use a new reference bandwidth, changing its cost calculation per interface.

Example 10-9 *Changing RIDs, Clearing OSPF, and Cost Settings*

```
R3# clear ip ospf process
Reset ALL OSPF processes? [no]: y
! Above, all OSPF processes are cleared on R3. R3 has the log-adjacency-changes
! detail command configured, so that a message is generated at each state
! change, as shown below for neighbor R33 (RID 192.168.1.1). (Messages for
! other routers are omitted.)
00:02:46: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.1.1 on Ethernet0/0 from FULL to DOWN,
  Neighbor Down: Interface down or detached
00:02:53: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.1.1 on Ethernet0/0 from DOWN to INIT,
  Received Hello
00:02:53: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.1.1 on Ethernet0/0 from INIT to 2WAY,
  2-Way Received
00:02:53: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.1.1 on Ethernet0/0 from 2WAY to EXSTART,
  AdjOK?
00:02:53: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.1.1 on Ethernet0/0 from EXSTART
  to EXCHANGE, Negotiation Done
00:02:53: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.1.1 on Ethernet0/0 from EXCHANGE
  to LOADING, Exchange Done
00:02:53: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.1.1 on Ethernet0/0 from LOADING to FULL,
  Loading Done
```

```
! Next R5 has costs of 50 and 64, respectively, on interfaces s0.2 and s0.1.
R5# show ip ospf int brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Se0.2        1     5               10.5.25.5/24       50    P2P   1/1
Se0.1        1     5               10.5.15.5/24       64    P2P   1/1
Et0          1     5               10.5.1.5/24        10    DR    0/0
! Below, S0.1's cost was based on bandwidth of 64, using formula 10^8 / bandwidth,
! with bandwidth in bits/second.
R5# sh int s 0.1
Serial0.1 is up, line protocol is up
  Hardware is HD64570
  Internet address is 10.5.15.5/24
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation FRAME-RELAY
  Last clearing of "show interface" counters never
```

```
! Next, R2's interface costs are shown, including the minimum cost 1 on fa0/0.
R2# sho ip ospf int brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Fa0/0        2     0               10.1.1.2/24        1     BDR   3/3
Se0/0.5      2     5               10.5.25.2/24       64    P2P   1/1
! Below, R2 changes its reference bandwidth from the default of 100 Mbps to
! 10,000 Mbps. That in turn changes R2's calculated cost values to be 100 times
```

continues

Example 10-9 *Changing RIDs, Clearing OSPF, and Cost Settings (Continued)*

```
! larger than before. Note that IOS allows this setting to differ on the routers,
! but recommends against it.
R2# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R2(config)# router ospf 2
R2(config-router)# auto-cost reference-bandwidth 10000
% OSPF: Reference bandwidth is changed.
        Please ensure reference bandwidth is consistent across all routers.
R2# show ip ospf int brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Fa0/0        2     0               10.1.1.2/24        100   BDR   3/3
Se0/0.5      2     5               10.5.25.2/24       6476  P2P   1/1
```

While Examples 10-8 and 10-9 show some details, the following list summarizes how IOS chooses OSPF interface costs:

KEY POINT

1. Set the cost per neighbor using the **neighbor** *neighbor* **cost** *value* command. (This is valid only on OSPF network types that allow **neighbor** commands.)
2. Set the cost per interface using the **ip ospf cost** *value* interface subcommand.
3. Allow cost to default based on interface bandwidth and the OSPF Reference Bandwidth (Ref-BW) (default 10^8). The formula is Ref-BW / bandwidth (bps).
4. Default based on bandwidth, but change Ref-BW using the command **ospf auto-cost reference-bandwidth** value command within the OSPF process.

The only slightly tricky part of the cost calculation math is to keep the units straight, because the IOS interface bandwidth is kept in kbps, and the **auto-cost reference-bandwidth** command's units are Mbps. For instance, on R5 in Example 10-9, the cost is calculated as 100 Mbps divided by 1544 kbps, where 1544 kbps is equal to 1.544 Mbps. The result is rounded down to the nearest integer, 64 in this case. On R2's fa0/0, the bandwidth is 100,000 kbps, or 100 Mbps, making the calculation yield a cost of 1. After changing the reference bandwidth to 10,000, which means 10,000 Mbps, R2's calculated costs were 100 times larger.

NOTE When choosing the best routes to reach a subnet, OSPF also considers whether a route is an intra-area route, inter-area route, E1 route, or E2 route. OSPF prefers intra-area over all the rest, then interarea, then E1, and finally E2 routes. Under normal circumstances, routes to a single subnet should all be the same type; however, it is possible to have multiple route paths to reach a single subnet in the OSPF SPF tree, but with some of these routes being a different type. Example 11-7 in Chapter 11 demonstrates this.

Alternatives to the OSPF Network Command

As of Cisco IOS Software Release 12.3(11)T, OSPF configuration can completely omit the **network** command, instead relying on the **ip ospf** *process-id* ***area*** *area-id* interface subcommand. This new command enables OSPF on the interface and selects the area. For instance, on R3 in Example 10-8, the **network 10.3.0.0 0.0.255.255 area 3** command could have been deleted and replaced with the **ip ospf 1 area 3** command under S0/0.1 and e0/0.

The **network** and **ip ospf area** commands have some minor differences when secondary IP addresses are used. With the **network** command, OSPF advertises stub networks for any secondary IP subnets that are matched by the command. ("Secondary subnet" is jargon that refers to the subnet in which a secondary IP address resides.) The **ip ospf area** interface subcommand causes any and all secondary subnets on the interface to be advertised as stub networks—unless the optional **secondaries none** parameter is included at the end of the command.

OSPF Filtering

Intra-routing–protocol filtering presents some special challenges with link-state routing protocols like OSPF. Link-state protocols do not advertise routes—they advertise topology information. Also, SPF loop prevention relies on each router in the same area having an identical copy of the LSDB for that area. Filtering could conceivably make the LSDBs differ on different routers, causing routing irregularities.

IOS supports three variations of what could loosely be categorized as OSPF route filtering. These three major types of OSPF filtering are as follows:

- **Filtering routes, not LSAs**—Using the **distribute-list in** command, a router can filter the *routes* its SPF process is attempting to add to its routing table, without affecting the LSDB.
- **ABR type 3 LSA filtering**—A process of preventing an ABR from creating particular type 3 summary LSAs.
- **Using the area range no-advertise option**—Another process to prevent an ABR from creating specific type 3 summary LSAs.

Each of these three topics is discussed in sequence in the next few sections.

Filtering Routes Using the distribute-list Command

For RIP and EIGRP, the **distribute-list** command can be used to filter incoming and outgoing routing updates. The process is straightforward, with the **distribute-list** command referring to ACLs or prefix lists. With OSPF, the **distribute-list** command filters what ends up in the IP routing table, and on only the router on which the **distribute-list** command is configured.

NOTE The **distribute-list** command, when used for route distribution between OSPF and other routing protocols, does control what enters and leaves the LSDB. Chapter 11 covers more on route redistribution.

The following rules govern the use of distribute lists for OSPF, when not used for route redistribution with other routing protocols:

KEY POINT

- Distribute lists can be used only for inbound filtering, because filtering any outbound OSPF information would mean filtering LSAs, not routes.
- The inbound logic does not filter inbound LSAs; it instead filters the routes that SPF chooses to add to that one router's routing table.
- If the distribute list includes the incoming interface parameter, the incoming interface is checked as if it were the *outgoing interface* of the route.

That last bullet could use a little clarification. For example, if R2 learns routes via RIP or EIGRP updates that enter R2's s0/0 interface, those routes typically use R2's s0/0 interface as the outgoing interface of the routes. The OSPF LSAs may have been flooded into a router on several interfaces, so an OSPF router checks the outgoing interface of the route as if it had learned about the routes via updates coming in that interface.

Example 10-10 shows an example of two distribute lists on R5 from Figure 10-6. The example shows two options to achieve the same goal. In this case, R5 will filter the route to 10.4.8.0/24 via R5's S0.2 subinterface (to R2), instead using the route learned from R1. Later, it uses a **route map** to achieve the same result.

Example 10-10 *Filtering Routes with OSPF* **distribute-list** *Commands on R5*

```
! R5 has a route to 10.4.8.0/24 through R2 (10.5.25.2, s0.2)
R5# sh ip route ospf | incl 10.4.8.0
O IA    10.4.8.0/24 [110/1623] via 10.5.25.2, 00:00:28, Serial0.2
! Next, the distribute-list command refers to a prefix list that permits 10.4.8.0
! /24.
ip prefix-list prefix-10-4-8-0 seq 5 deny 10.4.8.0/24
ip prefix-list prefix-10-4-8-0 seq 10 permit 0.0.0.0/0 le 32
!
Router ospf 1
 distribute-list prefix prefix-10-4-8-0 in Serial0.2
! Below, note that R5's route through R2 is gone, and instead R5 uses its route
! through R1 (s0.1). But the LSDB is unchanged!
R5# sh ip route ospf | incl 10.4.8.0
O IA    10.4.8.0/24 [110/1636] via 10.5.15.1, 00:00:03, Serial0.1
! Not shown: the earlier distribute-list command is removed.
! Below, note that the distribute-list command with the route-map option does not
! have an option to refer to an interface, so the route map itself has been
! configured to refer to the advertising router's RID (2.2.2.2).
```

Example 10-10 *Filtering Routes with OSPF* **distribute-list** *Commands on R5 (Continued)*

```
Router ospf 1
distribute-list route-map lose-10-4-8-0 in
! Next, ACL 48 matches the 10.4.8.0/24 prefix, with ACL 51 matching R2's RID.
access-list 48 permit 10.4.8.0
access-list 51 permit 2.2.2.2
! Below, the route map matches the prefix (based on ACL 48) and the advertising
! RID (ACL 51, matching R2's 2.2.2.2 RID). Clause 20 permits all other prefixes.
route-map lose-10-4-8-0 deny 10
match ip address 48
match ip route-source 51
route-map lose-10-4-8-0 permit 20
! Above, note the same results as the previous distribute list.
R5# sh ip route ospf | incl 10.4.8.0
O IA    10.4.8.0/24 [110/1636] via 10.5.15.1, 00:01:18, Serial0.1
```

Example 10-10 shows only two ways to filter the routes. The **distribute-list route-map** option, added in Cisco IOS Software Release 12.2(15)T, allows a much greater variety of matching parameters, and much more detailed logic with route maps. For instance, this example showed matching a prefix as well as the RID that advertised the LSA to R5, namely 2.2.2.2 (R2). Refer to Chapter 11 for a more complete review of route maps and the **match** command.

OSPF ABR LSA Type 3 Filtering

ABRs do not forward type 1 and 2 LSAs from one area into another, but instead create type 3 LSAs for each subnet defined in the type 1 and 2 LSAs. Type 3 LSAs do not contain detailed information about the topology of the originating area; instead, each type 3 LSA represents a subnet, and a cost from the ABR to that subnet. The earlier section "LSA Type 3 and Inter-Area Costs" covers the details and provides an example.

The *OSPF ABR type 3 LSA filtering* feature allows an ABR to filter type 3 LSAs at the point where the LSAs would normally be created. By filtering at the ABR, before the type 3 LSA is injected into another area, the requirement for identical LSDBs inside the area can be met, while still filtering LSAs.

To configure type 3 LSA filtering, you use the **area** *number* **filter-list prefix** *name* **in** | **out** command under **router ospf**. The referenced **prefix list** is used to match the subnets and masks to be filtered. The **area** *number* and the **in** | **out** option of the **area filter-list** command work together, as follows:

- When **in** is configured, IOS filters prefixes going into the configured area.
- When **out** is configured, IOS filters prefixes coming out of the configured area.

Example 10-11 should clarify the basic operation. ABR R1 will use two alternative **area filter-list** commands, both to filter subnet 10.3.2.0/23, the subnet that exists between R3 and R33 in Figure 10-6. Remember that R1 is connected to areas 0, 3, 4, and 5. The first **area filter-list** command shows filtering the LSA as it goes out of area 3; as a result, R2 will not inject the LSA into any of the other areas. The second case shows the same subnet being filtered going into area 0, meaning that the type 3 LSA for that subnet still gets into the area 4 and 5 LSDBs.

Example 10-11 *Type 3 LSA Filtering on R1 with the* **area filter-list** *Command*

```
! The command lists three lines of extracted output. One line is for the
! type 3 LSA in area 0, one is for area 4, and one is for area 5.
R1# show ip ospf data summary | include 10.3.2.0
       Link State ID: 10.3.2.0 (summary Network Number)
       Link State ID: 10.3.2.0 (summary Network Number)
       Link State ID: 10.3.2.0 (summary Network Number)
! Below, the two-line prefix list denies subnet 10.3.2.0/23, and then permits
! all others.
ip prefix-list filter-type3-10-3-2-0 seq 5 deny 10.3.2.0/23
ip prefix-list filter-type3-10-3-2-0 seq 10 permit 0.0.0.0/0 le 32
Next, the area filter-list command filters type 3 LSAs going out of area 3.
R1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# router ospf 1
R1(config-router)# area 3 filter-list prefix filter-type3-10-3-2-0 out
R1(config-router)# ^Z
! Below, R1 no longer has any type 3 LSAs, in areas 0, 4, and 5. For
! comparison, this command was issued a few commands ago, listing 1 line
! of output for each of the other 3 areas besides area 3.
R1# show ip ospf data | include 10.3.2.0
! Below, the previous area filter-list command is replaced by the next command
! below, which filters type 3 LSAs going into area 0, with the same prefix list.
area 0 filter-list prefix filter-type3-10-3-2-0 in
! Next, only 2 type 3 LSAs for 10.3.2.0 are shown—the ones in areas 4 and 5.
R1# show ip ospf data | include 10.3.2.0
  Link State ID: 10.3.2.0 (summary Network Number)
  Link State ID: 10.3.2.0 (summary Network Number)
! Below, the configuration for filtering type 3 LSAs with the area range command,
! which is explained following this example. The existing area filter-list
! commands from earlier in this chapter have been removed at this point.
R1(config-router)# area 3 range 10.3.2.0 255.255.254.0 not-advertise
R1# show ip ospf data summary ¦ include 10.3.2.0
R1#
```

Filtering Type 3 LSAs with the area range Command

The third method to filter OSPF routes is to filter type 3 LSAs at an ABR using the **area range** command. The **area range** command performs route summarization at ABRs, telling a router to cease advertising smaller subnets in a particular address range, instead creating a single type 3 LSA whose address and prefix encompass the smaller subnets.

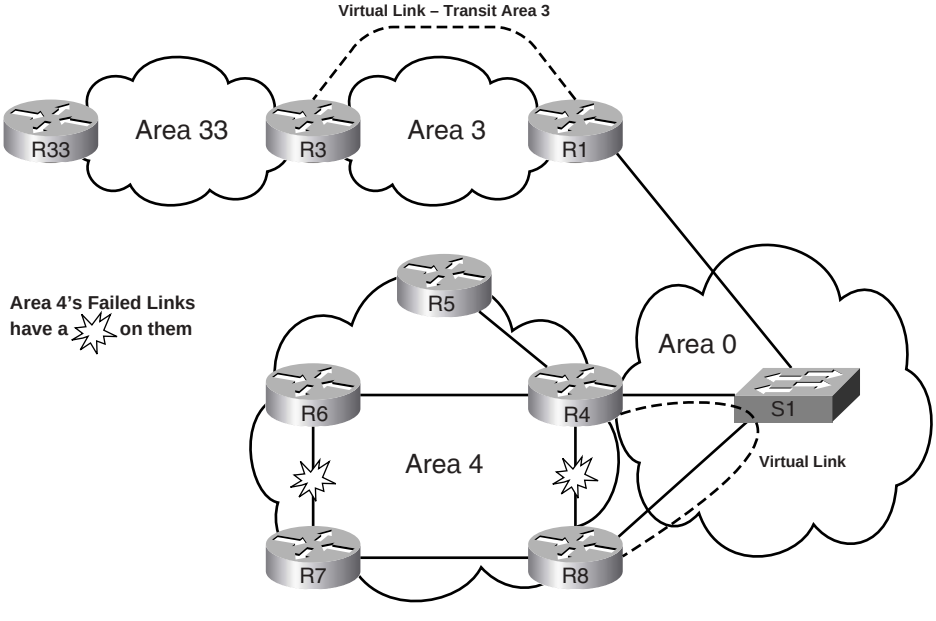
When the **area range** command includes the **not-advertise** keyword, not only are the smaller component subnets not advertised as type 3 LSAs, the summary route is not advertised as a type 3 LSA either. As a result, this command has the same effect as the **area filter-list** command with the **out** keyword, filtering the LSA from going out to any other areas. An example **area range** command is shown at the end of Example 10-11.

Virtual Link Configuration

OSPF requires that each non-backbone area be connected to the backbone area (area 0). OSPF also requires that the routers in each area have a contiguous intra-area path to the other routers in the same area, because without that path, LSA flooding inside the area would fail. However, in some designs, meeting these requirements might be a challenge. You can use OSPF *virtual links* to overcome these problems.

For instance, in the top part of Figure 10-10, area 33 connects only to area 3, and not to area 0.

Figure 10-10 *The Need for Virtual Links*



One straightforward solution to area 33's lack of connection to the backbone area would be to combine areas 3 and 33 into a single area, but OSPF virtual links could solve the problem as well. An OSPF virtual link allows a pair of routers to tunnel OSPF packets inside IP packets, across the IP network, to some other router that is not on the same data link. A virtual link between R3 and R1 gives area 33 a connection to area 0. Also note that R3 becomes an ABR, with a full copy of area 0's LSDB entries.

While the top part of Figure 10-10 simply shows a possibly poor OSPF area design, the lower part shows what could happen just because of a particular set of link failures. The figure shows several failed links that result in a *partitioned* area 4. As a result of the failures, R7 and R8 have no area 4 links connecting to the other three routers in area 4. A virtual link can be used to connect R4 and R8—the requirement being that both R4 and R8 connect to a common and working area—recombining the partitions through the virtual link. (A better solution than the virtual link in this particular topology might be to trunk on R4 and R8, create a small subnet through the LAN switch, and put it in area 4.)

Example 10-12 demonstrates a virtual link configuration between R33 and R1, as shown in Figure 10-10. Note that the virtual link cannot pass through a transit area that is a stubby area, so area 3 has been changed to no longer be a stubby area.

Example 10-12 *Virtual Link Between R3 and R1*

```
! R1 has not learned subnet 10.3.2.0 yet, because area 33 has no link to area 0.
R1# show ip route ospf | incl 10.3.2.0
R1#
! the area virtual link commands point to the other router's RID, and the
! transit area over which the virtual link exists—area 3 in this case. Note that
! timers can be set on the area virtual-link command, as well as authentication.
! It is important when authenticating virtual links to remember that
! the virtual links themselves are in area 0.
! R1 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
router ospf 1
  area 3 virtual-link 3.3.3.3
! R3 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
router ospf 1
  area 3 virtual-link 1.1.1.1
! Below, the status of the virtual link is listed.
R1# show ip ospf virtual-links
Virtual Link OSPF_VL0 to router 3.3.3.3 is up
  Run as demand circuit
  DoNotAge LSA allowed.
  Transit area 3, via interface Serial0/0.3, Cost of using 64
  Transmit Delay is 1 sec, State POINT_TO_POINT,
```

Example 10-12 *Virtual Link Between R3 and R1 (Continued)*

```
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
     Hello due in 00:00:02
     Adjacency State FULL (Hello suppressed)
     Index 3/6, retransmission queue length 0, number of retransmission 1
     First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
     Last retransmission scan length is 1, maximum is 1
     Last retransmission scan time is 0 msec, maximum is 0 msec
! Because R1 and R3 are also sharing the same link, there is a neighbor
! relationship in area 3 that has been seen in the other examples, listed off
! interface s0/0.3. The new virtual link neighbor relationship is shown as well,
! with interface VL0 listed.
R1# show ip ospf nei
! Lines omitted for brevity
Neighbor ID     Pri   State           Dead Time   Address         Interface
3.3.3.3           0   FULL/  -          -        10.3.13.3       OSPF_VL0
3.3.3.3           0   FULL/  -        00:00:10    10.3.13.3       Serial0/0.3
! Below, subnet 10.3.2.0/23, now in area 33, is learned by R1 over the Vlink.
R1# show ip route ospf ¦ incl 10.3.2.0
O IA    10.3.2.0/23 [110/75] via 10.3.13.3, 00:00:10, Serial0/0.3
```

Configuring OSPF Authentication

One of the keys to keeping OSPF authentication configuration straight is to remember that it differs significantly with RIPv2 and EIGRP, although some of the concepts are very similar. The basic rules for configuring OSPF authentication are as follows:

KEY POINT

- Three types are available: type 0 (none), type 1 (clear text), and type 2 (MD5).
- Authentication is enabled per interface using the **ip ospf authentication** interface subcommand.
- The default authentication is type 0 (no authentication).
- The default can be redefined using the **area authentication** subcommand under **router ospf**.
- The keys are configured as interface subcommands.
- Multiple keys are allowed per interface; if configured, OSPF sends multiple copies of each message, one for each key.

Table 10-7 lists the three OSPF authentication types, along with the commands to enable each type, and the commands to define the authentication keys. Note that the three authentication types can be seen in the messages generated by the **debug ip ospf adjacency** command.

Table 10-7 *OSPF Authentication Types*

KEY POINT

Type	Meaning	Enabling Interface Subcommand	Authentication Key Configuration Interface Subcommand
0	None	**ip ospf authentication null**	—
1	Clear text	**ip ospf authentication**	**ip ospf authentication-key** *key-value*
2	MD5	**ip ospf authentication message-digest**	**ip ospf message-digest-key** *key-number* **md5** *key-value*

Example 10-13 (again based on Figure 10-6) shows examples of type 1 and type 2 authentication configuration routers R1 and R2. (Note that S1 and S2 have been shut down for this example, but they would need the same configuration as shown on R1 and R2.) In this example, both R1 and R2 use their fa0/0 interfaces, so their authentication configuration will be identical. As such, the example shows only the configuration on R1.

Example 10-13 *OSPF Authentication Using Only Interface Subcommands*

```
! The two ip ospf commands are the same on R1 and R2. The first enables
! type 1 authentication, and the other defines the simple text key.
interface FastEthernet0/0
 ip ospf authentication
 ip ospf authentication-key key-t1
! Below, the neighbor relationship formed, proving that authentication works.
R1# show ip ospf neighbor fa 0/0
Neighbor ID     Pri   State           Dead Time   Address         Interface
2.2.2.2           1   FULL/BDR        00:00:37    10.1.1.2        FastEthernet0/0
! Next, each interface's OSPF authentication type can be seen in the last line
! or two in the output of the show ip ospf interface command.
R1# show ip ospf int fa 0/0
! Lines omitted for brevity
  Simple password authentication enabled

! Below, both R1 and R2 change to use type 2 authentication. Note that the key
! must be defined with the ip ospf message-digest-key interface subcommand. Key
! chains cannot be used.
interface FastEthernet0/0
 ip ospf authentication message-digest
 ip ospf message-digest-key 1 md5 key-t2
! Below, the command confirms type 2 (MD5) authentication, key number 1.
R1# show ip ospf int fa 0/0 | begin auth
! Lines omitted for brevity
Message digest authentication enabled
Youngest key id is 1
```

Example 10-13 shows two working examples of OSPF authentication, neither of which uses the **area** *number* **authentication** under **router ospf**. Some texts imply that the **area**

authentication command is required—in fact, it was required prior to Cisco IOS Software Release 12.0. In later IOS releases, the **area authentication** command simply tells the router to change that router's default OSPF authentication type for all interfaces in that area. Table 10-8 summarizes the effects and syntax of the **area authentication** router subcommand.

Table 10-8 *Effect of the* **area authentication** *Command on OSPF Interface Authentication Settings*

KEY POINT

area authentication Command	Interfaces in That Area Default to Use…
<no command>	Type 0
area *num* **authentication**	Type 1
area *num* **authentication message-digest**	Type 2

The keys themselves are kept in clear text in the configuration, unless you add the **service password-encryption** global command to the configuration.

The last piece of authentication configuration relates to OSPF virtual links. Because virtual links have no underlying interface on which to configure authentication, authentication is configured on the **area virtual-link** command itself. Table 10-9 shows the variations of the command options for configuring authentication on virtual links. Note that beyond the base **area** *number* **virtual-link** *rid* command, the parameters use similar keywords as compared with the equivalent interface subcommands.

Table 10-9 *Configuring OSPF Authentication on Virtual Links*

KEY POINT

Type	Command Syntax for Virtual Links
0	**area** *num* **virtual-link** *router-id* **authentication null**
1	**area** *num* **virtual-link** *router-id* **authentication authentication-key** *key-value*
2	**area** *num* **virtual-link** *router-id* **authentication message-digest message-digest-key** *key-num* **md5** *key-value*

NOTE OSPF authentication is a good place for tricky CCIE lab questions—ones that can be solved in a few minutes if you know all the intricacies.

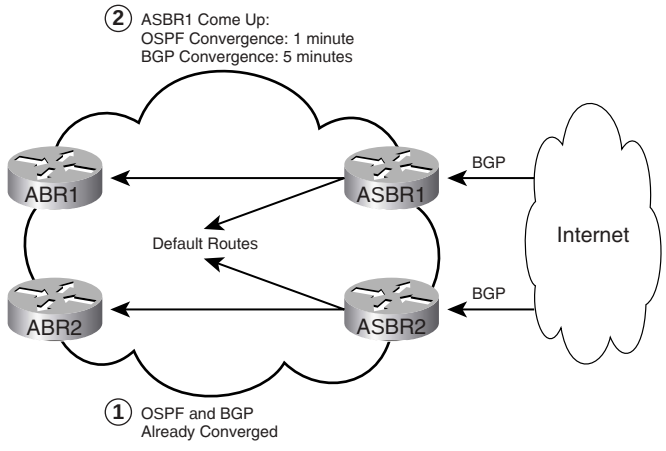
OSPF Stub Router Configuration

Defined in RFC 3137, and first supported in Cisco IOS Software Release 12.2(4)T, the OSPF *stub router* feature—not to be confused with stubby areas—allows a router to either temporarily or permanently be prevented from becoming a transit router. In this context, a transit router is simply

one to which packets are forwarded, with the expectation that the transit router will forward the packet to yet another router. Conversely, non-transit routers only forward packets to and from locally attached subnets.

Figure 10-11 shows one typical case in which a stub router might be useful.

Figure 10-11 *OSPF Stub Router*



Both ASBR1 and ASBR2 advertise defaults into the network, expecting to have the capability to route to the Internet through BGP-learned routes. In this case, ASBR2 is already up, fully converged. However, if ASBR1 reloads, when it comes back up, OSPF is likely to converge faster than BGP. As a result, ASBR1 will advertise its default route, and OSPF routers may send packets to ASBR1, but ASBR1 will end up discarding the packets until BGP converges.

Using the stub router feature on the ASBRs solves the problem by making them advertise infinite metric routes (cost 16,777,215) for any transit routes—either for a configured time period or until BGP convergence is complete. To do so, under **router ospf**, the ASBRs would use either the **max-metric router-lsa on-startup** *announce-time* command or the **max-metric router-lsa on-startup wait-for-bgp** command. With the first version, the actual time period (in seconds) can be set. With the second, OSPF waits until BGP signals that convergence is complete or until 10 minutes pass, whichever comes first.

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most of the Cisco Press *Exam Certification Guides*, this book does not repeat information presented in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 10-10 lists some of the key protocols regarding OSPF.

Table 10-10 *Protocols and Corresponding Standards for Chapter 10*

Name	Standard
OSPF Version 2	RFC 2328
The OSPF Opaque LSA Option	RFC 2370
The OSPF Not-So-Stubby Area (NSSA) Option	RFC 3101
OSPF Stub Router Advertisement	RFC 3137
Traffic Engineering (TE) Extensions to OSPF Version 2	RFC 3630

Table 10-11 lists some of the most popular IOS commands related to the topics in this chapter. Also, refer to Tables 10-7 through 10-9 for references to OSPF authentication commands.

Table 10-11 *Command Reference for Chapter 10*

Command	Command Mode and Description
router ospf *process-id*	Global config; puts user in OSPF configuration mode for that PID.
network *ip-address* [wildcard-*mask*] **area** *area*	OSPF config mode; defines matching parameters, compared to interface IP addresses, to pick interfaces on which to enable OSPF.
ip ospf *process-id* **area** *area-id* [**secondaries none**]	Interface config mode; alternative to the **network** command for enabling OSPF on an interface.
neighbor *ip-address* [**priority** *number*] [**poll-interval** *seconds*] [**cost** *number*] [**database-filter all**]	OSPF config mode; used when neighbors must be defined, it identifies the neighbor's IP address, priority, cost, and poll interval.
auto-cost reference-bandwidth *ref-bw*	OSPF config mode; changes the numerator in the formula to calculate interface cost for all OSPF interfaces on that router.

continues

Table 10-11 *Command Reference for Chapter 10 (Continued)*

Command	Command Mode and Description
router-id *ip-address*	OSPF config mode; statically sets the router ID.
ospf log-neighbor-changes [**detail**]	EIGRP subcommand; displays log messages when neighbor status changes. On by default.
passive-interface [**default**] {*interface-type interface-number*}	OSPF config mode; causes OSPF to stop sending Hellos on the specified interface. OSPF will still advertise the subnet as a stub network.
area *area-id* **stub** [**no-summary**]	OSPF config mode; sets the area type to stub or totally stubby.
area *area-id* **nssa** [**no-redistribution**] [**default-information-originate** [**metric**] [**metric-type**]] [**no-summary**]	OSPF config mode; sets the area type to NSSA or totally NSSA.
area *area-id* **default-cost** *cost*	OSPF config mode; sets the cost of default route created by ABRs and sent into stubby areas.
area *area-id* **nssa translate type7 suppress-fa**	OSPF config mode; sets an NSSA ABR to set the forwarding address to 0.0.0.0 for the type 5 LSAs it translates from type 7.
area *area-id* **range** *ip-address mask* [**advertise** | **not-advertise**] [**cost** *cost*]	OSPF config mode; summarizes routes into a larger prefix at ABRs. Optionally filters type 3 LSAs (**not-advertise** option).
area {*area-id*} **filter-list prefix** {*prefix-list-name* **in** | **out**}	OSPF config mode; filters type 3 LSA creation at ABR.
distribute-list [*ACL*] | [**route-map** *map-tag*] **in** [*int-type* | *int-number*]	OSPF config mode; defines ACL or prefix list to filter what OSPF puts into the routing table.
area *area-id* **virtual-link** *router-id* [**authentication** [**message-digest** | **null**]] [**hello-interval** *seconds*] [**retransmit-interval** *seconds*] [**transmit-delay** *seconds*] [[**dead-interval** *seconds*] [[**authentication-key** *key*] | [**message-digest-key** *key-id* ***md5*** *key*]]	OSPF config mode; creates a virtual link, with typical interface configuration settings to overcome fact that the link is virtual.
ip ospf hello-interval *seconds*	Interface subcommand; sets the interval for periodic Hellos.
ip ospf dead-interval {*seconds* | **minimal hello-multiplier** *multiplier*}	Interface subcommand; defines the dead interval, or optionally the minimal dead interval of 1 second.

Table 10-11 *Command Reference for Chapter 10 (Continued)*

Command	Command Mode and Description
ip ospf name-lookup	Global command; causes the router to use DNS to correlate RIDs to host names for **show** command output.
ip ospf cost *interface-cost*	Interface subcommand; sets the cost.
ip ospf mtu-ignore	Interface subcommand; tells the router to ignore the check for equal MTUs that occurs when sending DD packets.
ip ospf network {**broadcast** | **non-broadcast** | {**point-to-multipoint** [**non-broadcast**] | **point-to-point**}}	Interface subcommand; sets the OSPF network type on an interface.
ip ospf priority *number-value*	Interface subcommand; sets the OSPF priority on an interface.
ip ospf retransmit-interval *seconds*	Interface subcommand; sets the time between LSA transmissions for adjacencies belonging to an interface.
ip ospf transmit-delay *seconds*	Interface subcommand; defines the estimated time expected for the transmission of an LSU.
max-metric router-lsa [**on-startup** {*announce-time* | **wait-for-bgp**}]	OSPF config mode; configures a stub router, delaying the point at which it can become a transit router.
show ip ospf border-routers	User mode; displays hidden LSAs for ABRs and ASBRs.
show ip ospf [*process-id* [*area-id*]] **database**	User mode; has many options not shown here. Displays the OSPF LSDB.
show ip ospf neighbor [*interface-type interface-number*] [*neighbor-id*] [**detail**]	User mode; lists information about OSPF neighbors.
show ip ospf [*process-id*] **summary-address**	User mode; lists information about route summaries in OSPF.
show ip ospf virtual-links	User mode; displays status and info about virtual links.
show ip route ospf	User mode; displays all OSPF routes in the IP routing table.
show ip ospf interface [*interface-type interface-number*] [**brief**]	User mode; lists OSPF protocol timers and statistics per interface.
show ip ospf statistics [**detail**]	User mode; displays OSPF SPF calculation statistics.

continues

Table 10-11 *Command Reference for Chapter 10 (Continued)*

Command	Command Mode and Description
clear ip ospf [*pid*] {**process** | **redistribution** | **counters** [**neighbor** [*neighbor-interface*] [*neighbor-id*]]}	Enable mode; restarts the OSPF process, clears redistributed routes, or clears OSPF counters.
debug ip ospf hello	Enable mode; displays messages regarding Hellos, including Hello parameter mismatches.
debug ip ospf adj	Enable mode; displays messages regarding adjacency changes.
show ip ospf interface [*type number*] [**brief**]	User mode; lists many interface settings.

Table 10-12 summarizes many OSPF timers and their meaning.

Table 10-12 *OSPF Timer Summary*

Timer	Meaning
MaxAge	The maximum time an LSA can be in a router's LSDB, without receiving a newer copy of the LSA, before the LSA is removed. Default is 3600 seconds.
LSRefresh	The timer interval per LSA on which a router refloods an identical LSA, except for a 1-larger sequence number, to prevent the expiration of MaxAge. Default is 1800 seconds.
Hello	Per interface; time interval between Hellos. Default is 10 or 30 seconds, depending on interface type.
Dead	Per interface; time interval in which a Hello should be received from a neighbor. If not received, the neighbor is considered to have failed. Default is four times Hello.
Wait	Per interface; set to the same number as the dead interval. Defines the time a router will wait to get a Hello asserting a DR after reaching a 2WAY state with that neighbor.
Retransmission	Per interface; the time between sending an LSU, not receiving an acknowledgement, and then resending the LSU. Default is 5 seconds.
Inactivity	Countdown timer, per neighbor, used to detect when a neighbor has not been heard from for a complete dead interval. It starts equal to the dead interval, counts down, and is reset to be equal to the dead interval when each Hello is received.
Poll Interval	On NBMA networks, the period at which Hellos are sent to a neighbor when the neighbor is down. Default is 60 seconds.

Table 10-12 *OSPF Timer Summary (Continued)*

Timer	Meaning
Flood (Pacing)	Per interface; defines the interval between successive LSUs when flooding LSAs. Default is 33 ms.
Retransmission (Pacing)	Per interface; defines the interval between retransmitted packets as part of a single retransmission event. Default is 66 ms.
Lsa-group (Pacing)	Per OSPF process. LSA's LSRefresh intervals time out independently. This timer improves LSU reflooding efficiency by waiting, collecting several LSAs whose LSRefresh timers expire, and flooding all these LSAs together. Default is 240 seconds.

Table 10-13 lists OSPF neighbor states and their meaning.

Table 10-13 *OSPF Neighbor States*

State	Meaning
Down	No Hellos have been received from this neighbor for more than the dead interval.
Attempt	This router is sending Hellos to a manually configured neighbor.
Init	A Hello has been received from the neighbor, but it did not have the router's RID in it. This is a permanent state when Hello parameters do not match.
2WAY	A Hello has been received from the neighbor, and it has the router's RID in it. This is a stable state for pairs of DROther neighbors.
ExStart	Currently negotiating the DD sequence numbers and master/slave logic used for DD packets.
Exchange	Finished negotiating, and currently exchanging DD packets.
Loading	All DD packets exchanged, and currently pulling the complete LSDB entries with LSU packets.
Full	Neighbors are adjacent (fully adjacent), and should have identical LSDB entries for the area in which the link resides. Routing table calculations begin.

Table 10-14 lists several key OSPF numeric values.

Table 10-14 *OSPF Numeric Ranges*

Setting	Range of Values
Single interface cost	1 to 65,535 ($2^{16} - 1$)
Complete route cost	1 to 16,777,215 ($2^{24} - 1$)
Infinite route cost	16,777,215 ($2^{24} - 1$)
Reference bandwidth (units: Mbps)	1 to 4,294,967
OSPF PID	1 to 65,535 ($2^{16} - 1$)

Memory Builders

The *CCIE Routing and Switching* written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

LSDB, Dijkstra, link-state routing protocol, LSA, LSU, DD, Hello, LSAck, RID, neighbor state, neighbor, adjacent, fully adjacent, 2-Way, 224.0.0.5, 224.0.0.6, area, stub area type, network type, external route, E1 route, E2 route, Hello timer, Dead Time/Interval, sequence number, DR, BDR, DROther, priority, LSA flooding, DR election, SPF calculation, partial SPF calculation, full SPF calculation, LSRefresh, hello time/interval, MaxAge, ABR, ASBR, internal router, backbone area, transit network, stub network, LSA type, stub area, NSSA, totally stubby area, totally NSSA area, virtual link, stub router, transit router, SPF algorithm, All OSPF DR Routers, All OSPF Routers

Refer to the CD-based glossary to check your answers.

Further Reading

Jeff Doyle's *Routing TCP/IP*, Volume I, Second Edition—every word a must for CCIE Routing and Switching.

Cisco OSPF Command and Configuration Handbook, by Dr. William Parkhurst, covers every OSPF-related command, with examples of each one.

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- IP Routing
  - — OSPF
  - — EIGRP
  - — Route Filtering
  - — RIPv2
  - — The use of **show** and **debug** commands

CHAPTER 11

IGP Route Redistribution, Route Summarization, and Default Routing

This chapter covers several topics related to the use of multiple IGP routing protocols. IGPs can use default routes to pull packets toward a small set of routers, with those routers having learned routes from some external source. IGPs can use route summarization with a single routing protocol, but it is often used at redistribution points between IGPs as well. Finally, route redistribution by definition involves moving routes from one routing source to another. This chapter takes a look at each topic.

"Do I Know This Already?" Quiz

Table 11-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 11-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Route Maps, Prefix Lists, and Administrative Distance	1–2	
Route Redistribution	3–6	
Route Summarization	7–8	
Default Routes	9	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. A route map has several clauses. A route map's first clause has a **permit** action configured. The **match** command for this clause refers to an ACL that matches route 10.1.1.0/24 with a **permit** action, and matches route 10.1.2.0/24 with a **deny** action. If this route map is used for route redistribution, which of the following are true?

   a. The route map will attempt to redistribute 10.1.1.0/24.

   b. The question does not supply enough information to determine if 10.1.1.0/24 is redistributed.

   c. The route map will not attempt to redistribute 10.1.2.0/24.

   d. The question does not supply enough information to determine if 10.1.2.0/24 is redistributed.

2. Which of the following routes would be matched by this prefix list command: **ip prefix-list fred permit 10.128.0.0/9 ge 20**?

   a. 10.1.1.0 255.255.255.0

   b. 10.127.1.0 255.255.255.0

   c. 10.200.200.192 255.255.255.252

   d. 10.128.0.0 255.255.240.0

   e. None of these answers is correct.

3. A router is using the configuration shown below to redistribute routes. This router has several working interfaces with IP addresses in network 10.0.0.0, and has learned some network 10 routes with EIGRP and some with OSPF. Which of the following is true about the redistribution configuration?

```
router eigrp 1
 network 10.0.0.0
 redistribute ospf 2
!
router ospf 2
 network 10.0.0.0 0.255.255.255 area 3
 redistribute eigrp 1 subnets

R1# show ip route 15.0.0.0
Routing entry for 15.0.0.0/24, 5 known subnets
  Attached (2 connections)
  Redistributing via eigrp 1

O E1    10.6.11.0 [110/84] via 10.1.6.6, 00:21:52, Serial0/0/0.6
O E2    10.6.12.0 [110/20] via 10.1.6.6, 00:21:52, Serial0/0/0.6
C       10.1.6.0 is directly connected, Serial0/0/0.6
O IA    10.1.2.0 [110/65] via 10.1.1.5, 00:21:52, Serial0/0/0.5
C       10.1.1.0 is directly connected, Serial0/0/0.5
```

   a. EIGRP will not advertise any additional routes due to redistribution.

   b. OSPF will not advertise any additional routes due to redistribution.

c. Routes redistributed into OSPF will be advertised as E1 routes.

d. The **redistribute ospf 2** command would be rejected due to missing parameters.

4. Examine the following router configuration and excerpt from its IP routing table. Which routes could be redistributed into OSPF?

```
router eigrp 1
 network 12.0.0.0
router ospf 2
 redistribute eigrp 1 subnets
 network 13.0.0.0 0.255.255.255 area 3
An excerpt from the routing table is shown next:
C       12.1.6.0 is directly connected, Serial0/0/0.6
D       12.0.0.0/8 [90/2172416] via 13.1.1.1, 00:01:30, Serial0/0/0.5
C       13.1.1.0 is directly connected, Serial0/0/0.5
```

a. 12.1.6.0

b. 12.0.0.0

c. 13.1.1.0

d. None of the above

5. Two corporations merged. The network engineers decided to redistribute between one company's EIGRP network and the other company's OSPF network, using two mutually redistributing routers (R1 and R2) for redundancy. Assume that as many defaults as is possible are used for the redistribution configuration. Assume that one of the subnets in the OSPF domain is 10.1.1.0/24. Which of the following is true about a possible suboptimal route to 10.1.1.0/24 on R1—a route that sends packets through the EIGRP domain, and through R2 into the OSPF domain?

a. The suboptimal routes will occur unless the configuration filters routes at R1.

b. R1's administrative distance must be manipulated, such that OSPF routes have an administrative distance less than EIGRP's default of 90.

c. EIGRP prevents the suboptimal routes by default.

d. Using route tags is the only way to prevent the suboptimal routes.

6. Which of the following statements is true about the type of routes created when redistributing routes?

a. Routes redistributed into OSPF default to be external type 2.

b. Routes redistributed into EIGRP default to external, but can be set to internal with a route map.

c. Routes redistributed into RIP are external by default.

d. Routes redistributed into OSPF by a router in an NSSA area default to be external type 1.

7. Which of the following is not true about route summarization?

   a. The advertised summary is assigned the same metric as the lowest-metric component subnet.

   b. The router does not advertise the summary when its routing table does not have any of the component subnets.

   c. The router does not advertise the component subnets.

   d. Summarization, when used with redistribution, prevents all cases of suboptimal routes.

8. Which of the following is true of route summarization?

   a. OSPF can summarize routes only on ABRs and ASBRs.

   b. EIGRP summarization is configured with a **router eigrp** subcommand.

   c. RIPv2 summarization has the same features as EIGRP summarization, other than the syntax differences in the **ip summary-address** command.

   d. RIPv1 allows the **ip summary-address** command to be used.

9. Which of the following is/are true regarding the **default-information originate** router subcommand?

   a. It is not supported by EIGRP.

   b. It causes OSPF to advertise a default route, but only if a static route to 0.0.0.0/0 is in that router's routing table.

   c. The **always** keyword on the **default-information originate** command, when used for OSPF, means OSPF will originate a default route even if no default route exists in its own IP routing table.

   d. None of the other answers are correct.

Foundation Topics

Route Maps, Prefix Lists, and Administrative Distance

Route maps, IP prefix lists, and administrative distance (AD) must be well understood to do well with route redistribution topics on the CCIE Routing and Switching written exam. This section focuses on the tools themselves, followed by coverage of route redistribution.

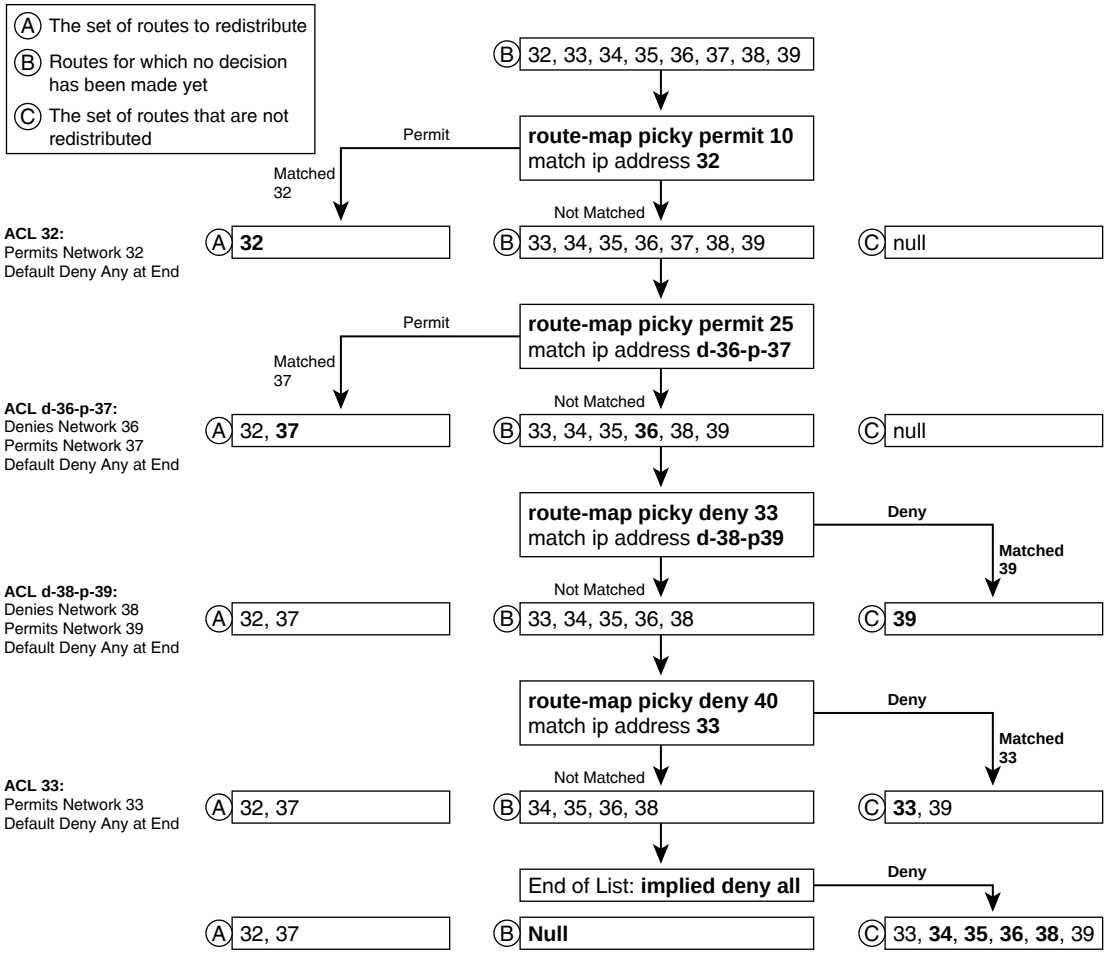
Configuring Route Maps with the route-map Command

Route maps provide programming logic similar to the If/Then/Else logic seen in other programming languages. A single route map has one or more **route-map** commands in it, and routers process **route-map** commands in sequential order based on sequence numbers. Each **route-map** command has underlying matching parameters, configured with the aptly named **match** command. (To match all packets, the **route-map** clause simply omits the **match** command.) Each **route-map** command also has one or more optional **set** commands that you can use to manipulate information—for instance, to set the metric for some redistributed routes. The general rules for route maps are as follows:

- Each **route-map** command must have an explicitly configured name, with all commands that use the same name being part of the same route map.
- Each **route-map** command has an action (**permit** or **deny**).
- Each **route-map** command in the same route map has a unique sequence number, allowing deletion and insertion of single **route-map** commands.
- When a route map is used for redistribution, the route map processes routes taken from the then-current routing table.
- The route map is processed sequentially based on the sequence numbers.
- Once a particular route is matched by the route map, it is not processed beyond that matching **route-map** command (specific to route redistribution).
- When a route is matched in a **route-map** statement, if the **route-map** command has a **permit** parameter, the route is redistributed (specific to route redistribution).
- When a route is matched in a **route-map** statement, if the **route-map** statement has a **deny** parameter, the route is not redistributed (specific to route redistribution).

Route maps can be confusing at times, especially when using the **deny** option on the **route-map** command. To help make sure the logic is clear before getting into redistribution, Figure 11-1 shows a logic diagram for an example route map. (This example is contrived to demonstrate some nuances of route map logic; a better, more efficient route map could be created to achieve the same results.) In the figure, R1 has eight loopback interfaces configured to be in class A networks 32 through 39. Figure 11-1 shows how the contrived **route-map picky** would process the routes.

Figure 11-1 *Route Map Logic Example*



First, a few clarifications about the meaning of Figure 11-1 are in order. The top of the figure begins with the set of connected networks (32 through 39), labeled with a "B," which is the set of routes still being considered for redistribution. Moving down the figure, four separate **route-map** commands sit inside this single route map. Each **route-map** clause (the clause includes the underlying **match** and **set** commands) in turn moves routes from the list of possible routes ("B")

to either the list of routes to redistribute (“A”) or the list to not redistribute (“C”). By the bottom of the figure, all routes will be noted as either to be redistributed or not to be redistributed.

The route map chooses to redistribute a route only if the **route-map** command has a **permit** option; the only time a **route-map** clause chooses to *not redistribute* a route is when the clause has a **deny** option. Ignoring the matching logic for a moment, the first two **route-map** commands (sequence numbers 10 and 25) use the **permit** option. As a result of those clauses, routes are either added to the list of routes to redistribute (“A”) or left in the list of candidate routes (“B”). The third and fourth clauses (sequence numbers 33 and 40) use the **deny** option, so those clauses cause routes to be either added to the list of routes to not redistribute (“C”), or left in the list of candidate routes (“B”). In effect, once a **route-map** clause has matched a route, that route is flagged either as to be redistributed or as not to be redistributed, and the route is no longer processed by the route map.

One point that can sometimes be confused is that if a route is denied by an ACL used by a **match** command, it does not mean that the route is prevented from being redistributed. For instance, the **match ip address 32** command in clause 10 refers to ACL 32, which has one explicit *access control entry (ACE)* that matches network 32, with a **permit** action. Of course, ACL 32 has an implied deny all at the end, so ACL 32 permits network 32, and denies 33 through 39. However, denying networks 33 through 39 in the ACL does not mean that those routes are not redistributed—it simply means that those routes do not match **route-map** clause 10, so those routes are eligible for consideration by a later **route-map** clause.

The following list summarizes the key points about route map logic when used for redistribution:

KEY POINT

- **route-map** commands with the **permit** option either cause a route to be redistributed or leave the route in the list of routes to be examined by the next **route-map** clause.
- **route-map** commands with the **deny** option either filter the route or leave the route in the list of routes to be examined by the next **route-map** clause.
- If a clause’s **match** commands use an ACL, an ACL match with the **deny** action does not cause the route to be filtered. Instead, it just means that route does not match that particular **route-map** clause.
- The **route-map** command includes an implied deny all clause at the end; to configure a permit all, use the **route-map** command, with a **permit** action, but without a **match** command.

Route Map match Commands for Route Redistribution

Route maps use the **match** command to define the fields and values used for matching the routes being processed. If more than one **match** command is configured in a single **route-map** clause, a route is matched only if all the **match** commands’ parameters match the route. The logic in each

match command itself is relatively straightforward. Table 11-2 lists the **match** command options when used for IGP route redistribution.

Table 11-2 **match** *Command Options for IGP Redistribution*

match Command	Description
match interface *interface-type interface-number* [*... interface-type interface-number*]	Looks at outgoing interface of routes
*__match ip address__ {[*access-list-number* | *access-list-name*] | *prefix-list prefix-list-name*}	Examines route prefix and prefix length
*__match ip next-hop__ {*access-list-number* | *access-list-name*}	Examines route's next-hop address
*__match ip route-source__ {*access-list-number* | *access-list-name*}	Matches advertising router's IP address
match metric *metric-value*	Matches route's metric
match route-type {**internal** | **external** [**type-1** | **type-2**] | **level-1** | **level-2**}	Matches route type
match tag *tag-value* [*...tag-value*]	Tag must have been set earlier

*Can reference multiple numbered and named ACLs on a single command.

Route Map set Commands for Route Redistribution

When used for redistribution, route maps have an implied action—either to allow the route to be redistributed or to filter the route so that it is not redistributed. As described earlier in this chapter, that choice is implied by the **permit** or **deny** option on the **route-map** command. Route maps can also change information about the redistributed routes by using the **set** command. Table 11-3 lists the **set** command options when used for IGP route redistribution.

Table 11-3 **set** *Command Options for IGP Redistribution*

set Command	Description
set level {**level-1** | **level-2** | **level-1-2** | **stub-area** | **backbone**}	Defines database(s) into which the route is redistributed
set metric *metric-value*	Sets the route's metric for OSPF, RIP, and IS-IS
set metric *bandwidth delay reliability loading mtu*	Sets the IGRP/EIGRP route's metric values
set metric-type {**internal** | **external** | **type-1** | **type-2**}	Sets type of route for IS-IS and OSPF
set tag *tag-value*	Sets the unitless tag value in the route

IP Prefix Lists

IP prefix lists provide mechanisms to match two components of an IP route:

- The route prefix (the subnet number)
- The prefix length (the subnet mask)

The **redistribute** command cannot directly reference a prefix list, but a route map can refer to a prefix list by using the **match** command.

A prefix list itself has similar characteristics to a route map. The list consists of one or more statements with the same text name. Each statement has a sequence number to allow deletion of individual commands, and insertion of commands into a particular sequence position. Each command has a **permit** or **deny** action—but because it is used only for matching packets, the **permit** or **deny** keyword just implies whether a route is matched (**permit**) or not (**deny**). The generic command syntax is as follows:

```
ip prefix-list list-name [seq seq-value] {deny network/length | permit network/
length}[ge ge-value] [le le-value]
```

The sometimes tricky and interesting part of working with prefix lists is that the meaning of the *network/length*, *ge-value*, and *le-value* parameters changes depending on the syntax. The *network/length* parameters define the values to use to match the route prefix. For example, a *network/length* of 10.0.0.0/8 means "any route that begins with a 10 in the first octet." The **ge** and **le** options are used for comparison to the prefix length—in other words, to the number of binary 1s in the subnet mask. For instance, **ge 20 le 22** matches only routes whose masks are /20, /21, or /22. So, prefix list logic can be summarized into a two-step comparison process for each route:

1. The *route's prefix* must be within the range of addresses implied by the **prefix-list** command's *network/length* parameters.
2. The *route's prefix length* must match the *range of prefixes* implied by the **prefix-list** command.

The potentially tricky part of the logic relates to knowing the range of prefix lengths checked by this logic. The range is defined by the *ge-value* and *le-value* parameters, which stand for *greater-than-or-equal-to* and *less-than-or-equal-to*. Table 11-4 formalizes the logic, including the default values for *ge-value* and *le-value*. In the table, note that *conf-length* refers to the prefix length configured in the *network/prefix* (required) parameter, and *route-length* refers to the prefix length of a route being examined by the prefix list.

Table 11-4 *LE and GE Parameters on IP Prefix List, and the Implied Range of Prefix Lengths*

Prefix List Parameters	Range of Prefix Lengths
Neither	*conf-length* = *route-length*
Only **le**	*conf-length* <= *route-length* <= *le-value*
Only **ge**	*ge-value* <= *route-length* <= 32
Both **ge** and **le**	*ge-value* <= *route-length* <= *le-value*

Several examples can really help nail down prefix list logic. The following routes will be examined by a variety of prefix lists, with the routes numbered for easier reference:

1. 10.0.0.0/8
2. 10.128.0.0/9
3. 10.1.1.0/24
4. 10.1.2.0/24
5. 10.128.10.4/30
6. 10.128.10.8/30

Next, Table 11-5 shows the results of seven different one-line prefix lists applied to these six example routes. The table lists the matching parameters in the **prefix-list** commands, omitting the first part of the commands. The table explains which of the six routes would match the listed prefix list, and why.

Table 11-5 *Example Prefix Lists Applied to the List of Routes*

prefix-list Command Parameters	Routes Matched	Results
10.0.0.0/8	1	Without **ge** or **le** configured, both the prefix (10.0.0.0) and length (8) must be an exact match.
10.128.0.0/9	None	Without **ge** or **le** configured, the prefix (10.128.0.0) and length (9) must be an exact match, so none of the routes match.
10.0.0.0/8 ge 9	2–6	The 10.0.0.0/8 means "all routes whose first octet is 10," effectively representing an address range. The prefix length must be between 9 and 32, inclusive.
10.0.0.0/8 ge 24 le 24	3, 4	The 10.0.0.0/8 means "all routes whose first octet is 10," and the prefix range is 24 to 24—meaning only routes with prefix length 24.

Table 11-5 *Example Prefix Lists Applied to the List of Routes (Continued)*

prefix-list Command Parameters	Routes Matched	Results
10.0.0.0/8 le 28	1–4	The prefix length needs to be between 8 and 28, inclusive.
0.0.0.0/0	None	0.0.0.0/0 means "match all prefixes, with prefix length of exactly 0." So, it would match all routes' prefixes, but none of their prefix lengths. Only a default route would match this prefix list.
0.0.0.0/0 le 32	All	The range implied by 0.0.0.0/0 is all IPv4 addresses. The **le 32** then implies any prefix length between 0 and 32, inclusive. This is the syntax for "match all" prefix list logic.

Administrative Distance

A single router can learn routes using multiple IP routing protocols, as well as via connected and static routes. When a router learns a particular route from multiple sources, the router cannot use the metrics to determine the best route, because the metrics are based on different units. So, the router uses each route's *administrative distance (AD)* to determine which is best, with the lower number being better. Table 11-6 lists the default AD values for the various routing sources.

Table 11-6 *Administrative Distances*

KEY POINT

Route Type	Administrative Distance
Connected	0
Static	1
EIGRP summary route	5
EBGP	20
EIGRP (internal)	90
IGRP	100
OSPF	110
IS-IS	115
RIP	120
EIGRP (external)	170
iBGP (external)	200
Unreachable	255

The defaults can be changed by using the **distance** command. The command differs amongst all three IGPs covered in this book. The generic versions of the **distance** router subcommand for RIP, EIGRP, and OSPF, respectively, are as follows:

```
distance distance
distance eigrp internal-distance external-distance
distance ospf {[intra-area dist1] [inter-area dist2] [external dist3]}
```

As you can see, EIGRP and OSPF can set a different AD depending on the type of route as well, whereas RIP cannot. You can also use the **distance** command to set a router's view of the AD per route, as is covered later in this chapter.

Route Redistribution

Although using a single routing protocol throughout an enterprise might be preferred, many enterprises use multiple routing protocols due to business mergers and acquisitions, organizational history, or in some cases for technical reasons. Route redistribution allows one or more routers to take routes learned via one routing protocol and advertise those routes via another routing protocol so that all parts of the internetwork can be reached.

To perform redistribution, one or more routers run both routing protocols, with each routing protocol placing routes into that router's routing table. Then, each routing protocol can take all or some of the other routing protocol's routes from the routing table and advertise those routes. This section begins by looking at the mechanics of how to perform simple redistribution on a single router, and ends with discussion of tools and issues that matter most when redistributing on multiple routers.

The Mechanics of the redistribute Command

The **redistribute** router subcommand tells one routing protocol to take routes from another routing protocol. This command can simply redistribute all routes or, by using matching logic, redistribute only a subset of the routes. The **redistribute** command also supports actions for setting some parameters about the redistributed routes—for example, the metric.

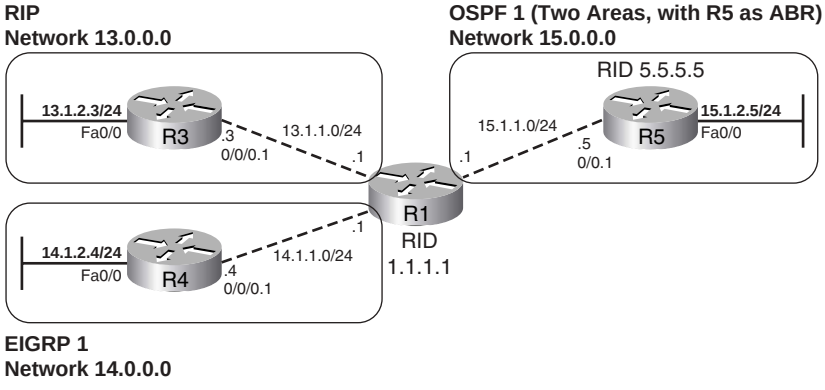
The full syntax of the **redistribute** command is as follows:

```
redistribute protocol [process-id] [level-1 | level-1-2 | level-2] [as-number] [metric
metric-value] [metric-type type-value] [match {internal | external 1 | external 2}] [tag
tag-value] [route-map map-tag] [subnets]
```

The **redistribute** command identifies the routing source from which routes are taken, and the **router** command identifies the routing process into which the routes are advertised. For example, the command **redistribute eigrp 1** tells the router to *take routes from* EIGRP process 1; if that command were under **router rip**, the routes would be redistributed into RIP, enabling other RIP routers in the network to see some or all routes coming from EIGRP AS 1.

The **redistribute** command has a lot of other parameters as well, most of which will be described in upcoming examples. The first few examples use the network shown in Figure 11-2. In this network, each IGP uses a different class A network just to make the results of redistribution more obvious. Also note that the numbering convention is such that each of R1's connected WAN subnets has 1 as the third octet, and each LAN subnet off R3, R4, and R5 has 2 as the third octet.

Figure 11-2 *Sample Network for Default Route Examples*



Redistribution Using Default Settings

The first example configuration meets the following design goals:

- R1 redistributes between each pair of IGPs—RIP, EIGRP, and OSPF.
- Default metrics are used whenever possible; when required, the metrics are configured on the **redistribute** command.
- Redistribution into OSPF uses the non-default **subnets** parameter, which causes subnets to be advertised into OSPF.
- All other settings use default values.

Example 11-1 shows R1's configuration for each routing protocol, along with **show** commands from all four routers to highlight the results of the redistribution.

Example 11-1 *Route Redistribution with Minimal Options*

```
! EIGRP redistributes from OSPF (process ID 1) and RIP. EIGRP must
! set the metric, as it has no default values. It also uses the
! no auto-summary command so that subnets will be redistributed into
! EIGRP.
```

continues

Example 11-1 *Route Redistribution with Minimal Options (Continued)*

```
router eigrp 1
 redistribute ospf 1 metric 1544 5 255 1 1500
 redistribute rip metric 1544 5 255 1 1500
 network 14.0.0.0
 no auto-summary
! OSPF redistributes from EIGRP (ASN 1) and RIP. OSPF defaults the
! metric to 20 for redistributed IGP routes. It must also use the
! subnets option in order to redistribute subnets.
router ospf 1
 router-id 1.1.1.1
 redistribute eigrp 1 subnets
 redistribute rip subnets
 network 15.0.0.0 0.255.255.255 area 0
! RIP redistributes from OSPF (process ID 1) and EIGRP (ASN 1). RIP
! must set the metric, as it has no default values. It also uses the
! no auto-summary command so that subnets will be redistributed into
! EIGRP.
router rip
 version 2
 redistribute eigrp 1 metric 2
 redistribute ospf 1 metric 3
 network 13.0.0.0
 no auto-summary
! R1 has a connected route (x.x.1.0) in networks 13, 14, and 15, as well as
! an IGP-learned route (x.x.2.0).
R1# show ip route
! lines omitted for brevity
     10.0.0.0/24 is subnetted, 1 subnets
C       10.1.1.0 is directly connected, FastEthernet0/0
     13.0.0.0/24 is subnetted, 2 subnets
C       13.1.1.0 is directly connected, Serial0/0/0.3
R       13.1.2.0 [120/1] via 13.1.1.3, 00:00:07, Serial0/0/0.3
     14.0.0.0/24 is subnetted, 2 subnets
D       14.1.2.0 [90/2172416] via 14.1.1.4, 00:58:20, Serial0/0/0.4
C       14.1.1.0 is directly connected, Serial0/0/0.4
     15.0.0.0/24 is subnetted, 2 subnets
O IA    15.1.2.0 [110/65] via 15.1.1.5, 00:04:25, Serial0/0/0.5
C       15.1.1.0 is directly connected, Serial0/0/0.5
```

```
! R3 learned two routes each from nets 14 and 15.
! Compare the metrics set on R1's RIP redistribute command to the metrics below.
R3# show ip route rip
     14.0.0.0/24 is subnetted, 2 subnets
R       14.1.2.0 [120/2] via 13.1.1.1, 00:00:19, Serial0/0/0.1
R       14.1.1.0 [120/2] via 13.1.1.1, 00:00:19, Serial0/0/0.1
     15.0.0.0/24 is subnetted, 2 subnets
R       15.1.2.0 [120/3] via 13.1.1.1, 00:00:19, Serial0/0/0.1
R       15.1.1.0 [120/3] via 13.1.1.1, 00:00:19, Serial0/0/0.1
```

Example 11-1 *Route Redistribution with Minimal Options (Continued)*

```
! R4 learned two routes each from nets 13 and 15.
! EIGRP injected the routes as external (EX), which are considered AD 170.
R4# show ip route eigrp
     13.0.0.0/24 is subnetted, 2 subnets
D EX    13.1.1.0 [170/2171136] via 14.1.1.1, 00:09:57, Serial0/0/0.1
D EX    13.1.2.0 [170/2171136] via 14.1.1.1, 00:09:57, Serial0/0/0.1
     15.0.0.0/24 is subnetted, 2 subnets
D EX    15.1.2.0 [170/2171136] via 14.1.1.1, 01:00:27, Serial0/0/0.1
D EX    15.1.1.0 [170/2171136] via 14.1.1.1, 01:00:27, Serial0/0/0.1
```

```
! R5 learned two routes each from nets 13 and 14.
! OSPF by default injected the routes as external type 2, cost 20.
R5# show ip route ospf
     13.0.0.0/24 is subnetted, 2 subnets
O E2    13.1.1.0 [110/20] via 15.1.1.1, 00:36:12, Serial0/0.1
O E2    13.1.2.0 [110/20] via 15.1.1.1, 00:36:12, Serial0/0.1
     14.0.0.0/24 is subnetted, 2 subnets
O E2    14.1.2.0 [110/20] via 15.1.1.1, 00:29:56, Serial0/0.1
O E2    14.1.1.0 [110/20] via 15.1.1.1, 00:36:12, Serial0/0.1
! As a backbone router, OSPF on R1 created type 5 LSAs for the four E2 subnets.
! If R1 had been inside an NSSA stub area, it would have created type 7 LSAs.
R5# show ip ospf data ¦ begin Type-5
        Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
13.1.1.0        1.1.1.1         1444        0x80000002 0x000785 0
13.1.2.0        1.1.1.1         1444        0x80000002 0x00FB8F 0
14.1.1.0        1.1.1.1         1444        0x80000002 0x00F991 0
14.1.2.0        1.1.1.1         1444        0x80000002 0x00EE9B 0
```

Metrics must be set via configuration when redistributing into RIP and EIGRP, whereas OSPF uses default values. In the example, the two **redistribute** commands under **router rip** used hop counts of 2 and 3 just so the metrics could be easily seen in the **show ip route** command output on R3. The EIGRP metric in the **redistribute** command must include all five metric components, even if the last three are ignored by EIGRP's metric calculation (as they are by default). The command **redistribute rip metric 1544 5 255 1 1500** lists EIGRP metric components of bandwidth, delay, reliability, load, and MTU, in order. OSPF defaults to cost 20 when redistributing from an IGP, and 1 when redistributing from BGP.

The **redistribute** command redistributes only routes in that router's current IP routing table. When redistributing from a given routing protocol, the **redistribute** command takes routes listed in the IP routing table as being learned from that routing protocol. Interestingly, the **redistribute** command can also pick up connected routes. For example, R1 has an OSPF route to 15.1.2.0/24, and a connected route to 15.1.1.0/24. However, R3 (RIP) and R4 (EIGRP) redistribute both of these routes—the OSPF-learned route and one connected route—as a result of their respective

redistribute ospf commands. As it turns out, the **redistribute** command causes the router to use the following logic to choose which routes to redistribute from a particular IGP protocol:

KEY POINT

1. Take all routes in my routing table that were learned by the routing protocol from which routes are being redistributed.
2. Take all connected subnets matched by that routing protocol's **network** commands.

Example 11-1 shows several instances of exactly how this two-part logic works. For instance, R3 (RIP) learns about connected subnet 14.1.1.0/24, because RIP redistributes from EIGRP, and R1's EIGRP **network 14.0.0.0** command matches that subnet.

The **redistribute** command includes a **subnets** option, but only OSPF needs to use it. By default, when redistributing into OSPF, OSPF redistributes only routes for classful networks, ignoring subnets. By including the **subnets** option, OSPF redistributes subnets as well. The other IGPs redistribute subnets automatically; however, if at a network boundary, the RIP or EIGRP **auto-summary** setting would still cause summarization to use the classful network. In Example 11-1, if either RIP or EIGRP had used **auto-summary**, each redistributed network would show just the classful networks. For example, if RIP had configured **auto-summary** in Example 11-1, R3 would have a route to networks 14.0.0.0/8 and 15.0.0.0/8, but no routes to subnets inside those class A networks.

Setting Metrics, Metric Types, and Tags

Cisco IOS provides three mechanisms for setting the metrics of redistributed routes, as follows:

KEY POINT

1. Call a route map from the **redistribute** command, with the route map using the **set metric** command. This method allows different metrics for different routes.
2. Use the **metric** option on the **redistribute** command. This sets the same metric for all routes redistributed by that **redistribute** command.
3. Use the **default-metric** command under the **router** command. This command sets the metric for all redistributed routes whose metric was not set by either of the other two methods.

The list implies the order of precedence if more than one method defines a metric. For instance, if a route's metric is set by all three methods, the route map's metric is used. If the metric is set on the **redistribute** command and there is a **default-metric** command as well, the setting on the **redistribute** command takes precedence.

The **redistribute** command also allows a setting for the *metric-type* option, which really refers to the route type. For example, routes redistributed into OSPF must be OSPF external routes, but they can be either external type 1 (E1) or type 2 (E2) routes. Table 11-7 summarizes the defaults for metrics and metric types.

Table 11-7 *Default Metrics and Route Metric Types in IGP Route Redistribution*

KEY POINT

IGP into Which Routes Are Redistributed	Default Metric	Default (and Possible) Metric Types
RIP	None	RIP has no concept of external routes
EIGRP	None	External
OSPF	20/1*	E2 (E1 or E2)
IS-IS	0	L1 (L1, L2, L1/L2, or external)

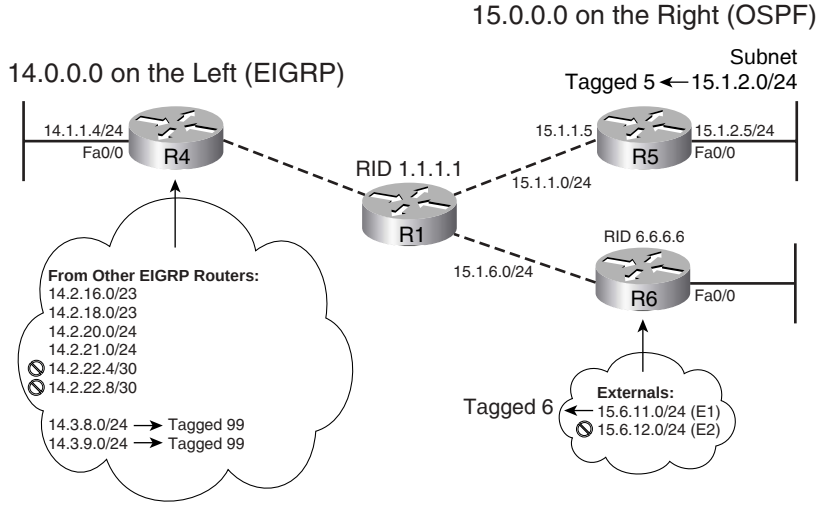
* OSPF uses cost 20 when redistributing from an IGP, and cost 1 when redistributing from BGP.

Redistributing a Subset of Routes Using a Route Map

Route maps can be referenced by any **redistribute** command. The route map may actually let all the routes through, setting different route attributes (for example, metrics) for different routes. Or, it may match some routes with a **deny** clause, which prevents the route from being redistributed. (Refer to Figure 11-1 for a review of route map logic.)

Figure 11-3 and Example 11-2 show an example of mutual redistribution between EIGRP and OSPF, with some routes being either filtered or changed using route maps.

Figure 11-3 *OSPF and EIGRP Mutual Redistribution Using Route Maps*



The following list details the requirements for redistribution from OSPF into EIGRP. These requirements use R1's perspective, because it is the router doing the redistribution.

- Routes with next-hop address 15.1.1.5 (R5) should be redistributed, with route tag 5.

- E1 routes sourced by R6 (RID 6.6.6.6) should be redistributed, and assigned a route tag of 6.
- No other routes should be redistributed.

The requirements for redistributing routes from EIGRP into OSPF are as follows, again from R1's perspective:

- Routes beginning with 14.2, and with masks /23 and /24, should be redistributed, with metric set to 300.
- Other routes beginning with 14.2 should not be redistributed.
- Routes beginning with 14.3 should be redistributed, with route tag 99.
- No other routes should be redistributed.

Most of the explanation of the configuration is provided in the comments in Example 11-2, with a few additional comments following the example.

Example 11-2 *Route Redistribution Using Route Maps*

```
! No metrics are set on the redistribute commands; either the default metric
! is used, or the route maps set the metrics. The default-metric command
! sets the unused EIGRP metric parameters to "1" because something must be
! configured, but the values are unimportant.
router eigrp 1
 redistribute ospf 1 route-map ospf-into-eigrp
 network 14.0.0.0
 default-metric 1544 5 1 1 1
 no auto-summary
! While this configuration strives to use other options besides the options
! directly on the redistribute command, when used by OSPF, you must still
! include the subnets keyword for OSPF to learn subnets from other IGPs.
router ospf 1
 router-id 1.1.1.1
 redistribute eigrp 1 subnets route-map eigrp-into-ospf
 network 15.0.0.0 0.255.255.255 area 0
! ACL A-14-3-x-x matches all addresses that begin 14.3. ACL A-15-1-1-5 matches
! exactly IP address 15.1.1.5. ACL A-6-6-6-6 matches exactly address 6.6.6.6.
ip access-list standard A-14-3-x-x
 permit 14.3.0.0 0.0.255.255
ip access-list standard A-15-1-1-5
 permit 15.1.1.5
ip access-list standard A-6-6-6-6
 permit 6.6.6.6
! The prefix lists matches prefixes in the range 14.2.0.0 through 14.2.255.255,
! with prefix length 23 or 24.
ip prefix-list e-into-o seq 5 permit 14.2.0.0/16 ge 23 le 24
```

Example 11-2 *Route Redistribution Using Route Maps (Continued)*

```
! route-map ospf-into-eigrp was called by the redistribute command under router
! eigrp, meaning that it controls redistribution from OSPF into EIGRP.
! Clause 10 matches OSPF routes whose next hop is 15.1.1.5, which is R5's serial
! IP address. R1's only route that meets this criteria is 15.1.2.0/24. This route
! will be redistributed because the route-map clause 10 has a permit action.
! The route tag is also set to 5.
route-map ospf-into-eigrp permit 10
 match ip next-hop A-15-1-1-5
 set tag 5
! Clause 15 matches OSPF routes whose LSAs are sourced by router with RID 6.6.6.6,
! namely R6, and also have metric type E1. R6 sources two external routes, but
! only 15.6.11.0/24 is E1. The route is tagged 6.
route-map ospf-into-eigrp permit 15
 match ip route-source A-6-6-6-6
 match route-type external type-1
 set tag 6
! route-map eigrp-into-ospf was called by the redistribute command under router
! ospf, meaning that it controls redistribution from EIGRP into OSPF.
! Clause 10 matches using a prefix list, which in turn matches prefixes that begin
! with 14.2, and which have either a /23 or /24 prefix length. By implication, it
! does not match prefix length /30. The metric is set to 300 for these routes.
route-map eigrp-into-ospf permit 10
 match ip address prefix-list e-into-o
 set metric 300
! Clause 18 matches routes that begin 14.3. They are tagged with a 99.
route-map eigrp-into-ospf permit 18
 match ip address A-14-3-x-x
 set tag 99
! Next, the example shows the routes that could be redistributed, and then
! shows the results of the redistribution, pointing out which routes were
! redistributed. First, the example shows, on R1, all routes that R1 could
! try to redistribute into EIGRP.
R1# show ip route 15.0.0.0
Routing entry for 15.0.0.0/24, 5 known subnets
Attached (2 connections)
Redistributing via eigrp 1

O E1    15.6.11.0 [110/84] via 15.1.6.6, 00:21:52, Serial0/0/0.6
O E2    15.6.12.0 [110/20] via 15.1.6.6, 00:21:52, Serial0/0/0.6
C       15.1.6.0 is directly connected, Serial0/0/0.6
O IA    15.1.2.0 [110/65] via 15.1.1.5, 00:21:52, Serial0/0/0.5
C       15.1.1.0 is directly connected, Serial0/0/0.5
```

```
! R4 sees only two of the five routes from 15.0.0.0, because only two matched either of
! the route-map clauses. The other three routes matched the default deny clause.
R4# show ip route 15.0.0.0
Routing entry for 15.0.0.0/24, 2 known subnets
```

continues

Example 11-2 *Route Redistribution Using Route Maps (Continued)*

```
Redistributing via eigrp 1
D EX    15.6.11.0 [170/2171136] via 14.1.1.1, 00:22:21, Serial0/0/0.1
D EX    15.1.2.0 [170/2171136] via 14.1.1.1, 00:22:21, Serial0/0/0.1
! Still on R4, the show ip eigrp topology command displays the tag. This command
! filters the output so that just one line of output lists the tag values.
R4# sho ip eigrp topo 15.6.1.0 255.255.255.0 ¦ incl tag
        Administrator tag is 5 (0x00000005)
R4# sho ip eigrp topo 15.6.11.0 255.255.255.0 ¦ incl tag
        Administrator tag is 6 (0x00000006)
```

```
! Next, the example shows the possible routes that could be redistributed from
! EIGRP into OSPF.
! The next command (R1) lists all routes that could be redistributed into OSPF.
R1# show ip route 14.0.0.0
Routing entry for 14.0.0.0/8, 10 known subnets
  Attached (1 connections)
  Variably subnetted with 3 masks
  Redistributing via eigrp 1, ospf 1

D       14.3.9.0/24 [90/2297856] via 14.1.1.4, 00:34:48, Serial0/0/0.4
D       14.3.8.0/24 [90/2297856] via 14.1.1.4, 00:34:52, Serial0/0/0.4
D       14.1.2.0/24 [90/2172416] via 14.1.1.4, 00:39:27, Serial0/0/0.4
C       14.1.1.0/24 is directly connected, Serial0/0/0.4
D       14.2.22.8/30 [90/2297856] via 14.1.1.4, 00:35:49, Serial0/0/0.4
D       14.2.20.0/24 [90/2297856] via 14.1.1.4, 00:36:12, Serial0/0/0.4
D       14.2.21.0/24 [90/2297856] via 14.1.1.4, 00:36:08, Serial0/0/0.4
D       14.2.16.0/23 [90/2297856] via 14.1.1.4, 00:36:34, Serial0/0/0.4
D       14.2.22.4/30 [90/2297856] via 14.1.1.4, 00:35:53, Serial0/0/0.4
D       14.2.18.0/23 [90/2297856] via 14.1.1.4, 00:36:23, Serial0/0/0.4
```

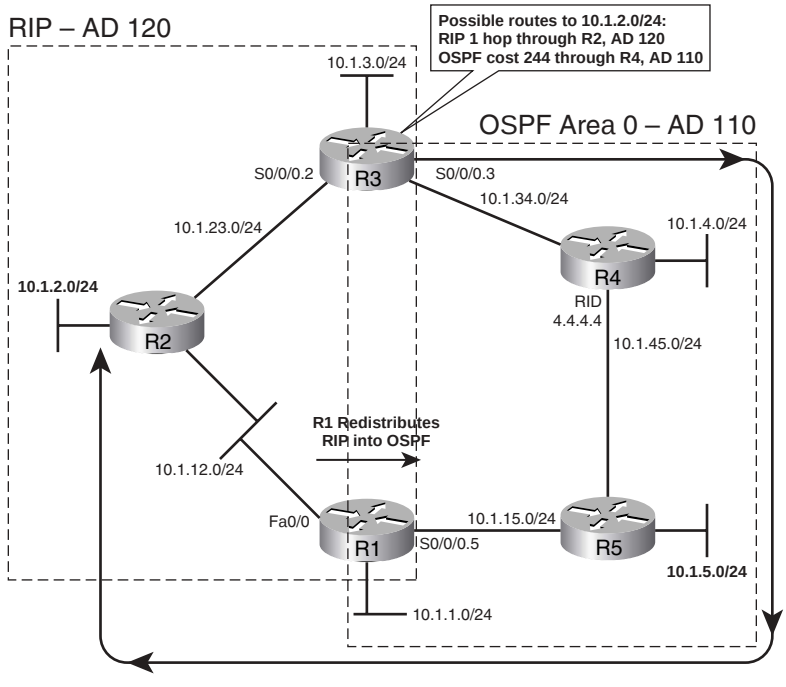
```
! Next, on R5, note that the two /30 routes beginning with 14.2 were correctly
! prevented from getting into OSPF. It also filtered the redistribution of the
! two routes that begin with 14.1. As a result, R5 knows only 6 routes in
! network 14.0.0.0, whereas R1 had 10 subnets of that network it could have
! redistributed. Also below, note that the /23 and /24 routes inside 14.2 have
! metric 300.
R5# show ip route 14.0.0.0
Routing entry for 14.0.0.0/8, 6 known subnets
  Variably subnetted with 2 masks

O E2    14.3.9.0/24 [110/20] via 15.1.1.1, 00:22:41, Serial0/0.1
O E2    14.3.8.0/24 [110/20] via 15.1.1.1, 00:22:41, Serial0/0.1
O E2    14.2.20.0/24 [110/300] via 15.1.1.1, 00:22:41, Serial0/0.1
O E2    14.2.21.0/24 [110/300] via 15.1.1.1, 00:22:41, Serial0/0.1
O E2    14.2.16.0/23 [110/300] via 15.1.1.1, 00:22:41, Serial0/0.1
O E2    14.2.18.0/23 [110/300] via 15.1.1.1, 00:22:41, Serial0/0.1
! The show ip ospf database command confirms that the route tag was set
! correctly.
R5# show ip ospf data external 14.3.8.0 | incl Tag
External Route Tag: 99
```

NOTE Route maps have an implied **deny** clause at the end of the route map. This implied **deny** clause matches all packets. As a result, any routes not matched in the explicitly configured **route-map** clauses match the implied **deny** clause, and are filtered. Both route maps in the example used the implied **deny** clause to actually filter the routes.

Mutual Redistribution at Multiple Routers

When multiple routers redistribute between the same two routing protocol domains, several potential problems can occur. One type of problem occurs on the redistributing routers, because those routers will learn a route to most subnets via both routing protocols. That router uses the AD to determine the best route when comparing the best routes from each of the two routing protocols; this typically results in some routes using suboptimal paths. For example, Figure 11-4 shows a sample network, with R3 choosing its AD 110 OSPF route to 10.1.2.0/24 over the probably better AD 120 RIP route.

Figure 11-4 *OSPF and RIP Redistribution*



NOTE The OSPF configuration for this network matches only the interfaces implied by the OSPF box in Figure 11-4. RIP does not have a *wildcard-mask* option on the **network** command, so R1's and R3's **network** commands will match all of their interfaces, as all are in network 10.0.0.0.

In Figure 11-4, R3 learns of subnet 10.1.2.0/24 via RIP updates from R2. Also, R1 learns of the subnet with RIP and redistributes the route into OSPF, and then R3 learns of a route to 10.1.2.0/24 via OSPF. R3 chooses the route with the lower administrative distance; with all default settings, OSPF's AD of 110 is better that RIP's 120.

If both R1 and R3 mutually redistribute between RIP and OSPF, the suboptimal route problem would occur on either R1 or R3 for each RIP subnet, all depending on timing. Example 11-3 shows the redistribution configuration, along with R3 having the suboptimal route shown in Figure 11-4. However, after R1's fa0/0 interface flaps, R1 now has a suboptimal route to 10.1.2.0/24, but R3 has an optimal route.

Example 11-3 *Suboptimal Routing at Different Redistribution Points*

```
! R1's related configuration follows:
router ospf 1
 router-id 1.1.1.1
 redistribute rip subnets
 network 10.1.15.1 0.0.0.0 area 0
!
router rip
 redistribute ospf 1
 network 10.0.0.0
 default-metric 1
```

```
! R3's related configuration follows:
router ospf 1
 router-id 3.3.3.3
 redistribute rip subnets
 network 10.1.34.3 0.0.0.0 area 0
!
router rip
 redistribute ospf 1
 network 10.0.0.0
 default-metric 1
! R3 begins with an AD 120 OSPF route, and not a RIP route, to 10.1.2.0/24.
R3# sh ip route | incl 10.1.2.0
O E2    10.1.2.0 [110/20] via 10.1.34.4, 00:02:01, Serial0/0/0.4
```

```
! R1 has a RIP route to 10.1.2.0/24, and redistributes it into OSPF, causing R3
! to learn an OSPF route to 10.1.2.0/24.
R1# sh ip route | incl 10.1.2.0
R       10.1.2.0 [120/1] via 10.1.12.2, 00:00:08, FastEthernet0/0
! Next, R1 loses its RIP route to 10.1.2.0/24, causing R3 to lose its OSPF route.
R1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# int fa 0/0
R1(config-if)# shut
```

```
! R3 loses its OSPF route, but can then insert the RIP route into its table.
```

Example 11-3 *Suboptimal Routing at Different Redistribution Points (Continued)*

```
R3# sh ip route | incl 10.1.2.0
R       10.1.2.0 [120/1] via 10.1.23.2, 00:00:12, Serial0/0/0.2
! Not shown: R1 brings up its fa0/0 again
! However, R1 now has the suboptimal route to 10.1.2.0/24, through OSPF.
R1# sh ip route | incl 10.1.2.0
O E2    10.1.2.0 [110/20] via 10.1.15.5, 00:00:09, Serial0/0/0.5
```

The key concept behind this seemingly odd example is that a redistributing router processes only the current contents of its IP routing table. When this network first came up, R1 learned its RIP route to 10.1.2.0/24, and redistributed into OSPF, *before* R3 could do the same. So, R3 was faced with the choice of putting the AD 110 (OSPF) or AD 120 (RIP) route into its routing table, and R3 chose the lower AD OSPF route. Because R3 never had the RIP route to 10.1.2.0/24 in its routing table, R3 could not redistribute that RIP route into OSPF.

Later, when R1's fa0/0 failed (as shown in Example 11-3), R3 had time to remove the OSPF route and add the RIP route for 10.1.2.0/24 to its routing table—which then allowed R3 to redistribute that RIP route into OSPF, causing R1 to have the suboptimal route.

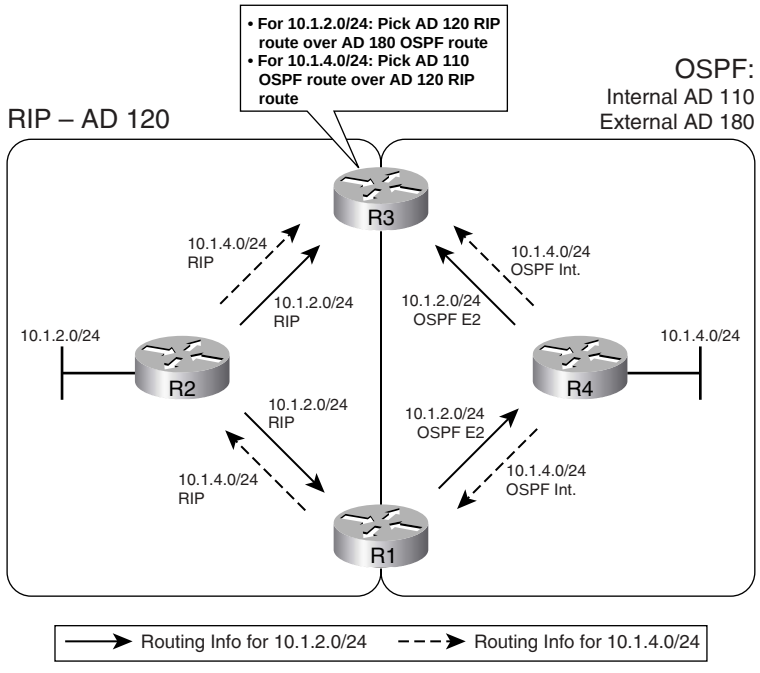
To solve this type of problem, the redistributing routers must have some awareness of which routes came from the other routing domain. In particular, the lower-AD routing protocol needs to decide which routes came from the higher-AD routing protocol, and either use a different AD for those routes or filter the routes. The next few sections show a few different methods of preventing this type of problem.

Preventing Suboptimal Routes by Setting the Administrative Distance

One simple and elegant solution to the problem of suboptimal routes on redistributing routers is to flag the redistributed routes with a higher AD. A route's AD is not advertised by the routing protocol; however, a single router can be configured such that it assigns different AD values to different routes, which then impacts that one router's choice of which routes end up in that router's routing table. For example, back in Figure 11-4 and Example 11-3, R3 could have assigned the OSPF-learned route to 10.1.2.0/24 an AD higher than 120, thereby preventing the original problem.

Figure 11-5 shows a more complete example, with a route from the RIP domain (10.1.2.0/24) and another from the OSPF domain (10.1.4.0/24). Redistributing router R3 will learn the two routes both from RIP and OSPF. By configuring R3's logic to treat OSPF internal routes with default AD 110, and OSPF external routes with AD 180 (or any other value larger than RIP's default of 120), R3 will choose the optimal path for both RIP and OSPF routes.

Figure 11-5 *The Effect of Differing ADs for Internal and External Routes*



Example 11-4 shows how to configure both R1 and R3 to use a different AD for external routes by using the **distance ospf external 180** command, under the **router ospf** process.

Example 11-4 *Preventing Suboptimal Routes with the* **distance** *Router Subcommand*

```
! Both R1's and R3's configurations look like they do in Example 11-3's, but with the
! addition of the distance command.
router ospf 1
 distance ospf external 180
! R3 has a more optimal RIP route to 10.1.2.0/24, as does R1.
R3# sh ip route | incl 10.1.2.0
R       10.1.2.0 [120/1] via 10.1.23.2, 00:00:19, Serial0/0/0.2
```

```
! R1 next…
R1# show ip route | incl 10.1.2.0_
R       10.1.2.0 [120/1] via 10.1.12.2, 00:00:11, FastEthernet0/0
! R1 loses its next-hop interface for the RIP route, so now its OSPF route, with
! AD 180, is its only and best route to 10.1.2.0/24.
R1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# int fa 0/0
R1(config-if)# shut
R1(config-if)# do sh ip route | incl 10.1.2.0
O E2    10.1.2.0 [180/20] via 10.1.15.5, 00:00:05, Serial0/0/0.5
```

EIGRP supports the exact same concept by default, using AD 170 for external routes and 90 for internal routes. In fact, if EIGRP were used instead of OSPF in this example, neither R1 nor R3 would have experienced any of the suboptimal routing. You can reset EIGRP's distance for internal and external routes by using the **distance eigrp** router subcommand. (At presstime, neither the IS-IS nor RIP **distance** commands support setting external route ADs and internal route ADs to different values.)

In some cases, the requirements may not allow for setting all external routes' ADs to another value. For instance, if R4 injected some legitimate external routes into OSPF, the configuration in Example 11-4 would result in either R1 or R3 having a suboptimal route to those external routes that pointed through the RIP domain. In those cases, the **distance** router subcommand can be used in a different way, influencing some or all of the routes that come from a particular router. The syntax is as follows:

```
distance {distance-value ip-address {wildcard-mask} [ip-standard-list] [ip-extended-
  list]
```

This command sets three key pieces of information: the AD to be set, the IP address of the router advertising the routes, and, optionally, an ACL with which to match routes. With RIP, EIGRP, and IS-IS, this command identifies a neighboring router's interface address using the *ip-address wildcard-mask* parameters. With OSPF, those same parameters identify the RID of the router owning (creating) the LSA for the route. The optional ACL then identifies the subset of routes for which the AD will be set. The logic boils down to something like this:

> Set this AD value for all routes, learned from a router that is defined by the IP address and wildcard mask, and for which the ACL permits the route.

Example 11-5 shows how the command could be used to solve the same suboptimal route problem on R1 and R3, while not causing suboptimal routing for other external routes. The design goals are summarized as follows:

- Set a router's local AD for its OSPF routes for subnets in the RIP domain to a value of 179, thereby making the RIP routes to those subnets better than the OSPF routes to those same subnets.
- Do not set the AD for any other routes.

Example 11-5 *Using the* **distance** *Command to Reset Particular Routes' ADs*

```
! R1 config. Note that the command refers to 3.3.3.3, which is R3's RID. Other
! commands not related to resetting the AD are omitted. Of particular importance,
! the distance command on R1 refers to R3's OSPF RID, because R3 created the OSPF
! LSAs that we are trying to match—the LSAs created when R3 injected the
! routes redistributed from RIP.
router ospf 1
 distance 179 3.3.3.3 0.0.0.0 only-rip-routes
```

continues

Example 11-5 *Using the* **distance** *Command to Reset Particular Routes' ADs (Continued)*

```
!
ip access-list standard only-rip-routes
 permit 10.1.12.0
 permit 10.1.3.0
 permit 10.1.2.0
 permit 10.1.23.0
```

```
! R3 config. Note that the command refers to 1.1.1.1, which is R1's RID. Other
! commands not related to resetting the AD are omitted. Also, the only-rip-routes
! ACL is identical to R1's only-rip-routes ACL.
router ospf 1
distance 179 1.1.1.1 0.0.0.0 only-rip-routes
```

Preventing Suboptimal Routes by Using Route Tags

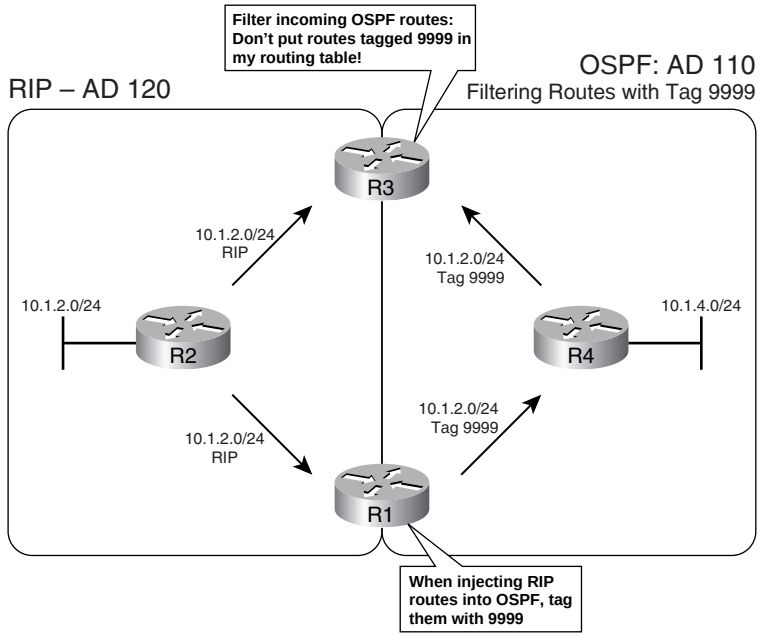
Another method of preventing suboptimal routing on the redistributing routers is to simply filter the problematic routes. Using subnet 10.1.2.0/24 as an example again, R3 could use an incoming **distribute-list** command to filter the OSPF route to 10.1.2.0/24, allowing R3 to use its RIP route to 10.1.2.0/24. R1 would need to perform similar route filtering as well to prevent its suboptimal route.

Performing simple route filtering based on IP subnet number works, but the redistributing routers will need to be reconfigured every time subnets change in the higher-AD routing domain. The administrative effort can be improved by adding *route tagging* to the process. By tagging all routes taken from the higher-AD domain and advertised into the lower-AD domain, the **distribute-list** command can make a simple check for that tag. Figure 11-6 shows the use of this idea for subnet 10.1.2.0/24.

Route tags are simply unitless integer values in the data structure of a route. These tags, typically either 16 or 32 bits long depending on the routing protocol, allow a router to imply something about a route that was redistributed from another routing protocol. For instance, R1 can tag its OSPF-advertised route to 10.1.2.0/24 with a tag—say, 9999. OSPF does not define what a tag of 9999 means, but the OSPF protocol includes the tag field in the LSA so that it can be used for administrative purposes. Later, R3 can filter routes based on their tag, solving the suboptimal route problem.

Figure 11-6 and Example 11-6 depict an example of route tagging and route filtering, used to solve the same old problem with suboptimal routes. R1 and R3 tag all redistributed RIP routes with tag 9999 as they enter the OSPF domain, and then R1 and R3 filter incoming OSPF routes based on the tags. This design works well because R1 can tag all redistributed RIP routes, thereby removing the need to change the configuration every time a new subnet is added to the RIP domain. (Note that both R1 and R3 will tag routes injected from RIP into OSPF as 9999, and both will then filter OSPF-learned routes with tag 9999. Figure 11-6 just shows one direction to keep the figure less cluttered.)

Figure 11-6 *Filtering with Reliance on Route Tags*



Example 11-6 *Using Route Tags and Distribute Lists to Prevent Suboptimal Routes at Redistributing Routers*

```
! R1 config. The redistribute command calls the route map that tags routes taken
! from RIP as 9999. distribute-list looks at routes learned in OSPF that were
! earlier tagged by R3.
router ospf 1
 redistribute rip subnets route-map tag-rip-9999
 network 10.1.15.1 0.0.0.0 area 0
 distribute-list route-map check-tag-9999 in
! Clause 10, a deny clause, matches all tagged 9999 routes—so those
! routes are filtered. Clause 20 permits all other routes, because with no match
! subcommand, the clause is considered to "match all."
route-map check-tag-9999 deny 10
 match tag 9999
!
route-map check-tag-9999 permit 20
! tag-rip-9999 matches all routes (it has no match command), and then
! tags them all with tag 9999. This route-map is used only for routes taken from
! RIP into OSPF.
route-map tag-rip-9999 permit 10
 set tag 9999
```

```
! R3 Config
! The R3 configuration does not have to use the same names for route maps, but
```

continues

Example 11-6 *Using Route Tags and Distribute Lists to Prevent Suboptimal Routes at Redistributing Routers (Continued)*

```
! the essential elements are identical, so the route maps are not repeated here.
router ospf 1
 redistribute rip subnets route-map tag-rip-9999
 network 10.1.34.3 0.0.0.0 area 0
 distribute-list route-map check-tag-9999 in
! R3 (shown) and R1 have RIP routes to 10.1.2.0, as well as other routes from the
! RIP domain. Also, note that the OSPF LSDB shows the tagged values on the routes.
R3# show ip route | incl 10.1.2.0
R       10.1.2.0 [120/1] via 10.1.23.2, 00:00:26, Serial0/0/0.2
R3# sh ip ospf data  begin Type-5
                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
10.1.1.0        1.1.1.1         834         0x80000006 0x00CE86 9999
10.1.1.0        3.3.3.3         458         0x80000003 0x0098B7 9999
10.1.2.0        1.1.1.1         834         0x80000006 0x00C390 9999
10.1.2.0        3.3.3.3         458         0x80000003 0x008DC1 9999
! lines omitted for brevity
! Next, the unfortunate side effect of filtering the routes—R3 does not have an
! alternative route to RIP subnets, although OSPF internal routers (like R4
! in Figure 11-6) will.
R3# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R3(config)# int s0/0/0.2
R3(config-subif)# shut
R3(config-subif)# ^Z
R3# sh ip route | incl 10.1.2.0
R3#
```

The last few lines of the example show the largest negative of using route filtering to prevent the suboptimal routes. When R3 loses connectivity to R2, R3 does not use the alternate route through the OSPF domain. R3's filtering of those routes occurs regardless of whether R3's RIP routes are available or not. As a result, using a solution that manipulates the AD may ultimately be the better solution to this suboptimal-routing problem.

Using Metrics and Metric Types to Influence Redistributed Routes

A different set of issues can occur for a router that is internal to a single routing domain, like R4 and R5 in Figure 11-4. The issue is simple—with multiple redistributing routers, an internal router learns multiple routes to the same subnet, so it must pick the best route. As covered earlier in the chapter, the redistributing routers can set the metrics; by setting those metrics with meaningful values, the internal routers can be influenced to use a particular redistribution point.

Interestingly, internal routers may not use metric as their first consideration when choosing the best route. For instance, an OSPF internal router will first take an intra-area route over an inter-area route, regardless of their metrics. Table 11-8 lists the criteria an internal router will use when picking the best route, before considering the metrics of the different routes.

Table 11-8 *IGP Order of Precedence for Choosing Routes Before Considering the Metric*

IGP	Order of Precedence of Metric
RIP	No other considerations
EIGRP	Internal, then external
OSPF	Intra-area, inter-area, E1, then E2*
IS-IS	L1, L2, external

* For E2 routes whose metric ties, OSPF also checks the cost to the advertising ASBR.

To illustrate some of these details, Example 11-7 focuses on R4 and its routes to 10.1.2.0/24 and 10.1.5.0/24 from Figure 11-4. The example shows the following, in order:

1. R1 and R3 advertise 10.1.2.0/24 as an E2 route, metric 20. R4 uses the route through R3, because R4's cost to reach ASBR R3 is lower than its cost to reach ASBR R1.
2. After changing R1 to advertise redistributed routes into OSPF as E1 routes, R4 uses the E1 routes through R1, even though the metric is larger than the E2 route through R3.
3. R4 uses it higher-metric intra-area route to 10.1.5.0/24 through R5. Then, the R4-R5 link fails, causing R4 to use the OSPF external E2 route to 10.1.5.0/24—the route that leads through the RIP domain and back into OSPF via the R3-R2-R1-R5 path.

Example 11-7 *Demonstration of the Other Decision Criteria for Choosing the Best Routes*

```
! R4 has E2 routes to all the subnets in the RIP domain, and they all point to R3.
R4# sh ip route ospf
10.0.0.0/24 is subnetted, 10 subnets
O       10.1.15.0 [110/128] via 10.1.45.5, 00:03:23, Serial0/0/0.5
O E2    10.1.12.0 [110/20] via 10.1.34.3, 00:03:23, Serial0/0/0.3
O E2    10.1.3.0 [110/20] via 10.1.34.3, 00:03:23, Serial0/0/0.3
O E2    10.1.2.0 [110/20] via 10.1.34.3, 00:03:23, Serial0/0/0.3
O E2    10.1.1.0 [110/20] via 10.1.34.3, 00:03:23, Serial0/0/0.3
O       10.1.5.0 [110/65] via 10.1.45.5, 00:03:23, Serial0/0/0.5
O E2    10.1.23.0 [110/20] via 10.1.34.3, 00:03:23, Serial0/0/0.3
! R4 chose the routes through R3 instead of R1 due to the lower cost to R3.
R4# show ip ospf border-routers
OSPF Process 1 internal Routing Table
Codes: i - Intra-area route, I - Inter-area route

```

continues

Example 11-7 *Demonstration of the Other Decision Criteria for Choosing the Best Routes (Continued)*

```
i 1.1.1.1 [128] via 10.1.45.5, Serial0/0/0.5, ASBR, Area 0, SPF 13
i 3.3.3.3 [64] via 10.1.34.3, Serial0/0/0.3, ASBR, Area 0, SPF 13
! (Not Shown): R1 is changed to redistribute RIP routes as E1 routes by
! adding the metric-type 1 option on the redistribute command on R1.
! R4 picks routes through R1 because they are E1 routes, even though the metric
! (148) is higher than the routes through R3 (cost 20)
R4# show ip route ospf
10.0.0.0/24 is subnetted, 10 subnets
O E1    10.1.2.0 [110/148] via 10.1.45.5, 00:00:11, Serial0/0/0.5
! lines omitted for brevity
! R4's route to 10.1.5.0/24 below is intra-area, metric 65
R4# show ip route | incl 10.1.5.0
O       10.1.5.0 [110/65] via 10.1.45.5, 00:04:48, Serial0/0/0.5
! (Not Shown): R4 shuts down link to R5
! R4's new route to 10.1.5.0/24 is E2, learned from R3, with metric 20
R4# show ip route | incl 10.1.5.0\
O E2    10.1.5.0 [110/20] via 10.1.34.3, 00:10:52, Serial0/0/0.3
```

Route Summarization

Route summarization creates a single route whose numeric range, as implied by the prefix/prefix length, is larger than the one or more smaller component routes. For example, 10.1.0.0/16 is a summary route that includes component subnets 10.1.1.0/24, 10.1.4.132/30, and any other subnets with the range 10.1.0.0 through 10.1.255.255.

NOTE I use the term *component route* to refer to a route whose range of IP addresses is a subset of the range specified by a summary route; however, I have not seen this term in other reference materials from Cisco.

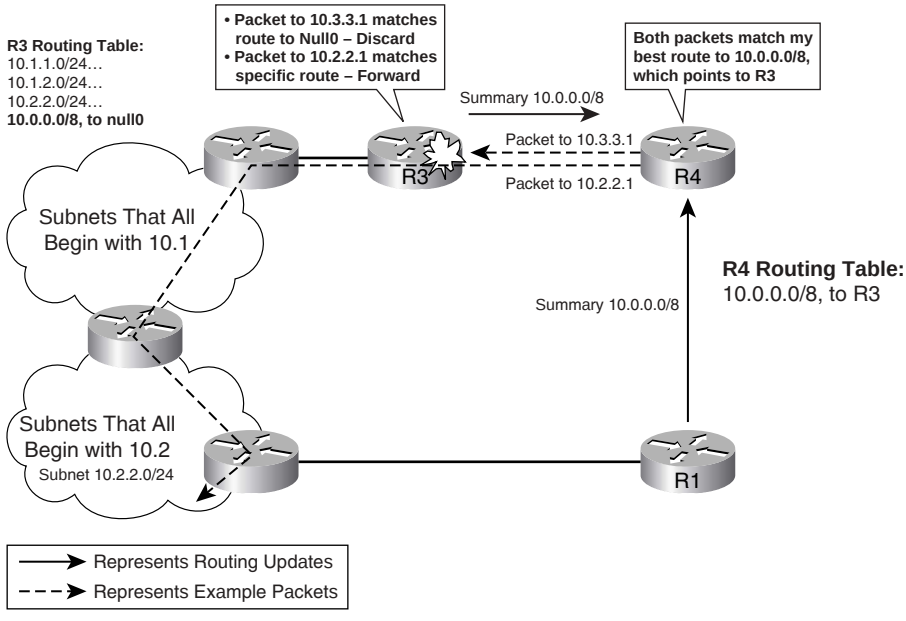
The following list details some of the key features that the three IGPs covered in this book have in common with regard to how route summarization works (by default):

- The advertised summary is assigned the same metric as the currently lowest-metric component subnet.
- The router does not advertise the component subnets.
- The router does not advertise the summary when its routing table does not have any of the component subnets.
- The summarizing router creates a local route to the summary, with destination null0, to prevent routing loops.

- Summary routes reduce the size of routing tables and topology databases, indirectly improving convergence.
- Summary routes decrease the amount of specific information in routing tables, sometimes causing suboptimal routing.

Figure 11-7 depicts the suboptimal-routing side effect when using route summarization. It also depicts the effect of using a summary to null0 on the summarizing router.

Figure 11-7 *Route Summarization Suboptimal Routing and Routing to Null0*



In Figure 11-7, R4 learned two paths to summary route 10.0.0.0/8, and picked the route through R3 based on the metric. Because R4 does not have a route for 10.2.2.0/24, R4 then sends any packets to that subnet based on its route to network 10.0.0.0/8, through R3. So, although subnets like 10.2.2.0/24 may be topologically closer to R4 through R1, R4 sends the packets via the scenic, suboptimal route through R3.

Also note that R4's summary route to 10.0.0.0/8 matches packets for which the component subnet does not exist anywhere in the network. In that case, routers like R4 forward the packets based on the larger summary, but once the packet reaches the router that created the summary, the packet is discarded by the summarizing router due to its null route. For instance, Figure 11-7 shows R4 forwarding a packet destined to 10.3.3.1 to R3. R3 does not have a more specific route than its route to 10.0.0.0/8, with next-hop interface null0. As a result, R3 discards the packet.

The sections that follow provide a few details about summarization with each routing protocol.

EIGRP Route Summarization

EIGRP provides the easiest and most straightforward rules for summarizing routes as compared with RIPv2, OSPF, and IS-IS. To summarize routes, the **ip summary-address eigrp** *as-number network-address subnet-mask* [*admin-distance*] command is placed under an interface. If any of the component routes are in that router's routing table, EIGRP advertises the summary route *out* that interface. The summary is defined by the *network-address subnet-mask* parameters.

One of the more interesting features of the EIGRP summary is the ability to set the AD of the summary route. The AD is not advertised with the route; the summarizing router, however, uses the configured AD to determine whether the null route for the summary should be put into its routing table. The EIGRP AD for summary routes defaults to 5.

OSPF Route Summarization

All OSPF routers in the same area must have identical LSDBs after flooding is complete. As a result, all routers in the same OSPF area must have the same summary routes, and must be missing the same component subnets of each summary. To make that happen, OSPF allows route summarization only as routes are injected into an area, either by an ABR (inter-area routes) or by an ASBR (external routes).

OSPF uses two different configuration commands to create the summary routes, depending on whether the summary is for inter-area or external routes. Table 11-9 lists the two commands. Both commands are configured under **router ospf**.

Table 11-9 *OSPF Route Summarization Commands*

KEY POINT

Where used	Command
ASBR	**summary-address** {{*ip-address mask*} | {*prefix mask*}} [**not-advertise**] [**tag** *tag*]
ABR	**area** *area-id* **range** *ip-address mask* [**advertise** | **not-advertise**] [**cost** *cost*]

The commands have a couple of important attributes. First, the **area range** command specifies an area; this area is the area in which the component subnets reside, with the summary being advertised into *all other areas*. Also, the **area range** command can set the cost for the summary route, instead of using the lowest cost of all component routes. Also, the **not-advertise** keyword can essentially be used to filter the subnets implied by the summary, as covered in Chapter 10, "OSPF."

The **summary-address** command summarizes external routes as they are injected into OSPF as an ASBR. The cost can be assigned, and the routes can be filtered using the **not-advertise** keyword.

RIP Route Summarization

RIP route summarization is weird in comparison to route summarization in the other IGPs, but at first glance, it appears to work just like EIGRP. To summarize routes, RIP uses the interface subcommand **ip summary-address rip** *ip-address ip-network-mask*. It can be used on any interface out which RIP advertises routes. If any of the component routes are in that router's routing table, RIP advertises the summary route out the interface, as defined by the *ip-address ip-network-mask* parameters. It also ceases advertising the component routes out that interface.

So far its sounds just like EIGRP summarization; however, there are a couple of unique restrictions. First, RIP route summarization works only with RIPv2, because RIPv1 does not support VLSM. Also, RIP does not allow supernetting—for instance, the command **ip summary-address rip 172.16.0.0 255.254.0.0**, which would combine two class B networks into one summary, is not supported by RIP. Finally, on a single interface, only one **ip summary-address rip** command is allowed per classful network. In other words, RIP would not allow a router to create two summary routes, one for 10.1.0.0/16 and one for 10.2.0.0/16, and advertise both out the same interface. EIGRP has none of these restrictions.

Default Routes

Routers forward packets using a default route when there are no specific routes that match a packet's destination IP address in the IP routing table. Routing protocols can advertise default routes, with each router choosing the best default route to list as that router's *gateway of last resort*. This section covers how a router can create a default route and then cause an IGP to advertise the default route.

In addition to the advertisement of default routes, each router may use one of two options for how the default route is used. As described in Chapter 7, "IP Forwarding (Routing)," each router's configuration includes either the (default) **ip classless** command or the **no ip classless** command. With **ip classless**, if a packet's destination does not match a specific route in the IP routing table, the router uses the default route. With **no ip classless**, the router first checks to see if any part of the destination address's classful network is in the routing table. If so, that router will not use the default route for forwarding that packet.

NOTE The topic of default routing requires discussion of the configuration on one router, plus configuration of the other routers using the same IGP. For this section, I will call the router with the default routing configuration the "local" router, and other routers using the same IGP "other" routers.

Cisco IOS supports five basic methods of advertising default routes with IGPs, four of which are covered here. One method for advertising a default route is for one routing protocol to redistribute another routing protocol's default route. Because route redistribution has already been covered

heavily, this section of the chapter covers other methods. Of the other four methods, not all are supported by all IGPs, as you can see in Table 11-10.

Table 11-10 *Four Methods for Learning Default Routes*

KEY POINT

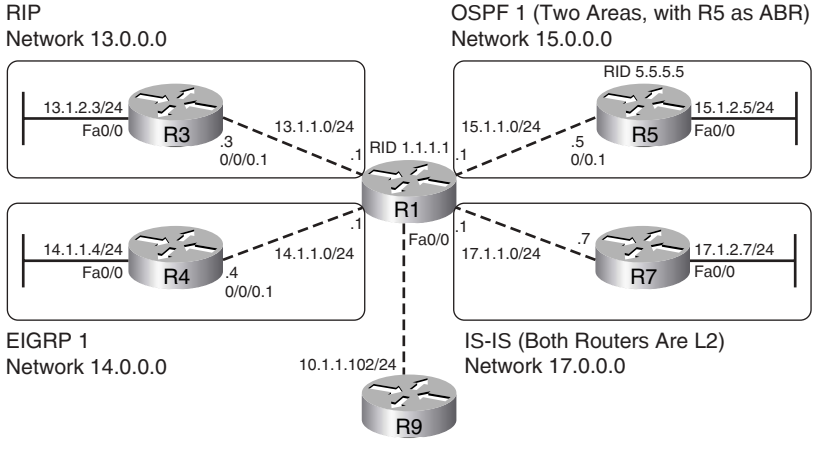
Feature	RIP	EIGRP	OSPF
Static route to 0.0.0.0, with the **redistribute static** command	Yes	Yes	No
The **default-information originate** command	Yes	No	Yes
The **ip default-network** command	Yes	Yes	No
Using summary routes	No	Yes	No

Interestingly, when a router learns of multiple default routes, using any of these methods, it will use the usual process for choosing the best route: administrative distance, route type (per Table 11-9, earlier in this chapter), and lowest metric, in that order.

NOTE Table 11-10 has details that may be difficult to memorize. To make it easier, you could start by ignoring the use of summary static routes, because it is not recommended by Cisco. Then, note that RIP supports the other three methods, whereas EIGRP supports two methods and OSPF supports only one—with EIGRP and OSPF not supporting any of the same options.

Figure 11-8 shows a sample network used with all the default route examples, in which R1 is the local router that configures the default routing commands.

Figure 11-8 *Sample Network for Default Route Examples*



Using Static Routes to 0.0.0.0, with redistribute static

Routers consider a route to 0.0.0.0/0 as a default route. RIP and EIGRP support redistribution of static routes, including such a default static route. The rules and conditions for redistributing static defaults into RIP and EIGRP are as follows:

KEY POINT

- The static **ip route 0.0.0.0 0.0.0.0** and **redistribute static** commands need to be configured on the same local router.
- The metric must be defaulted or set, using the same methods covered earlier in this chapter.
- The **redistribute** command can refer to a route map, which examines all static routes (not just the default).
- EIGRP treats the default route as an external route by default, with default AD 170.
- This method is not supported by OSPF.

Example 11-8 shows how R1 can inject defaults via RIP to R3 and via EIGRP to R4. The EIGRP configuration refers to a route map that examines all static routes, matching only static default routes. If other static routes existed, EIGRP would not advertise those routes based on the route map.

Example 11-8 *Static Default Route with Route Redistribution*

```
! R1 Config—note that ip classless is configured, but it does not impact the
! advertisement of the static route at all.
router eigrp 1
 redistribute static route-map just-default
 network 10.0.0.0
 network 14.0.0.0
 default-metric 1544 10 1 1 1
!
router rip
 version 2
 redistribute static
 network 13.0.0.0
 default-metric 1
!
ip classless
! The static route is configured next, followed by the prefix list that matches
! the default route, and the route map that refers to the prefix list.
ip route 0.0.0.0 0.0.0.0 10.1.1.102
!
ip prefix-list zero-prefix seq 5 permit 0.0.0.0/0
!
route-map just-default permit 10
 match ip address prefix-list zero-prefix
!
```

continues

Example 11-8 *Static Default Route with Route Redistribution (Continued)*

```
route-map just-default deny 20
```

```
! Next, R3, the RIP router, lists R1 (13.1.1.1) as its gateway of last resort,
! based on the RIP route to 0.0.0.0/0, next hop 13.1.1.1.
R3# sh ip route
! Lines omitted for brevity
Gateway of last resort is 13.1.1.1 to network 0.0.0.0

     13.0.0.0/24 is subnetted, 2 subnets
C       13.1.1.0 is directly connected, Serial0/0/0.1
C       13.1.2.0 is directly connected, FastEthernet0/0
R*   0.0.0.0/0 [120/1] via 13.1.1.1, 00:00:12, Serial0/0/0.1
```

```
! Next, R4, the EIGRP router, lists R1 (14.1.1.1) as its gateway of last resort,
! based on the EIGRP route to 0.0.0.0/0, next hop 14.1.1.1. Note that the default
! points to 0.0.0.0/0, AD 170, as it is an external route, due to the EX listed
! in the output of the show ip route command.
R4# sh ip route
! lines omitted for brevity
Gateway of last resort is 14.1.1.1 to network 0.0.0.0

D    10.0.0.0/8 [90/2172416] via 14.1.1.1, 00:01:30, Serial0/0/0.1
     14.0.0.0/24 is subnetted, 2 subnets
C       14.1.2.0 is directly connected, FastEthernet0/0
C       14.1.1.0 is directly connected, Serial0/0/0.1
D*EX 0.0.0.0/0 [170/2172416] via 14.1.1.1, 00:01:30, Serial0/0/0.1
```

Using the default-information originate Command

OSPF does not support redistribution of statically defined default routes. Instead, OSPF requires the **default-information originate** router subcommand, which essentially tells OSPF to redistribute any default routes found in the routing table, either static routes or routes from another routing protocol. The following list summarizes the default routing features when using the **default-information originate** command with OSPF:

KEY POINT

- Redistributes any default route (0.0.0.0/0) in the routing table.
- The command can set the metric and metric type directly, with OSPF defaulting to cost 1 and type E2.
- OSPF allows the use of the **always** keyword, which means a default is sourced regardless of whether a default route is in the routing table.
- Not supported by EIGRP.
- Supported by RIP, with some differences. (Refer to the text following Example 11-9 for an explanation of the differences.)

Example 11-9 shows an example of using the **default-information originate** command with OSPF. In this case, R1 has learned a route to 0.0.0.0/0 via BGP from R9 in Figure 11-8.

Example 11-9 *Static Default Route with Route Redistribution*

```
router ospf 1
 network 15.0.0.0 0.255.255.255 area 0
 default-information originate
```

```
! R5 has a default route, defaulting to type E2, cost 1. It as advertised as a
! type 5 LSA.
R5# show ip route ospf
O*E2 0.0.0.0/0 [110/1] via 15.1.1.1, 00:18:07, Serial0/0.1
R5# sh ip ospf data | begin Type-5
                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
0.0.0.0         1.1.1.1         1257        0x80000001 0x008C12 1
```

As mentioned earlier, RIP does support the **default-information originate** command; however, the command behaves slightly differently in RIP than it does in OSPF. With RIP, this command creates and advertises a default route if either no default route exists or a default route was learned from another routing protocol. However, if a static route to 0.0.0.0/0 is in the local routing table, the **default-information originate** command does *not* cause RIP to inject a default—the reason behind this behavior is that RIP already supports redistribution of static routes, so **redistribute static** should be used in that case.

Using the ip default-network Command

RIP and EIGRP can inject default routes by using the **ip default-network** command. To do so, the following must be true on the local router:

- The local router must configure the **ip default-network** *net-number* command, with *net-number* being a classful network number.
- The classful network must be in the local router's IP routing table, via any means.
- For EIGRP only, the classful network must be advertised by the local router into EIGRP, again through any means.
- This method is not supported by OSPF.

When using the **ip default-network** command, RIP and EIGRP differ in how they advertise the default. RIP advertises a route to 0.0.0.0/0, but EIGRP flags its route to the classful network as a candidate default route. Because EIGRP flags these routes as candidates, EIGRP must then also be advertising those classful networks. However, because RIP does not flag the classful network as a candidate default route, RIP does not actually have to advertise the classful network referenced in the **ip default-network** command.

Example 11-10 shows the key difference between RIP and EIGRP with regard to the **ip default-network** command. In this case, R1 will advertise about classful network 10.0.0.0 using EIGRP due to the **auto-summary** command.

Example 11-10 *Static Default Route with Route Redistribution*

```
! EIGRP will advertise classful network 10.0.0.0/8 due to its network command,
! matching R1's fa0/0 interface, and the auto-summary command. Also, R1 must have
! a route to classful network 10.0.0.0/8, in this case due to a static route.
! RIP will not advertise classful network 10.0.0.0/8, but it will still be able
! to inject a default route based on the ip default-network command.
router eigrp 1
 network 10.0.0.0
 network 14.0.0.0
 auto-summary
!
router rip
 version 2
 network 13.0.0.0
!
ip classless
ip default-network 10.0.0.0
ip route 10.0.0.0 255.0.0.0 10.1.1.102
```

```
! On R3, RIP learns a route to 0.0.0.0/0 as its default.
R3# show ip route rip
R*   0.0.0.0/0 [120/1] via 13.1.1.1, 00:00:19, Serial0/0/0.1
```

```
! On R4, note that EIGRP learned a route to 10.0.0.0/8, shown with a * that
! flags the route as a candidate default route.
R4# show ip route
! lines omitted for brevity
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 14.1.1.1 to network 10.0.0.0

D*   10.0.0.0/8 [90/2172416] via 14.1.1.1, 00:05:35, Serial0/0/0.1
     14.0.0.0/24 is subnetted, 2 subnets
C       14.1.2.0 is directly connected, FastEthernet0/0
C       14.1.1.0 is directly connected, Serial0/0/0.1
```

Using Route Summarization to Create Default Routes

Generally speaking, route summarization combines smaller address ranges into a small number of larger address ranges. From that perspective, 0.0.0.0/0 is the largest possible summary, because it includes all possible IPv4 addresses. And, as it turns out, EIGRP route summarization supports summarizing the 0.0.0.0/0 supernet, effectively creating a default route.

Because route summarization causes a null route to be created for the summary, some Cisco documentation advises against using route summarization to create a default route. For example,

in Figure 11-8, imagine that R9 is owned by this network's ISP, and R1 learns a default route (0.0.0.0/0) via EBGP from R9. However, when R1 configures an EIGRP default route using route summarization, R1 will also create a local route to 0.0.0.0/0 as well, but with destination null0. The EBGP route has a higher AD (20) than the EIGRP summary route to null0 (AD 5), so R1 will now replace its BGP-learned default route with the summary route to null0—preventing R1 from being able to send packets to the Internet.

Route summarization can still be used to create default routes with the proper precautions. The following list details a few of the requirements and options:

- The local router creates a local summary route, destination null0, using AD 5 (EIGRP), when deciding if its route is the best one to add to the local routing table.
- EIGRP advertises the summary to other routers as AD 90 (internal).
- This method is not supported by RIP and OSPF.
- To overcome the caveat of EIGRP's default route being set to null by having a low AD, set the AD higher (as needed) with the **ip summary-address** command.

Example 11-11 lists a sample configuration on R1 again, this time creating summary routes to 0.0.0.0/0 for EIGRP.

Example 11-11 *EIGRP and IS-IS Configuration for Creating Default Summary Routes*

```
! EIGRP route summarization is done under s0/0/0.4, the subnet connected to R4. In this
! example, the AD was changed to 7 (default 5) just to show how to change the AD. To
! avoid the problem with the default route to null0 on R1, the AD should have been set
! higher than the default learned via BGP.
interface Serial0/0/0.4 point-to-point
 ip address 14.1.1.1 255.255.255.0
 ip summary-address eigrp 1 0.0.0.0 0.0.0.0 7
! In this example, R1 has two sources for a local route to 0.0.0.0/0: EIGRP
! (AD 7, per the ip summary-address command), and BGP from R9
! (AD 20). R1 installs the EIGRP route based on the lowest AD.
R1# show ip route eigrp
      14.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
D       14.1.2.0/24 [90/2172416] via 14.1.1.4, 00:01:03, Serial0/0/0.4
D       14.0.0.0/8 is a summary, 05:53:19, Null0
D*   0.0.0.0/0 is a summary, 00:01:08, Null0
```

```
! Next, R4's EIGRP route shows AD 90, instead of the AD 7 configured at R1. AD is
! a local parameter—R4 uses its default AD of 90 for internal routes.
R4# show ip route eigrp
D*   0.0.0.0/0 [90/2172416] via 14.1.1.1, 00:01:14, Serial0/0/0.1
```

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most of the Cisco Press *Exam Certification Guides*, this book does not repeat information presented in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with Key Point icons.

Table 11-11 lists some of the most relevant Cisco IOS commands related to the topics in this chapter. Also refer to Tables 11-2 and 11-3 for the **match** and **set** commands.

Table 11-11 *Command Reference for Chapter 11*

Command	Command Mode and Description
redistribute *protocol* [*process-id*] {**level-1** | **level-1-2** | **level-2**} [*as-number*] [**metric** *metric-value*] [**metric-type** *type-value*] [**match** {**internal** | **external 1** | **external 2**}] [**tag** *tag-value*] [**route-map** *map-tag*] [**subnets**]	Router config mode; defines the routing protocol from which to take routes, several matching parameters, and several things that can be marked on the redistributed routes.
ip prefix-list *list-name* [**seq** *seq-value*] {**deny** *network/length* | **permit** *network/length*} [**ge** *ge-value*] [**le** *le-value*]	Global config mode; defines members of a prefix list, which match a prefix (subnet) and prefix length (subnet mask).
ip prefix-list *list-name sequence-number* **description** *text*	Global config; sets a description to a line in a prefix list.
distance {*ip-address* {*wildcard-mask*}} [*ip-standard-list*] [*ip-extended-list*]	Router config mode; identifies the route source, and an optional ACL to define a subnet of routes, for which this router's AD is changed. Influences the selection of routes by selectively overriding default AD.
distance eigrp *internal-distance external-distance*	EIGRP config; sets the AD for all internal and external routes.
distance ospf {[**intra-area** *dist1*] [**inter-area** *dist2*] [**external** *dist3*]}	OSPF config; sets the AD for all intra-area, interarea, and external routes.
ip summary-address eigrp *as-number network-address subnet-mask* [*admin-distance*]	Interface mode; configures an EIGRP route summary.
ip summary-address rip *ip-address ip-network-mask*	Interface mode; configures a RIP route summary.
area *area-id* **range** *ip-address mask* [**advertise** | **not-advertise**] [**cost** *cost*]	OSPF mode; configures an OSPF summary between areas.
summary-address *address mask* {**level-1** | **level-1-2** | **level-2**}	IS-IS mode; configures an IP summary route.

Table 11-11 *Command Reference for Chapter 11 (Continued)*

Command	Command Mode and Description
summary-address {{*ip-address mask*} | {*prefix mask*}} [**not-advertise**] [**tag** *tag*]	OSPF mode; configures an OSPF summary of external routes.
ip default-network *network-number*	Global config; sets a network from which to derive default routes.
default-information originate [**route-map** *map-name*]	IS-IS config; tells IS-IS to advertise a default route if it is in the routing table.
default-information originate [**always**] [**metric** *metric-value*] [**metric-type** *type-value*] [**route-map** *map-name*]	OSPF config; tells OSPF to advertise a default route, either if it is in the routing table or always.
ip route *prefix mask* {*ip-address* | *interface-type interface-number* [*ip-address*]} [*distance*] [*name*] [**permanent**] [**tag** *tag*]	Global config; used to create static IP routes, including static routes to 0.0.0.0 0.0.0.0, which denotes a default route.

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" sections tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

default route, route redistribution, external route, aggregate route, route map, IP prefix list, summary route, component route, gateway of last resort

Refer to the CD-based glossary to check your answers.

Further Reading

Routing TCP/IP, Volume I, Second Edition, by Jeff Doyle and Jennifer DeHaven Carroll

CCIE Practical Studies, Volume II, by Karl Solie and Leah Lynch

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- IP Routing
  - — BGP
  - — The use of **show** and **debug** commands

CHAPTER 12

Fundamental BGP Operations

Chapters 12 and 13 of this book cover what might be the single most important topic on both the CCIE Routing and Switching written and lab exams—Border Gateway Protocol (BGP) Version 4. This chapter focuses on how BGP accomplishes its fundamental tasks:

1. Forming neighbor relationships
2. Injecting routes into BGP from some other source
3. Exchanging those routes with other routers
4. Placing routes into IP routing tables

All of these BGP topics have close analogies with those of BGP's IGP cousins, but of course there are many differences in the details.

While this chapter focuses on how BGP performs its central role as a routing protocol, Chapter 13, "BGP Routing Policies," moves on to cover routing policies—the methods used with BGP to limit or change how BGP filters routes and chooses the best routes among multiple routes to the same prefix.

"Do I Know This Already?" Quiz

Table 12-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 12-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Building BGP Neighbor Relationships	1–3	
Building the BGP Table	4–8	
Building the IP Routing Table	9–12	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. Into which of the following neighbor states must a neighbor stabilize before BGP Update messages may be sent?
   a. Active
   b. Idle
   c. Connected
   d. Established

2. BGP neighbors check several parameters before the neighbor relationship can be completed. Which of the following is not checked?
   a. That the neighbor's router ID is not duplicated with other routers
   b. That the **neighbor** command on one router matches the update source IP address on the other router
   c. If eBGP, that the **neighbor** command points to an IP address in a connected network
   d. That a router's **neighbor remote-as** command refers to the same autonomous system number (ASN) as in the other router's **router bgp** command (assuming confederations are not used)

3. A group of BGP routers, some with iBGP and some with eBGP connections, all use loopback IP addresses to refer to each other in their **neighbor** commands. Which of the following statements are false regarding the configuration of these peers?
   a. IBGP peers require a **neighbor** *ip-address* **ibgp-multihop** command for the peer to become established.
   b. eBGP peers require a **neighbor** *ip-address* **ebgp-multihop** command for the peer to become established.
   c. eBGP and iBGP peers cannot be placed into the same peer group.
   d. For eBGP peers, a router's BGP router ID must be equal to the IP address listed in the eBGP neighbor's **neighbor** command.

4. A router has routes in the IP routing table for 20.0.0.0/8, 20.1.0.0/16, and 20.1.2.0/24. BGP on this router is configured with the **no auto-summary** command. Which of the following is true when using the BGP **network** command to cause these routes to be injected into the BGP table?

   a. The **network 20.0.0.0** command would cause all three routes to be added to the BGP table.

   b. The **network 20.0.0.0 mask 255.0.0.0** command would cause all three routes to be added to the BGP table.

   c. The **network 20.1.0.0 mask 255.255.0.0** command would cause 20.1.0.0/16 and 20.1.2.0/24 to be added to the BGP table.

   d. The **network 20.0.0.0** command would cause only 20.0.0.0/8 to be added to the BGP table.

5. A router has configured redistribution of EIGRP routes into BGP using the command **redistribute eigrp 1 route-map fred**. This router's BGP configuration includes the **no auto-summary** command. Which of the following are true?

   a. **route-map fred** can consider for redistribution routes listed in the IP routing table as EIGRP-learned routes.

   b. **route-map fred** can consider for redistribution routes in the IP routing table listed as connected routes, but only if those interfaces are matched by EIGRP 1's **network** commands.

   c. **route-map fred** can consider for redistribution routes that are listed in the EIGRP topology table as successor routes but that are not in the IP routing table because a lower administrative distance (AD) route from a competing routing protocol exists.

   d. **route-map fred** can consider for redistribution routes listed in the IP routing table as EIGRP-learned routes, but only if those routes also have at least one feasible successor route.

6. Using BGP, R1 has learned its best route to 9.1.0.0/16 from R3. R1 has a neighbor connection to R2, over a point-to-point serial link using subnet 8.1.1.4/30. R1 has **auto-summary** configured. Which of the following is true regarding what R1 advertises to R2?

   a. R1 advertises only 9.0.0.0/8 to R2, and not 9.1.0.0/16.

   b. If the **aggregate-address 9.0.0.0 255.0.0.0** BGP subcommand is configured, R1 advertises only 9.0.0.0/8 to R2, and not 9.1.0.0/16.

   c. If the **network 9.0.0.0 mask 255.0.0.0** BGP subcommand is configured, R1 advertises only 9.0.0.0/8 to R2, and not 9.1.0.0/16.

   d. None of the other answers is correct.

7. Which of the following statements are false regarding what routes a BGP router can advertise to a neighbor? (Assume no confederations or route reflectors are in use.)

   a. To advertise a route to an eBGP peer, the route cannot have been learned from an iBGP peer.

   b. To advertise a route to an iBGP peer, the route must have been learned from an eBGP peer.

   c. The NEXT_HOP IP address must respond to a ping command.

   d. Do not advertise routes if the neighboring router's AS is in the AS_PATH.

   e. The route must be listed as **valid** in the output of the **show ip bgp** command, but it does not have to be listed as **best**.

8. Several different routes were injected into BGP via various methods on R1. Those routes were then advertised via iBGP to R2. R2 summarized the routes using the **aggregate-address summary-only** command, and then advertised via eBGP to R3. Which of the following are true about the ORIGIN path attribute of these routes?

   a. The routes injected using the **network** command on R1 have an ORIGIN value of IGP.

   b. The routes injected using the **redistribute ospf** command on R1 have an ORIGIN value of IGP.

   c. The routes injected using the **redistribute** command on R1 have an ORIGIN value of EGP.

   d. The routes injected using the **redistribute static** command on R1 have an ORIGIN value of incomplete.

   e. If the **as-set** option was not used, the summary route created on R2 has an ORIGIN code of IGP.

9. Which of the following statements is true regarding the use of BGP synchronization?

   a. With BGP synchronization enabled, a router can add an iBGP-learned route to its IP routing table only if that same prefix is also learned via eBGP.

   b. With BGP synchronization enabled, a router cannot consider an iBGP-learned route as a "best" route to that prefix unless the NEXT_HOP IP address matches an IGP route in the IP routing table.

   c. BGP synchronization can be safely disabled when the routers inside a single AS either create a full mesh of BGP peers or create a hub-and-spoke to the router that learns the prefix via eBGP.

   d. None of the other answers is correct.

10. Which of the following statements are true regarding the operation of BGP confederations?

- **a.** Confederation eBGP connections act like normal (nonconfederation) eBGP connections with regard to the need for the **neighbor ebgp-multihop** command for nonadjacent neighbor IP addresses.
- **b.** iBGP-learned routes are advertised over confederation eBGP connections.
- **c.** A full mesh of iBGP peers inside a confederation sub-AS is not required.
- **d.** None of the other answers are correct.

11. R1 is BGP peered to R2, R3, R4, and R5 inside ASN 1, with no other peer connections inside the AS. R1 is a route reflector, serving R2 and R3 only. Each router also has an eBGP connection, through which it learns the following routes: 1.0.0.0/8 by R1, 2.0.0.0/8 by R2, 3.0.0.0/8 by R3, 4.0.0.0 by R4, and 5.0.0.0/8 by R5. Which of the following are true regarding the propagation of these routes?

- **a.** NLRI 1.0.0.0/8 is forwarded by R1 to each of the other routers.
- **b.** NLRI 2.0.0.0/8 is sent by R2 to R1, with R1 forwarding only to R3.
- **c.** NLRI 3.0.0.0/8 is sent by R3 to R1, with R1 forwarding to R2, R4, and R5.
- **d.** NLRI 4.0.0.0/8 is sent by R4 to R1, but R1 does not forward the information to R2 or R3.
- **e.** NLRI 5.0.0.0/8 is sent by R5 to R1; R1 reflects the route to R2 and R3, but not to R4.

12. R1 is in confederation ASN 65001; R2 and R3 are in confederation ASN 65023. R1 is peered to R2, and R2 is peered to R3. These three routers are perceived to be in AS 1 by eBGP peers. Which of the following is true regarding the configuration of these routers?

- **a.** Each of the three routers has a **router bgp 1** command.
- **b.** Both R2 and R3 need a **bgp confederation peers 65001** BGP subcommand.
- **c.** R1 needs a **bgp confederation identifier 1** BGP subcommand.
- **d.** Both R2 and R3 need a **bgp confederation identifier 65023** BGP subcommand.

Foundation Topics

Like Interior Gateway Protocols (IGPs), BGP exchanges topology information in order for routers to eventually learn the best routes to a set of IP prefixes. Unlike IGPs, BGP does not use a metric to select the best route among alternate routes to the same destination. Instead, BGP uses several BGP *path attributes (PAs)* and an involved decision process when choosing between multiple possible routes to the same subnet.

BGP uses the BGP *autonomous system path (AS_PATH)* PA as its default metric mechanism when none of the other PAs has been overly set and configured. Generally speaking, BGP uses PAs to describe the characteristics of a route; both this chapter and Chapter 13 cover the wide variety of BGP PAs. The AS_PATH attribute lists the path, as defined by a sequence of *autonomous system numbers (ASNs)* through which a packet must pass to reach a prefix. Figure 12-1 shows an example.

Figure 12-1 *BGP AS_PATHs and Path Vector Logic*

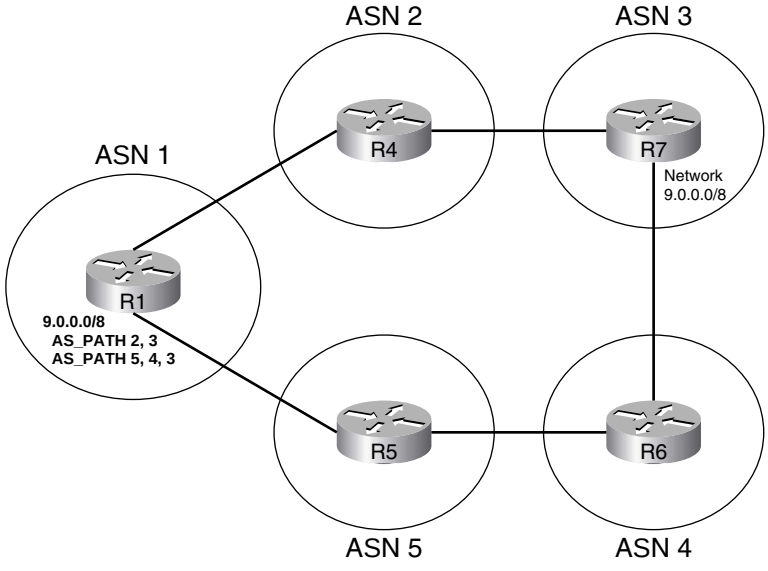


Figure 12-1 shows a classic case of how BGP uses *path vector* logic to choose routes. In the figure, R1 learns of two AS_PATHs by which to reach 9.0.0.0/8—through ASNs 2-3 and through ASNs 5-4-3. If none of the routers has used routing policies to influence other PAs that influence BGP's choice of which route is best, R1 will choose the shortest AS_PATH—in this case, AS_PATH 2-3. In effect, BGP treats the AS_PATH as a vector, and the length of the vector (the number of ASNs in the path) determines the best route. With BGP, the term *route* still refers to traditional hop-by-hop IP routes, but the term *path* refers to the sequence of autonomous systems used to reach a particular destination.

This chapter follows a similar sequence as several of the IGP chapters. First, the text focuses on neighbor relationships, followed by how BGP exchanges routing information with its neighbors. The chapter ends with a section covering how BGP adds IP routes to a router's IP routing table based on the BGP topology table.

Building BGP Neighbor Relationships

BGP neighbors form a TCP connection with each neighbor, sending BGP messages over the connections—culminating in *BGP Update* messages that contain the routing information. Each router explicitly configures its neighbors' IP addresses, using these definitions to tell a router with which IP addresses to attempt a TCP connection. Also, if a router receives a TCP connection request (to BGP port 179) from a source IP address that is not configured as a BGP neighbor, the router rejects the request.

After the TCP connection is established, BGP begins with BGP *Open* messages. Once a pair of BGP Open messages has been exchanged, the neighbors have reached the *established* state, which is the stable state of two working BGP peers. At this point, BGP Update messages can be exchanged.

This section examines many of the details about protocols and configuration for BGP neighbor formation. If you are already familiar with BGP, Table 12-2 summarizes some of the key facts found in this section.

Table 12-2 *BGP Neighbor Summary Table*

BGP Feature	Description and Values
TCP port	179
Setting the keepalive interval and hold time (using the **bgp timers** *keepalive holdtime* router subcommand or **neighbor timers** command, per neighbor)	Default to 60 and 180 seconds; define time between keepalives and time for which silence means the neighbor has failed
What makes a neighbor internal BGP (iBGP)?	Neighbor is in the same AS
What makes a neighbor external BGP (eBGP)?	Neighbor is in another AS
How is the BGP router ID (RID) determined?	In order: The **bgp router-id** command The highest IP of an up/up loopback at the time that the BGP process starts The highest IP of another up/up interface at the time that the BGP process starts.

continues

Table 12-2 *BGP Neighbor Summary Table (Continued)*

BGP Feature	Description and Values
How is the source IP address used to reach a neighbor determined?	Defined with the **neighbor update-source** command; or, by default, uses the outgoing interface IP address for the route used to reach the neighbor
How is the destination IP address used to reach a neighbor determined?	Explicitly defined on the **neighbor** command
Auto-summary*	Off by default, enabled with **auto-summary** router subcommand
Neighbor authentication	MD5 only, using the **neighbor password** command

* Cisco changed the IOS default for BGP **auto-summary** to be disabled as of Cisco IOS Software Release 12.3.

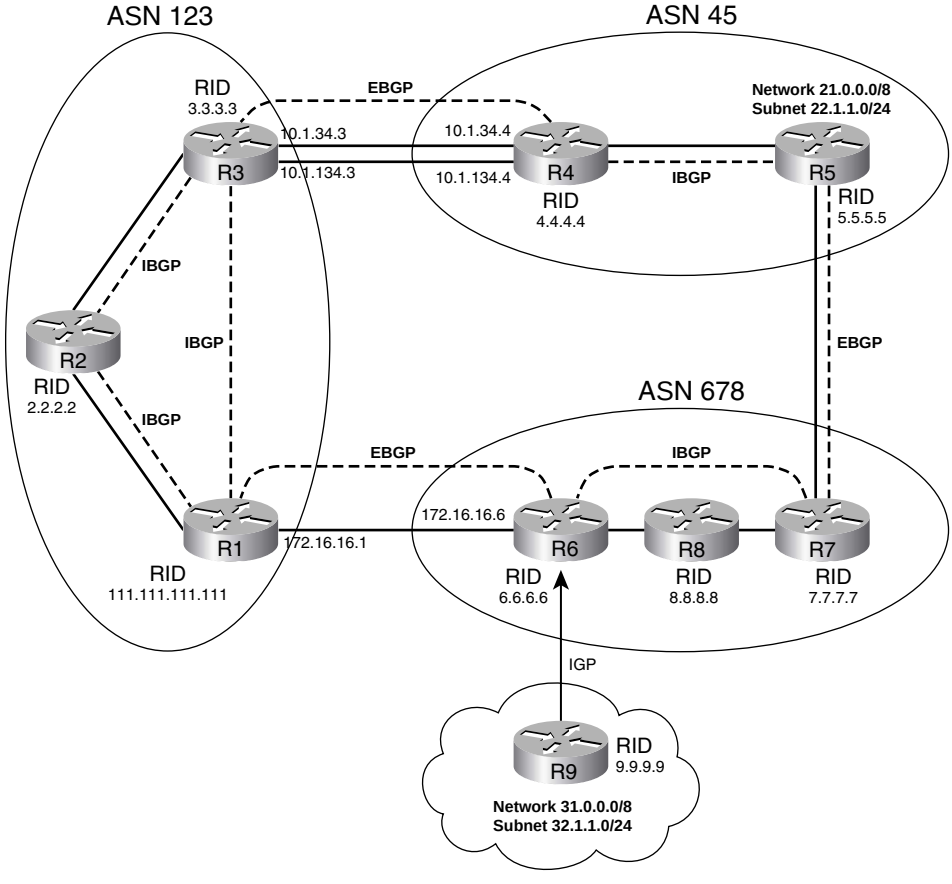
Internal BGP Neighbors

A BGP router considers each neighbor to be either an *internal BGP (iBGP)* peer or an *external BGP (eBGP)* peer. Each BGP router resides in a single AS, so neighbor relationships are either with other routers in the same AS (iBGP neighbors) or with routers in other autonomous systems (eBGP neighbors). The two types of neighbors differ only slightly in regard to forming neighbor relationships, with more significant differences in how the type of neighbor (iBGP or eBGP) impacts the BGP update process and the addition of routes to the routing tables.

iBGP peers often use loopback interface IP addresses for BGP peering to achieve higher availability. Inside a single AS, the physical topology often has at least two routes between each pair of routers. If BGP peers use an interface IP address for their TCP connections, and that interface fails, there still might be a route between the two routers, but the underlying BGP TCP connection will fail. Any time two BGP peers have more than one route through which they can reach the other router, peering using loopbacks makes the most sense.

Several examples that follow demonstrate BGP neighbor configuration and protocols, beginning with Example 12-1. The example shows some basic BGP configuration for iBGP peers R1, R2, and R3 in AS 123, with the following features, based on Figure 12-2.

Figure 12-2 *Sample Network for BGP Neighbor Configuration*



- The three routers in ASN 123 will form iBGP neighbor relationships with each other (full mesh).
- R1 will use the **bgp router-id** command to configure its RID, rather than use a loopback.
- R3 uses a **peer-group** configuration for neighbors R1 and R2. This allows fewer configuration commands, and improves processing efficiency by having to prepare only one set of outbound Update packets for the peer group. (Identical Updates are sent to all peers in the peer group.)
- The R1-R3 relationship uses BGP MD5 authentication, which is the only type of BGP authentication supported in Cisco IOS.

Example 12-1 *Basic iBGP Configuration of Neighbors*

```
! R1 Config—R1 correctly sets its update-source to 1.1.1.1 for both R2 and R3,
! in order to match the R2 and R3 neighbor commands. The first three highlighted
! commands below were not typed, but added automatically as defaults by IOS 12.3
!—in fact, IOS 12.3 docs imply that the defaults of sync and auto-summary at
! IOS 12.2 has changed to no sync and no auto-summary as of IOS 12.3. Also, R1
! knows that neighbors 2.2.2.2 and 3.3.3.3 are iBGP because their remote-as values
! match R1's router BGP command.
interface Loopback1
 ip address 1.1.1.1 255.255.255.255
!
router bgp 123
 no synchronization
 bgp router-id 111.111.111.111
 bgp log-neighbor-changes
 neighbor 2.2.2.2 remote-as 123
 neighbor 2.2.2.2 update-source Loopback1
 neighbor 3.3.3.3 remote-as 123
 neighbor 3.3.3.3 password secret-pw
 neighbor 3.3.3.3 update-source Loopback1
 no auto-summary
```

```
! R3 Config—R3 uses a peer group called "my-as" for combining commands related
! to R1 and R2. Note that not all parameters must be in the peer group: R3-R2 does
! not use authentication, but R3-R1 does, so the neighbor password command was
! not placed inside the peer group, but instead on a neighbor 1.1.1.1 command.
interface Loopback1
 ip address 3.3.3.3 255.255.255.255
!
router bgp 123
 no synchronization
 bgp log-neighbor-changes
 neighbor my-as peer-group
 neighbor my-as remote-as 123
 neighbor my-as update-source Loopback1
 neighbor 1.1.1.1 peer-group my-as
 neighbor 1.1.1.1 password secret-pw
 neighbor 2.2.2.2 peer-group my-as
 no auto-summary
```

```
! Next, R1 has two established peers, but the fact that the status is "established"
! is implied by not having the state listed on the right side of the output, under
! the heading State/PfxRcd. Once established, that column lists the number of
! prefixes learned via BGP Updates received from each peer. Note also R1's
! configured RID, and the fact that it is not used as the update source.
R1# show ip bgp summary
BGP router identifier 111.111.111.111, local AS number 123
BGP table version is 1, main routing table version 1

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
2.2.2.2         4   123      59      59        0    0    0 00:56:52        0
3.3.3.3         4   123      64      64        0    0    0 00:11:14        0
```

A few features in Example 12-1 are particularly important. First, note that the configuration does not overtly define peers as iBGP or eBGP. Instead, each router examines its own ASN as defined on the **router bgp** command, and compares that value to the neighbor's ASN listed in the **neighbor remote-as** command. If they match, the peer is iBGP; if not, the peer is eBGP.

R3 in Example 12-1 shows how to use the **peer-group** construct to reduce the number of configuration commands. BGP peer groups do not allow any new BGP configuration settings; they simply allow you to group BGP neighbor configuration settings into a group, and then apply that set of settings to a neighbor using the **neighbor peer-group** command. Additionally, BGP builds one set of Update messages for the peer group, applying routing policies (as covered in Chapter 13) for the entire group—rather than one router at a time—thereby reducing some BGP processing and memory overhead.

External BGP Neighbors

The physical topology between eBGP peers is often a single link, mainly because the connection is between different companies in different autonomous systems. As a result, eBGP peering can simply use the interface IP addresses for redundancy, because if the link fails, the TCP connection will fail because there is no longer an IP route between the peers. For instance, in Figure 12-2, the R1-R6 eBGP peering uses interface IP addresses defined in the **neighbor** commands.

When IP redundancy exists between two eBGP peers, the eBGP **neighbor** commands should use loopback IP addresses to take advantage of that redundancy. For example, two parallel links exist between R3 and R4. With **neighbor** commands that reference loopback addresses, either of these links could fail, but the TCP connection would remain. Example 12-2 shows additional configuration for the network in Figure 12-2, showing the use of loopbacks between R3 and R4, and interface addresses between R1 and R6.

Example 12-2 *Basic eBGP Configuration of Neighbors*

```
! R1 Config—This example shows only commands added since Example 12-1.
router bgp 123
 neighbor 172.16.16.6 remote-as 678
! R1 does not have a neighbor 172.16.16.6 update-source command configured. R1
! uses its s0/0/0.6 IP address, 172.16.16.1, because R1's route to 172.16.16.6
! uses s0/0/0.6 as the outgoing interface, as seen below.
R1# show ip route 172.1.16.6
Routing entry for 172.16.16.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Routing Descriptor Blocks:
  * directly connected, via Serial0/0/0.6
      Route metric is 0, traffic share count is 1
R1# show ip int brief | include 0/0/0.6
Serial0/0/0.6              172.16.16.1     YES manual up                    up
```

continues

Example 12-2 *Basic eBGP Configuration of Neighbors (Continued)*

```
! R3 Config—Because R3 refers to R4's loopback (4.4.4.4), and R4 is an eBGP
! peer, R3 and R4 have added the neighbor ebgp-multihop command to set TTL to 2.
! R3's update source must be identified as its loopback in order to match
! R4's neighbor 3.3.3.3 commands.
router bgp 123
 neighbor 4.4.4.4 remote-as 45
 neighbor 4.4.4.4 update-source loopback1
 neighbor 4.4.4.4 ebgp-multihop 2
! R3 now has three working neighbors. Also note the three TCP connections, one for
! each BGP peer. Note that because R3 is listed using a dynamic port number, and
! R4 as using port 179, R3 actually initiated the TCP connection to R4.
R3# show ip bgp summary
BGP router identifier 3.3.3.3, local AS number 123
BGP table version is 1, main routing table version 1

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
1.1.1.1         4   123     247     247        0    0    0 03:14:49        0
2.2.2.2         4   123     263     263        0    0    0 03:15:07        0
4.4.4.4         4    45      17      17        0    0    0 00:00:11        0
R3# show tcp brief
TCB       Local Address           Foreign Address        (state)
649DD08C  3.3.3.3.179             2.2.2.2.43521          ESTAB
649DD550  3.3.3.3.179             1.1.1.1.27222          ESTAB
647D928C  3.3.3.3.21449           4.4.4.4.179            ESTAB
```

The eBGP configurations differ from iBGP configuration in a couple of small ways. First, the **neighbor remote-as** commands refer to a different AS than does the **router bgp** command, which implies that the peer is an eBGP peer. Second, R3 had to configure the **neighbor 4.4.4.4 ebgp-multihop 2** command (and R4 with a similar command) or the peer connection would not have formed. For eBGP connections, Cisco IOS defaults the IP packet's TTL field to a value of 1, based on the assumption that the interface IP addresses will be used for peering (like R1-R6 in Example 12-2). In this example, if R3 had not used multihop, it would have sent packets to R4 with TTL 1. R4 would have received the packet (TTL 1 at that point), then attempt to route the packet to its loopback interface—a process that would decrement the TTL to 0, causing R4 to drop the packet. So, even though the router is only one hop away, think of the loopback as being on the other side of the router, requiring that extra hop.

Checks Before Becoming BGP Neighbors

Similar to IGPs, BGP checks certain requirements before another router may become a neighbor, reaching the BGP established state. Most of the settings are straightforward; the only tricky part relates to the use of IP addresses. The following list describes the checks that BGP performs when forming neighbor relationships:

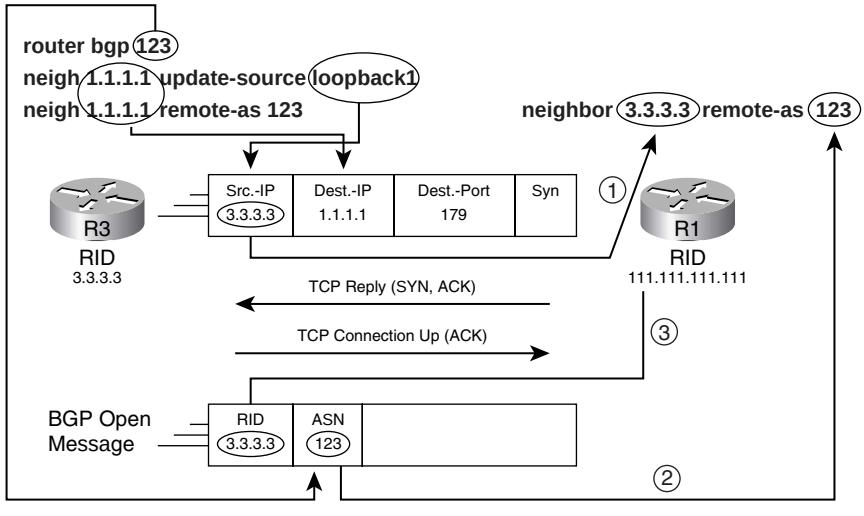
KEY POINT

1. The router must receive a TCP connection request with a source address that the router finds in a BGP **neighbor** command.

2. A router's ASN (on the **router bgp** *asn* command) must match the neighboring router's reference to that ASN with its **neighbor remote-as** *asn* command. (This requirement is not true of confederation configurations.)
3. The BGP RIDs of the two routers must not be the same.
4. If configured, MD5 authentication must pass.

Figure 12-3 shows the first three items in the list graphically, with R3 initiating a BGP TCP connection to R1. The circled numbers 1, 2, and 3 in the figure correspond to the item numbers in the previous list. Note that R1's check at Step 2 uses the **neighbor** command R1 identified as part of Step 1.

Figure 12-3 *BGP Neighbor Parameter Checking*



Note: R3's Loopback IP Address is 3.3.3.3

In Figure 12-3, R3 initiates a TCP connection with its update source IP address (3.3.3.3) as the source address of the packet. The first check occurs when R1 receives the first packet, looks at the source IP address of the packet (3.3.3.3), and finds that address in a **neighbor** command. The second check has R1 comparing R3's stated ASN (in R3's BGP Open message) to R1's **neighbor** command it identified at Step 1. Step 3 checks to ensure the BGP RIDs are unique, with the BGP Open message stating the sender's BGP RID.

While the check at Step 1 might seem intuitive, interestingly, the reverse bit of logic does not have to be true for the neighbors to come up. For instance, if R1 did not have a **neighbor 3.3.3.3 update-source 1.1.1.1** command, the process shown in Figure 12-3 would still work. Succinctly put, only one of the two routers' update source IP addresses needs to be in the other router's **neighbor** command for the neighbor to come up. Examples 12-1 and 12-2 showed the correct update source on both routers, and that makes good sense, but it works with only one of the two.

KEY POINT BGP uses a *keepalive timer* to define how often that router sends BGP keepalive messages, and a *Hold* timer to define how long a router will wait without receiving a keepalive message before resetting a neighbor connection. The Open message includes each router's stated keepalive timer. If they do not match, each router uses the lower of the values for each of the two timers, respectively. *Mismatched settings do not prevent the routers from becoming neighbors.*

BGP Messages and Neighbor States

The desired state for BGP neighbors is the established state. In that state, the routers have formed a TCP connection, and they have exchanged Open messages, with the parameter checks having passed. At this point, topology information can be exchanged using Update messages. Table 12-3 lists the BGP neighbor states, along with some of their characteristics. Note that if the IP addresses mismatch, the neighbors settle into an active state.

Table 12-3 *BGP Neighbor States*

KEY POINT

State	Listen for TCP?	Initiate TCP?	TCP Up?	Open Sent?	Open Received?	Neighbor Up?
Idle	No					
Connect	Yes					
Active	Yes	Yes				
Open sent	Yes	Yes	Yes	Yes		
Open confirm	Yes	Yes	Yes	Yes	Yes	
Established	Yes	Yes	Yes	Yes	Yes	Yes

BGP Message Types

BGP uses four basic messages. Table 12-4 lists the message types and provides a brief description of each.

Table 12-4 *BGP Message Types*

KEY POINT

Message	Purpose
Open	Used to establish a neighbor relationship and exchange basic parameters.
Keepalive	Used to maintain the neighbor relationship, with nonreceipt of a keepalive message within the negotiated Hold timer causing BGP to bring down the neighbor connection. (The timers can be configured with the **bgp timers** *keepalive holdtime* subcommand or the **neighbor** [*ip-address* | *peer-group-name*] **timers** *keepalive holdtime* BGP subcommand.)

Table 12-4 *BGP Message Types (Continued)*

Message	Purpose
Update	Used to exchange routing information, as covered more fully in the next section.
Notification	Used when BGP errors occur; causes a reset to the neighbor relationship when sent.

Purposefully Resetting BGP Peer Connections

Example 12-3 shows how to reset neighbor connections by using the **neighbor shutdown** command and, along the way, shows the various BGP neighbor states. The example uses routers R1 and R6 from Figure 12-2, as configured in Example 12-2.

Example 12-3 *Examples of Neighbor States*

```
! R1 shuts down R6's peer connection. debug ip bgp shows moving to a down state,
! which shows as "Idle (Admin)" under show ip bgp summary.
R1# debug ip bgp
BGP debugging is on for address family: BGP IPv4
R1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# router bgp 123
R1(config-router)# neigh 10.1.16.6 shutdown
R1#
*Mar  4 21:01:45.946: BGP: 10.1.16.6 went from Established to Idle
*Mar  4 21:01:45.946: %BGP-5-ADJCHANGE: neighbor 10.1.16.6 Down Admin. shutdown
*Mar  4 21:01:45.946: BGP: 10.1.16.6 closing
R1# show ip bgp summary ¦ include 10.1.16.6
10.1.16.6     4   678     353     353        0    0    0 00:00:06 Idle (Admin)
! Next, the no neighbor shutdown command reverses the admin state. The various
! debug messages (with some omitted) list the various states. Also note that the
! final message is the one log message in this example that occurs due to the
! default configuration of bgp log-neighbor-changes. The rest are the result of
! a debug ip bgp command.
R1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# router bgp 123
R1(config-router)# no neigh 10.1.16.6 shutdown
*Mar  4 21:02:16.958: BGP: 10.1.16.6 went from Idle to Active
*Mar  4 21:02:16.958: BGP: 10.1.16.6 open active, delay 15571ms
*Mar  4 21:02:29.378: BGP: 10.1.16.6 went from Idle to Connect
*Mar  4 21:02:29.382: BGP: 10.1.16.6 rcv message type 1, length (excl. header) 26
*Mar  4 21:02:29.382: BGP: 10.1.16.6 rcv OPEN, version 4, holdtime 180 seconds
*Mar  4 21:02:29.382: BGP: 10.1.16.6 went from Connect to OpenSent
*Mar  4 21:02:29.382: BGP: 10.1.16.6 sending OPEN, version 4, my as: 123, holdtime 180
  seconds
*Mar  4 21:02:29.382: BGP: 10.1.16.6 rcv OPEN w/ OPTION parameter len: 16
BGP: 10.1.16.6 rcvd OPEN w/ remote AS 678
```

continues

Example 12-3 *Examples of Neighbor States (Continued)*

```
*Mar  4 21:02:29.382: BGP: 10.1.16.6 went from OpenSent to OpenConfirm
*Mar  4 21:02:29.382: BGP: 10.1.16.6 send message type 1, length (incl. header) 45
*Mar  4 21:02:29.394: BGP: 10.1.16.6 went from OpenConfirm to Established
*Mar  4 21:02:29.398: %BGP-5-ADJCHANGE: neighbor 10.1.16.6 Up
```

All BGP neighbors can be reset with the **clear ip bgp *** exec command, which, like the **neighbor shutdown** command, resets the neighbor connection, closes the TCP connection to that neighbor, and removes all entries from the BGP table learned from that neighbor. The **clear** command will be shown in the rest of the chapter as needed, including in coverage of how to clear just some neighbors.

NOTE Chapter 13 covers how the **clear** command can be used to implement routing policy changes without resetting the neighbor completely, using a feature called *soft reconfiguration*.

Building the BGP Table

The BGP *topology table*, also called the BGP *Routing Information Base (RIB)*, holds the *network layer reachability information* (NLRI) learned by BGP, as well as the associated PAs. An NLRI is simply an IP prefix and prefix length. This section focuses on the process of how BGP injects NLRI into a router's BGP table, followed by how routers advertise their associated PAs and NLRI to neighbors.

NOTE Technically, BGP does not advertise routes; rather, it advertises PAs plus a set of NLRI that shares the same PA values. However, most people simply refer to NLRI as *BGP prefixes* or *BGP routes*. This book uses all three terms. However, because there is a distinction between a BGP route in the BGP table and an IP route in the IP routing table, the text takes care to refer to the BGP table or IP routing table to distinguish the two tables.

Injecting Routes/Prefixes into the BGP Table

Unsurprisingly, an individual BGP router adds entries to its local BGP table by using the same general methods used by IGPs: by using the **network** command, by hearing the topology information via an Update message from a neighbor, or by redistributing from another routing protocol. The next few sections show examples of how a local BGP router adds routes to the BGP table by methods other than learning them from a BGP neighbor.

The BGP network Command

This section, and the next section, assumes the BGP **no auto-summary** command has been configured. Note that as of the Cisco IOS Software Release 12.3 Mainline, **no auto-summary** is the default; earlier releases defaulted to use **auto-summary**. Following that, the section, "The Impact of

Auto-Summary on Redistributed Routes and the **network** Command," discusses the impact of the **auto-summary** command on both the **network** command and the **redistribute** command.

The BGP **network** router subcommand differs significantly from the **network** command used by IGPs. The BGP **network** command instructs that router's BGP process to do the following:

KEY POINT

Look for a route in the router's current IP routing table that exactly matches the parameters of the **network** command; if the IP route exists, put the equivalent NLRI into the local BGP table.

With this logic, connected routes, static routes, or IGP routes could be taken from the IP routing table and placed into the BGP table for later advertisement. When the router removes that route from its IP routing table, BGP then removes the NLRI from the BGP table, and notifies neighbors that the route has been withdrawn.

Note that the IP route must be matched exactly when the **no auto-summary** command is configured or used by default.

Table 12-5 lists a few of the key features of the BGP **network** command, whose generic syntax is:

```
network {network-number [mask network-mask]} [route-map map-tag]
```

Table 12-5 *Key Features of the BGP* **network** *Command*

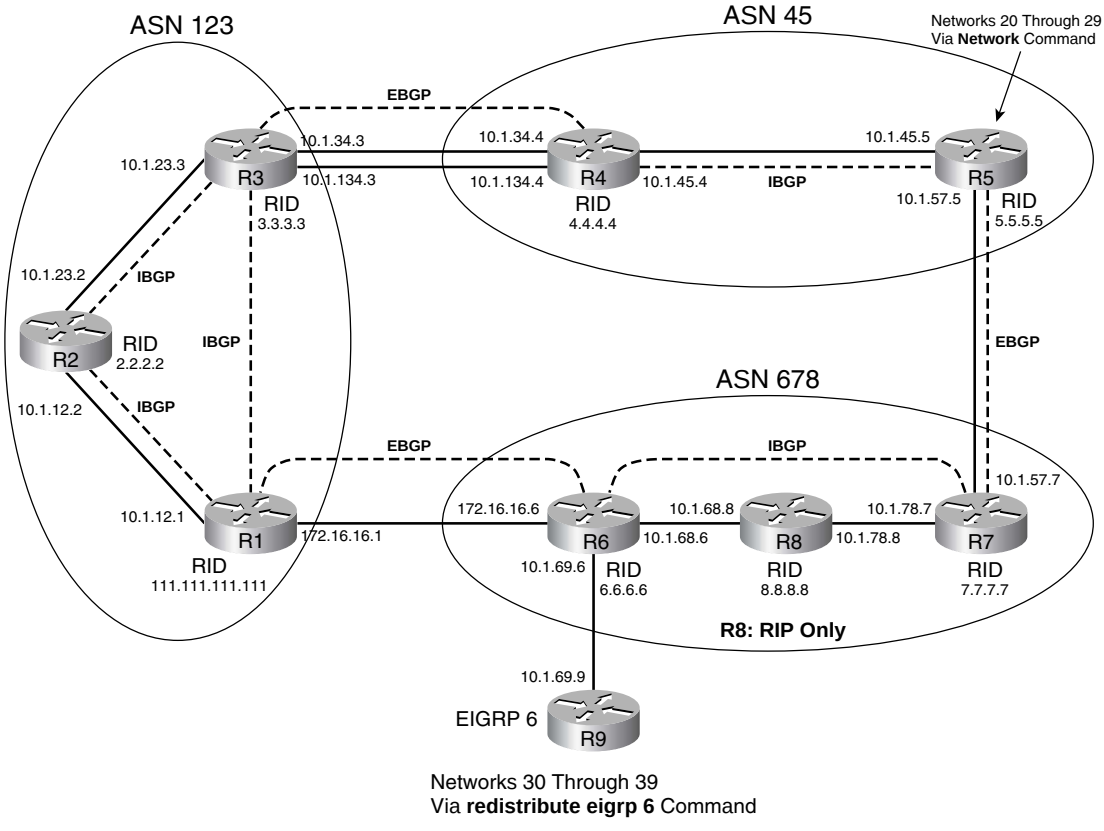
KEY POINT

Feature	Implication
No mask is configured	Assumes the default classful mask.
Matching logic with **no auto-summary** configured	An IP route must match both the prefix and prefix length (mask).
Matching logic with **auto-summary** configured	If the **network** command lists a classful network, it matches if any subnets of the classful network exist.
NEXT_HOP of BGP route added to the BGP table*	Uses next hop of IP route.
Maximum number injected by the **network** command into one BGP process	200
Purpose of the **route-map** option on the **network** command	Can be used to filter routes and manipulate PAs, including NEXT_HOP*.

*NEXT_HOP is a BGP PA that denotes the next-hop IP address that should be used to reach the NLRI.

Example 12-4 shows an example **network** command as implemented on R5 of Figure 12-4 (R5's BGP neighbors have been shut down so that the BGP table shows only BGP table entries created by the **network** commands on R5). In Example 12-4, R5 uses two **network** commands to add 21.0.0.0/8 and 22.1.1.0/24 to its BGP table.

Figure 12-4 *Sample BGP Network, with IP Addresses*



Example 12-4 *Examples of Populating the BGP Table via the* **network** *Command*

```
! On R5, the network commands specifically match prefixes 21.0.0.0/8 and 22.1.1.0/24. The
! omission of the mask on the first command implies the associated classful mask
! of 255.0.0.0, as the IP address listed (21.0.0.0) is a class A address.
router bgp 45
 no synchronization
 bgp log-neighbor-changes
 network 21.0.0.0
 network 22.1.1.0 mask 255.255.255.0
! The neighbor commands are not shown, as they are not pertinent to the topics
! covered in this example.
! Next, the two routes matched by the network commands are indeed in the IP
! routing table. Note that the route to 21.0.0.0/8 is a connected route, and the
! route to 22.1.1.0/24 is a static route.
R5# show ip route | incl 21 | 22
```

Example 12-4 *Examples of Populating the BGP Table via the* **network** *Command (Continued)*

```
C    21.0.0.0/8 is directly connected, Loopback20
     22.0.0.0/24 is subnetted, 1 subnets
S       22.1.1.0 [1/0] via 10.1.5.9
! Below, the prefixes have been added to the BGP table. Note that the NEXT_HOP
! PA has been set to 0.0.0.0 for the route (21.0.0.0/8) that was taken from a
! connected route, with the NEXT_HOP for 22.1.1.0/24 matching the IP route.
R5# show ip bgp
BGP table version is 38, local router ID is 5.5.5.5
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
*> 21.0.0.0         0.0.0.0                  0         32768 i
*> 22.1.1.0/24      10.1.5.9                 0         32768 i
```

Redistributing from an IGP, Static, or Connected Route

The BGP **redistribute** subcommand can redistribute static, connected, and IGP-learned routes. The mechanics of the BGP **redistribute** command work very similarly with redistribution as covered in Chapter 11, "IGP Route Redistribution, Route Summarization, and Default Routing"; however, this section covers a few nuances that are unique to BGP.

BGP does not use the concept of calculating a metric for each alternate route to reach a particular prefix. Instead, BGP uses a step-wise decision process that examines various PAs to determine the best route. As a result, redistribution into BGP does not require any consideration of setting metrics. However, as covered in Chapter 13, a router might need to apply a route map to the redistribution function to manipulate PAs, which in turn affects the BGP decision process. If a metric is assigned to a route injected into BGP, BGP assigns that metric value to the BGP *Multi-Exit Discriminator (MED)* PA, which is commonly referred to as *metric*.

NOTE Although this point is not unique to BGP, keep in mind that redistribution from an IGP causes two types of routes to be taken from the routing table—those learned by the routing protocol, and those connected routes for which that routing protocol matches with a **network** command.

Example 12-5 shows R6 (from Figure 12-4) filling its BGP table through route redistribution from Enhanced IGRP (EIGRP) process 6 (as configured in Example 12-5 with the **router eigrp 6** command) and redistributing a single static route. EIGRP on R6 learns routes only for networks 30 through 39. The goals of this example are as follows:

- Redistribute EIGRP routes for networks 31 and 32

- Redistribute the static route to network 34, and set the MED (metric) to 9
- Do not accidentally redistribute the connected routes that are matched by EIGRP's **network** commands
- Use the Cisco IOS 12.3 default setting of **no auto-summary**

Example 12-5 shows the mistake of accidentally redistributing additional routes—the connected subnets of network 10.0.0.0 matched by EIGRP **network** commands. Later in the example, a route map is added to prevent the problem.

Example 12-5 *Example of Populating the BGP Table via Redistribution*

```
! R6 redistributes EIGRP 6 routes and static routes below, setting the metric on
! redistributed static routes to 9. Note that EIGRP 6 matches subnets 10.1.68.0/24
! and 10.1.69.0/24 with its network command.
router bgp 678
 redistribute static metric 9
 redistribute eigrp 6
!
router eigrp 6
 network 10.0.0.0
!
ip route 34.0.0.0 255.0.0.0 null0
! Commands unrelated to populating the local BGP table are omitted.
! R6 has met the goal of injecting 31 and 32 from EIGRP, and 34 from static.
! It also accidentally picked up two subnets of 10.0.0.0/8 because EIGRP's network
! 10.0.0.0 command matched these connected subnets.
R6# show ip bgp
BGP table version is 1, local router ID is 6.6.6.6
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
*> 10.1.68.0/24     0.0.0.0                  0         32768 ?
*> 10.1.69.0/24     0.0.0.0                  0         32768 ?
*> 31.0.0.0         10.1.69.9           156160         32768 ?
*> 32.1.1.0/24      10.1.69.9           156160         32768 ?
*> 34.0.0.0/24      0.0.0.0                   9         32768 ?
! Below, note the metrics for the two EIGRP routes. The show ip bgp command output
! above shows how BGP assigned the MED (metric) that same value.
R6# show ip route eigrp
     32.0.0.0/24 is subnetted, 1 subnets
D       32.1.1.0 [90/156160] via 10.1.69.9, 00:12:17, FastEthernet0/0
D    31.0.0.0/8 [90/156160] via 10.1.69.9, 00:12:17, FastEthernet0/0
! Below, the redistribute eigrp command has been changed to the following, using
! a route map to only allow routes in networks in the 30s.
redist eigrp 6 route-map just-30-something
```

Example 12-5 *Example of Populating the BGP Table via Redistribution (Continued)*

```
! The route map and ACLs used for the filtering are shown next. As a result, the
! two subnets of 10.0.0.0/8 will not be redistributed into the BGP table.
R6# show route-map
route-map just-30-something, permit, sequence 10
  Match clauses:
    ip address (access-lists): permit-30-39
  Set clauses:
  Policy routing matches: 0 packets, 0 bytes
R6# show access-list
Standard IP access list permit-30-39
    10 permit 32.0.0.0, wildcard bits 7.255.255.255 (1538 matches)
    20 permit 30.0.0.0, wildcard bits 1.255.255.255 (1130 matches)
```

Also note that the NEXT_HOP PA for each route either matches the next hop of the redistributed route or is 0.0.0.0 for connected routes and routes to null0.

The Impact of Auto-Summary on Redistributed Routes and the network Command

As it does with IGPs, the BGP **auto-summary** command causes a classful summary route to be created if any component subnet of that summary exists. However, unlike IGPs, the BGP **auto-summary** router subcommand causes BGP to summarize only those routes *injected due to redistribution on that router*. BGP **auto-summary** does not look for classful network boundaries in the topology, and it does not look at routes already in the BGP table. It simply looks for routes injected into the BGP due to the **redistribute** and **network** commands on that same router.

The logic differs slightly based on whether the route is injected with the **redistribute** command or the **network** command. The logic for the two commands is summarized as follows:

KEY POINT

- **redistribute**—When the redistribution process would normally inject subnets of a classful network, do not inject the subnets into the routing table, but instead inject the classful network.
- **network**—For **network** commands that list a classful network number and no mask parameter, inject the classful network if at least one subnet of that classful network exists in the IP routing table.

While the preceding definitions are concise for study purposes, a few points deserve further emphasis and explanation. First, for redistribution, the **auto-summary** command causes the redistribution process to inject only classful networks into the local BGP table, and no subnets. The **network** command, with **auto-summary** configured, still injects subnets based on the same logic already described in this chapter. In addition to that logic, if a **network** command matches the classful **network** number, BGP injects the classful **network**, as long as at least any one subnet of that classful network exists in the IP routing table.

Example 12-6 shows an example that points out the impact of the **auto-summary** command. The example follows these steps on router R5 from Figure 12-2:

1. 10.15.0.0/16 is injected into BGP due to the **redistribute** command.
2. Auto-summary is configured, BGP is cleared, and now only 10.0.0.0/8 is in the BGP table.
3. Auto-summary and redistribution are disabled.
4. The **network 10.0.0.0** command, **network 10.12.0.0 mask 255.254.0.0** command, and **network 10.14.0.0 mask 255.255.0.0** command are configured. Only the last of these three commands exactly matches a current route, so only that route is injected into BGP.
5. Auto-summary is enabled, causing 10.0.0.0/8 to be injected, as well as the original 10.14.0.0/16 route.

Example 12-6 *Auto-Summary Impact on Routing Tables*

```
! R5 has shut down all neighbor connections, so the output of show ip bgp only shows
! routes injected on R5.
! Step 1 is below. Only 10.15.0.0/16 is injected by the current configuration. Note that
! the unrelated lines of output have been removed, and route-map only15 only
! matches 10.15.0.0/16.
R5# show run | be router bgp
router bgp 5
 no synchronization
 redistribute connected route-map only15
 no auto-summary
! Below, note the absence of 10.0.0.0/8 as a route, and the presence of 10.15.0.0/16,
! as well as the rest of the routes used in the upcoming steps.
R5# show ip route 10.0.0.0
Routing entry for 10.0.0.0/8, 4 known subnets
  Attached (4 connections)
  Redistributing via eigrp 99, bgp 5
  Advertised by bgp 5 route-map only15
C       10.14.0.0/16 is directly connected, Loopback10
C       10.15.0.0/16 is directly connected, Loopback10
C       10.12.0.0/16 is directly connected, Loopback10
C       10.13.0.0/16 is directly connected, Loopback10
! Only 10.15.0.0/16 is injected into BGP.
R5# show ip bgp
BGP table version is 2, local router ID is 5.5.5.5
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
*> 10.15.0.0/16     0.0.0.0                  0         32768 ?
! Next, step 2, where auto-summary is enabled. Now, 10.15.0.0/16 is no longer
```

Example 12-6 *Auto-Summary Impact on Routing Tables (Continued)*

```
! injected into BGP, but classful 10.0.0.0/8 is.
R5# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R5(config)# router bgp 5
R5(config-router)# no auto-summary
R5(config-router)# ^Z
R5# clear ip bgp *
R5# show ip bgp
BGP table version is 2, local router ID is 5.5.5.5
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
*> 10.0.0.0         0.0.0.0                  0         32768 ?
! Now, at step 3, no auto-summary disables automatic summarization, redistribution is
! disabled, and at step 4, the network commands are added. Note that 10.12.0.0/15 is
! not injected, as there is no exact match, nor is 10.0.0.0/8, as there is no exact
! match. However, 10.14.0.0/16 is injected due to the exact match of the prefix and
! prefix length.
R5# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R5(config)# router bgp 5
R5(config-router)# no auto-summary
R5(config-router)# no redist conn route-map only15
R5(config-router)# no redist connected
R5(config-router)# network 10.0.0.0
R5(config-router)# network 10.12.0.0 mask 255.254.0.0
R5(config-router)# network 10.14.0.0 mask 255.255.0.0
R5(config-router)# ^Z
R5# clear ip bgp *
R5# sh ip bgp ¦ begin network
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.14.0.0/16     0.0.0.0                  0         32768 i
! Finally, auto-summary is re-enabled (not shown in the example).
! 10.14.0.0/16 is still an exact match, so it is
! still injected. 10.0.0.0/8 is also injected because of the network 10.0.0.0 command.
R5# sh ip bgp | begin network
   Network          Next Hop            Metric LocPrf Weight Path
*  10.0.0.0         0.0.0.0                  0         32768 i
*  10.14.0.0/16     0.0.0.0                  0         32768 i
```

Manual Summaries and the AS_PATH Path Attribute

As covered in the last several pages, a router can add entries to its BGP table using the **network** command and route redistribution. Additionally, BGP can use manual route summarization to advertise summary routes to neighboring routers, causing the neighboring routers to learn additional BGP routes. BGP manual summarization with the **aggregate-address** command differs significantly from using the **auto-summary** command. It can summarize based on any routes in the BGP table, creating a summary of any prefix length. It does not always suppress the advertisement of the component subnets, although it can be configured to do so.

The aggregate route must include the AS_PATH PA, just like it is required for every other NLRI in the BGP table. However, to fully understand what this command does, you need to take a closer look at the AS_PATH PA.

The AS_PATH PA consists of up to four different components, called segments, as follows:

- AS_SEQ (short for AS Sequence)
- AS_SET
- AS_CONFED_SEQ (short for AS Confederation Sequence)
- AS_CONFED_SET

The most commonly used segment is called AS_SEQ. AS_SEQ is the idea of AS_PATH as shown back in Figure 12-1, with the PA representing all ASNs, in order, through which the route has been advertised.

However, the **aggregate-address** command can create a summary route for which the AS_SEQ must be null. When the component subnets of the summary route have differing AS_SEQ values, the router simply can't create an accurate representation of AS_SEQ, so it uses a null AS_SEQ. However, this action introduces the possibility of creating routing loops, because the contents of AS_PATH, specifically AS_SEQ, are used to prevent a route from being re-advertised to an AS that has already heard about the route.

The AS_PATH AS_SET segment solves the problem when the summary route has a null AS_SEQ. The AS_SET segment holds an unordered list of all the ASNs in all the component subnets' AS_SEQ segments.

Example 12-7 shows an example in which the router does use a null AS_SEQ for a summary route, and then the same summary with the **as-set** option creating the AS_SET segment.

NOTE AS_PATH includes the AS_CONFED_SEQ and AS_CONFED_SET segments as well, which are covered later, in the section "Confederations."

The following list summarizes the actions taken by the **aggregate-address** command when it creates a summary route:

KEY POINT

- It does not create the summary if the BGP table does not currently have any routes for NLRI inside the summary.
- If all the component subnets are withdrawn from the aggregating router's BGP table, it also then withdraws the aggregate. (In other words, the router tells its neighbors that the aggregate route is no longer valid.)
- It sets the NEXT_HOP address of the summary, as listed in the local BGP table, as 0.0.0.0.
- It sets the NEXT_HOP address of the summary route, as advertised to neighbors, to the router's update source IP address for each neighbor, respectively.
- If the component subnets inside the summary all have the same AS_SEQ, it sets the new summary route's AS_SEQ to be exactly like the AS_SEQ of the component subnets.
- If the AS_SEQ of the component subnets differs in any way, it sets the AS_SEQ of the new summary route to null.
- When the **as-set** option has been configured, the router creates an AS_SET segment for the aggregate route, but only if the summary route's AS_SEQ is null.
- As usual, if the summary is advertised to an eBGP peer, the router prepends its own ASN to the AS_SEQ before sending the Update.
- It suppresses the advertisement of all component subnets if the **summary-only** keyword is used; advertises all of them if the **summary-only** keyword is omitted; or advertises a subset if the **suppress-map** option is configured. (Refer to Chapter 13 for an example of using the **suppress-map** option.)

Example 12-7 shows R3 from Figure 12-4 summarizing 23.0.0.0/8. R3 advertises the summary with ASN 123 as the only AS in the AS_SEQ, because some component subnets have AS_PATHS of 45, and others have 678 45. As a result, R3 uses a null AS_SEQ for the aggregate. The example goes on to show the impact of the **as-set** option.

Example 12-7 *Route Aggregation and the* **as-set** *Option*

```
! Note that R3's routes to network 23 all have the same AS_PATH except one new
! prefix, which has an AS_PATH that includes ASN 678. As a result, R3 will
! create a null AS_SEQ for the summary route.
R3# show ip bgp | include 23
*> 23.3.0.0/20      4.4.4.4                                0 45 i
*> 23.3.16.0/20     4.4.4.4                                0 45 i
*> 23.3.32.0/19     4.4.4.4                                0 45 i
```

continues

Example 12-7 *Route Aggregation and the* **as-set** *Option (Continued)*

```
*> 23.3.64.0/18     4.4.4.4                                0 45 i
*> 23.3.128.0/17    4.4.4.4                                0 45 i
*> 23.4.0.0/16      4.4.4.4                                0 45 678 i
! The following command is now added to R3's BGP configuration:
aggregate-address 23.0.0.0 255.0.0.0 summary-only
! Note: R3 will not have a BGP table entry for 23.0.0.0/8; however, R3 will
! advertise this summary to its peers, because at least one component subnet
! exists.
```

```
! R1 has learned the prefix, NEXT_HOP 3.3.3.3 (R3's update source IP address for
! R1), but the AS_PATH is now null because R1 is in the same AS as R3.
! (Had R3-R1 been an eBGP peering, R3 would have prepended its own ASN.)
! Note that the next command is on R1 R1 R1 R1.
R1# sh ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*>i21.0.0.0         3.3.3.3                  0    100      0 45 i
*>i23.0.0.0         3.3.3.3                  0    100      0 i
! Next, R1 displays the AGGREGATOR PA, which identifies R3 (3.3.3.3) and its AS
! (123) as the aggregation point at which information is lost. Also, the phrase
! "atomic-aggregate" refers to the fact that the ATOMIC_AGGREGATE PA has also
! been set; this PA simply states that this NLRI is a summary.
R1# show ip bgp 23.0.0.0
BGP routing table entry for 23.0.0.0/8, version 45
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Flag: 0x800
  Advertised to update-groups:
     2
  Local, (aggregated by 123 3.3.3.3), (received & used)
    3.3.3.3 (metric 2302976) from 3.3.3.3 (3.3.3.3)
      Origin IGP, metric 0, localpref 100, valid, internal, atomic-aggregate, best
```

```
! R6, in AS 678, receives the summary route from R1, but the lack of information
! in the current AS_PATH allows R6 to learn of the route, possibly causing
! a routing loop. (Remember, one of the component subnets, 23.4.0.0/16, came from
! ASN 678.)
R6# sh ip bgp nei 172.16.16.1 received-routes ¦ begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 21.0.0.0         172.16.16.1                            0 123 45 i
*> 23.0.0.0         172.16.16.1                            0 123 i
```

```
! The R3 configuration is changed as shown next to use the as-set option.
R3# aggregate-address 23.0.0.0 255.0.0.0  summary-only as-set
```

```
! R1 now has the AS_SET component of the AS_PATH PA, which includes an unordered
! list of all autonmous systems from all the component subnets' AS_PATHs on R3.
R1# sh ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*>i21.0.0.0         3.3.3.3                  0    100      0 45 i
*>i23.0.0.0         3.3.3.3                  0    100      0 {45,678} i
```

Example 12-7 *Route Aggregation and the* **as-set** *Option (Continued)*

```
! Now R6 does not receive the 23.0.0.0 prefix due to R1's check of the AS_SET PA,
! noticing that ASN 678 is in the AS_SET and is also R6's ASN.
R6# sh ip bgp nei 172.16.16.1 received-routes | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 21.0.0.0         172.16.16.1         0 123 45 i
```

NOTE Summary routes can also be added via another method. First, the router would create a static route, typically with destination of interface null0. Then, the prefix/length can be matched with the **network** command to inject the summary. This method does not filter any of the component subnets.

Table 12-6 summarizes the key points regarding summarization using the **aggregate-address**, **auto-summary**, and **network** commands.

Table 12-6 *Summary: Injecting Summary Routes in BGP*

KEY POINT

Command	Component Subnets Removed	Routes It Can Summarize
auto-summary (with redistribution)	All	Only those injected into BGP on that router using the **redistribute** command
aggregate-address	All, none, or a subset	Any prefixes already in the BGP table
auto-summary (with the **network** command)	None	Only those injected into BGP on that router using the **network** command

Adding Default Routes to BGP

The final method covered in this chapter for adding routes to a BGP table is to inject default routes into BGP. Default routes can be injected into BGP in one of three ways:

- By injecting the default using the **network** command
- By injecting the default using the **redistribute** command
- By injecting a default route into BGP using the **neighbor** *neighbor-id* **default-information** [**route-map** *route-map-name*] BGP subcommand

When injecting a default route into BGP using the **network** command, a route to 0.0.0.0/0 must exist in the local routing table, and the **network 0.0.0.0** command is required. The default IP route can be learned via any means, but if it is removed from the IP routing table, BGP removes the default route from the BGP table.

Injecting a default route through redistribution requires an additional configuration command—**default-information originate**. The default route must first exist in the IP routing table; for instance, a static default route to null0 could be created. Then, the **redistribute static** command

could be used to redistribute that static default route. However, in the special case of the default route, Cisco IOS also requires the **default-information originate** BGP subcommand.

Injecting a default route into BGP by using the **neighbor** *neighbor-id* **default-information** [**route-map** *route-map-name*] BGP subcommand does not add a default route to the local BGP table; instead, it causes the advertisement of a default to the specified neighbor. In fact, this method does not even check for the existence of a default route in the IP routing table by default, but it can. With the **route-map** option, the referenced route map examines the entries in the IP routing table (not the BGP table); if a route map **permit** clause is matched, then the default route is advertised to the neighbor. Example 12-8 shows just such an example on R1, with **route-map check-default** checking for the existence of a default route before R1 would originate a default route to R3.

Example 12-8 *Originating a Default Route to a Neighbor with the* **neighbor default-originate** *Command*

```
! The pertinent parts of the R1 configuration are listed next, with the route map
! matching an IP route to 0.0.0.0/0 with a permit action, enabling the
! advertisement of a default route to neighbor 3.3.3.3 (R3).
router bgp 123
 neighbor 3.3.3.3 remote-as 123
 neighbor 3.3.3.3 update-source Loopback1
 neighbor 3.3.3.3 default-originate route-map check-default
!
ip route 0.0.0.0 0.0.0.0 Null0
!
ip prefix-list def-route seq 5 permit 0.0.0.0/0
!
route-map check-default permit 10
 match ip address prefix-list def-route
! R1 indeed has a default route, as seen below.
R1# show ip route | include 0.0.0.0/0
S*   0.0.0.0/0 is directly connected, Null0
```

```
! R3 now learns a default route from R1, as seen below.
R3# show ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*>i0.0.0.0          1.1.1.1                       100      0 i
```

The ORIGIN Path Attribute

Depending on the method used to inject a route into a local BGP table, BGP assigns one of three BGP ORIGIN PA codes: IGP, EGP, or incomplete. The ORIGIN PA provides a general descriptor as to how a particular NLRI was first injected into a router's BGP table. The **show ip bgp** command includes the three possible values in the legend at the top of the command output, listing the actual ORIGIN code for each BGP route at the far right of each output line. Table 12-7 lists the three ORIGIN code names, the single-letter abbreviation used by Cisco IOS, and the reasons why a route is assigned a particular code.

The ORIGIN codes and meanings hide a few concepts that many people find counterintuitive. First, routes redistributed into BGP from an IGP actually have an ORIGIN code of incomplete. Also, do

not confuse EGP with eBGP; an ORIGIN of EGP refers to Exterior Gateway Protocol, the very old and deprecated predecessor to BGP. In practice, the EGP ORIGIN code should not be seen today.

Table 12-7 *BGP ORIGIN Codes*

KEY POINT

ORIGIN Code	Cisco IOS Notation	Used for Routes Injected Due to the Following Commands
IGP	**i**	**network**, **aggregate-address** (in some cases), and **neighbor default-originate** commands
EGP	**e**	Exterior Gateway Protocol (EGP). No specific commands apply.
Incomplete	**?**	**redistribute**, **aggregate-address** (in some cases), and **default-information originate** command

The rules regarding the ORIGIN codes used for summary routes created with the **aggregate-address** command can also be a bit surprising. The rules are summarized as follows:

KEY POINT

- If the **as-set** option is not used, the aggregate route uses ORIGIN code **i**.
- If the **as-set** option is used, and all component subnets being summarized use ORIGIN code **i**, the aggregate has ORIGIN code **i**.
- If the **as-set** option is used, and at least one of the component subnets has an ORIGIN code **?**, the aggregate has ORIGIN code **?**.

NOTE The BGP ORIGIN PA provides a minor descriptor for the origin of a BGP table entry, which is used as part of the BGP decision tree (covered in Chapter 13).

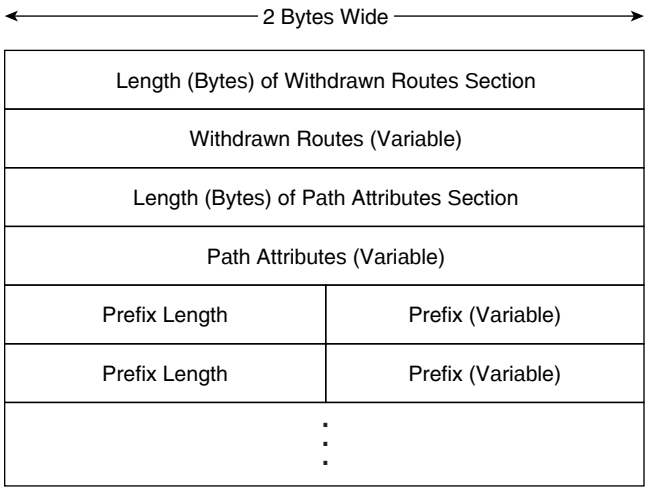
Advertising BGP Routes to Neighbors

The previous section focused on the tools that BGP can use to inject routes into a local router's BGP table. BGP routers take routes from the local BGP table and advertise a subset of those routes to their BGP neighbors. This section continues focusing on the BGP table because the BGP route advertisement process takes routes from the BGP table and sends them to neighboring routers, where the routes are added to the neighbors' BGP tables. Later, the final major section in the chapter, "Building the IP Routing Table," focuses on the rules regarding how BGP places routes into the IP routing table.

The BGP Update Message

Once a BGP table has a list of routes, paths, and prefixes, the router needs to advertise the information to neighboring routers. To do so, a router sends BGP Update messages to its neighbors. Figure 12-5 shows the general format of the BGP Update message.

Figure 12-5 *BGP Update Message Format*



Each Update message has three main parts:

- The Withdrawn Routes field enables BGP to inform its neighbors about failed routes.
- The Path Attributes field lists the PAs for each route. NEXT_HOP and AS_PATH are sample values for this field.
- The Prefix and Prefix Length fields define each individual NLRI.

The central concept in an individual Update message is the set of PAs. Then, all the prefixes (NLRIs) that share the exact same set of PAs and PA values are included at the end of the Update message. If a router needs to advertise a set of NLRIs, and each NLRI has a different setting for at least one PA, then separate Update messages will be required for each NLRI. However, when many routes share the same PAs—typical of prefixes owned by a particular ISP, for instance—multiple NLRIs are included in a single Update. This reduces router CPU load and uses less link bandwidth.

Determining the Contents of Updates

A router builds the contents of its Update messages based on the contents of its BGP table. However, the router must choose which subset of its BGP table entries to advertise to each neighbor, with the set likely varying from neighbor to neighbor. Table 12-8 summarizes the rules about which routes BGP does *not* include in routing updates to each neighbor; each rule is described more fully following the table.

Table 12-8 *Summary of Rules Regarding Which Routes BGP Does Not Include in an Update*

KEY POINT

iBGP and/or eBGP	Routes Not Taken from the BGP Table
Both	Routes that are not considered "best"
Both	Routes matched by a **deny** clause in an outbound BGP filter
iBGP	iBGP-learned routes*
eBGP	Routes whose AS_PATH includes the ASN of the eBGP peer to which a BGP Update will be sent

*This rule is relaxed or changed as a result of using route reflectors or confederations.

BGP only advertises a route to reach a particular subnet (NLRI) if that route is considered to be the best route. If a BGP router learns of only one route to reach a particular prefix, the decision process is very simple. However, when choosing between multiple paths to reach the same prefix, BGP determines the best route based on a lengthy BGP decision process, as described in detail in Chapter 13. Assuming that none of the routers has configured any routing policies that impact the decision process, the decision tree reduces to a four-step process that is mainly comprised of tie-breakers, as follows:

1. Choose the route with the shortest AS_PATH.
2. If AS_PATH length is a tie, prefer a single eBGP-learned route over one or more iBGP routes.
3. If the best route has not yet been chosen, choose the route with the lowest IGP metric to the NEXT_HOP of the routes.
4. If the IGP metric ties, choose the iBGP-learned route with the lowest BGP RID of the advertising router.

Additionally, BGP rules out some routes from being considered best based on the value of the NEXT_HOP PA. For a route to be a candidate to be considered best, the NEXT_HOP must be either:

- 0.0.0.0, as the result of the route being injected on the local router.
- Reachable according to that router's current IP routing table. In other words, the NEXT_HOP IP address must match a route in the routing table.

Because the NEXT_HOP PA is so important with regard to BGP's choice of its best path to reach each NLRI, this section summarizes the logic and provides several examples. The logic is separated into two parts based on whether the route is being advertised to an iBGP or eBGP peer. By default, when sending to an eBGP peer, the NEXT_HOP is changed to an IP address on the advertising router—specifically, to the same IP address the router used as the source IP address of the BGP Update message, for each respective neighbor. When sending to an iBGP peer, the default action is to leave the NEXT_HOP PA unchanged. Both of these default behaviors can be changed via the commands listed in Table 12-9.

Table 12-9 *Conditions for Changing the NEXT_HOP PA*

KEY POINT

Type of Neighbor	Default Action for Advertised Routes	Command to Switch to Other Behavior
iBGP	Do not change the NEXT_HOP	**neighbor… next-hop-self**
eBGP	Change the NEXT_HOP to the update source IP address	**neighbor… next-hop-unchanged**

Note that the NEXT_HOP PA cannot be set via a route map; the only way to change the NEXT_HOP PA is through the methods listed in Table 12-9.

Example: Impact of the Decision Process and NEXT_HOP on BGP Updates

The next several examples together show a sequence of events regarding the propagation of network 31.0.0.0/8 by BGP throughout the network of Figure 12-4. R6 originated the routes in the 30s (as in Example 12-4) by redistributing EIGRP routes learned from R9. The purpose of this series of examples is to explain how BGP chooses which routes to include in Updates under various conditions.

The first example, Example 12-9, focuses on the commands used to examine what R6 sends to R1, what R1 receives, and the resulting entries in R1's BGP table. The second example, Example 12-10, then examines those same routes propagated from R1 to R3, including problems related to R1's default behavior of not changing the NEXT_HOP PA of those routes. Finally, Example 12-11 shows the solution of R1's use of the **neighbor 3.3.3.3 next-hop-self** command, and the impact that has on the contents of the BGP Updates in AS 123.

Example 12-9 *R6 Sending the 30s Networks to R1 Using BGP*

```
! R6 has injected the three routes listed below; they were not learned from
! another BGP neighbor. Note all three show up as >, meaning they are the best
! (and only in this case) routes to the destination NLRIs.
R6# show ip bgp
BGP table version is 5, local router ID is 6.6.6.6
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
*> 31.0.0.0         10.1.69.9           156160         32768 ?
*> 32.0.0.0         0.0.0.0                            32768 i
*> 32.1.1.0/24      10.1.69.9           156160         32768 ?
! R6 now lists the routes it advertises to R1—sort of. This command lists R6's
! BGP table entries that are intended to be sent, but R6 can (and will in this
! case) change the information before advertising to R1. Pay particular attention
! to the Next Hop column, versus upcoming commands on R1. In effect, this command
shows R6's current BGP table entries that will be sent to R1, but it shows them
```

Example 12-9 *R6 Sending the 30s Networks to R1 Using BGP (Continued)*

```
before R6 makes any changes, including NEXT_HOP.
R6# show ip bgp neighbor 172.16.16.1 advertised-routes
BGP table version is 5, local router ID is 6.6.6.6
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
*> 31.0.0.0         10.1.69.9           156160         32768 ?
*> 32.0.0.0         0.0.0.0                            32768 i
*> 32.1.1.0/24      10.1.69.9           156160         32768 ?
Total number of prefixes 3
```

```
! The next command (R1) lists the info in the received BGP update from R6. Note
! that the NEXT_HOP is different; R6 changed the NEXT_HOP before sending the
! update, because it has an eBGP peer connection to R1, and eBGP defaults to set
! NEXT_HOP to itself. As R6 was using 172.16.16.6 as the IP address from which to
! send BGP messages to R1, R6 set NEXT_HOP to that number. Also note that R1 lists
! the neighboring AS (678) in the Path column at the end, signifying the AS_PATH
! for the route.
R1# show ip bgp neighbor 172.16.16.6 received-routes
BGP table version is 7, local router ID is 111.111.111.111
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
*> 31.0.0.0         172.16.16.6         156160             0 678 ?
*> 32.0.0.0         172.16.16.6              0             0 678 i
*> 32.1.1.0/24      172.16.16.6         156160             0 678 ?
Total number of prefixes 3
! The show ip bgp summary command lists the state of the neighbor until the
! neighbor becomes established; at that point, the State/PfxRcd column lists the number
! of NLRIs (prefixes) received (and still valid) from that neighbor.
R1# show ip bgp summary | begin Neighbor
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
2.2.2.2         4   123      55      57        7    0    0 00:52:30        0
3.3.3.3         4   123      57      57        7    0    0 00:52:28        3
172.16.16.6     4   678      53      51        7    0    0 00:48:50        3
! R1 has also learned of these prefixes from R3, as seen below. The routes through
! R6 have one AS in the AS_PATH, and the routes through R3 have two autonmous systems, so the
! routes through R6 are best. Also, the iBGP routes have an "i" for "internal"
! just before the prefix.
R1# show ip bgp
BGP table version is 7, local router ID is 111.111.111.111
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
```

continues

Example 12-9 *R6 Sending the 30s Networks to R1 Using BGP (Continued)*

```
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i31.0.0.0         3.3.3.3                  0    100      0 45 678 ?
*>                  172.16.16.6         156160             0 678 ?
* i32.0.0.0         3.3.3.3                  0    100      0 45 678 i
*>                  172.16.16.6              0             0 678 i
* i32.1.1.0/24      3.3.3.3                  0    100      0 45 678 ?
*>                  172.16.16.6         156160             0 678 ?
```

Example 12-9 showed examples of how you can view the contents of the actual Updates sent to neighbors (using the **show ip bgp neighbor advertised-routes** command) and the contents of Updates received from a neighbor (using the **show ip bgp neighbor received-routes** command). RFC 1771 suggests that the BGP RIB can be separated into components for received Updates from each neighbor and sent Updates for each neighbor. Most implementations (including Cisco IOS) keep a single RIB, with notations as to which entries were sent and received to and from each neighbor.

NOTE For the **received-routes** option to work, the router on which the command is used must have the **neighbor** *neighbor-id* **soft-reconfiguration inbound** BGP subcommand configured for the other neighbor.

These **show ip bgp neighbor** commands with the **advertised-routes** option list the BGP table entries that will be advertised to that neighbor. However, note that any changes to the PAs inside each entry are not shown in the command output. For example, the **show ip bgp neighbor 172.16.16.1 advertised-routes** command on R6 listed the NEXT_HOP for 31/8 as 10.1.69.9, which is true of that entry in R6's BGP table. R6 then changes the NEXT_HOP PA before sending the actual Update, with a NEXT_HOP of 172.16.16.6.

By the end of Example 12-9, R1 knows of both paths to each of the three prefixes in the 30s (AS_PATH 678 and 45-678), but has chosen the shortest AS_PATH (through R6) as the best path in each case. Note that the > in the **show ip bgp** output designates the routes as R1's best routes. Next, Example 12-10 shows some possibly surprising results on R3 related to its choices of best routes.

Example 12-10 *Examining the BGP Table on R3*

```
! R1 now updates R3 with R1's "best" routes
R1# show ip bgp neighbor 3.3.3.3 advertised-routes | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 31.0.0.0         172.16.16.6         156160             0 678 ?
*> 32.0.0.0         172.16.16.6              0             0 678 i
*> 32.1.1.0/24      172.16.16.6         156160             0 678 ?
```

Example 12-10 *Examining the BGP Table on R3 (Continued)*

```
Total number of prefixes 3
! R3 received the routes, but R3's best routes to each prefix point back to
! R4 in AS 45, with AS_PATH 45-678, which is a longer path. The route through R1
! cannot be "best" because the NEXT_HOP was sent unchanged by iBGP neighbor R1.
R3# show ip bgp
BGP table version is 7, local router ID is 3.3.3.3
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
*> 31.0.0.0         4.4.4.4                                0 45 678 ?
* i                 172.16.16.6         156160    100      0 678 ?
*> 32.0.0.0         4.4.4.4                                0 45 678 i
* i                 172.16.16.6              0    100      0 678 i
*> 32.1.1.0/24      4.4.4.4                                0 45 678 ?
* i                 172.16.16.6         156160    100      0 678 ?
! Proof that R3 cannot reach the next-hop IP address is shown next.
R3# ping 172.16.16.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.16.6, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
```

Example 12-10 points out a quirk with some terminology in the **show ip bgp** command output, as well as an important design choice with BGP. First, the command output lists * as meaning valid; however, that designation simply means that the route is a candidate for use. Before the route can be actually used and added to the IP routing table, the NEXT_HOP must also be reachable. In some cases, routes that the **show ip bgp** command considers "valid" might not be usable routes, with Example 12-10 showing just such an example.

Each BGP route's NEXT_HOP must be reachable for a route to be truly valid. With all default settings, an iBGP-learned route has a NEXT_HOP IP address of the last eBGP router to advertise the route. For example, R3's route to 31.0.0.0/8 through R1 lists R6's IP address (172.16.16.6) in the NEXT_HOP field. Unfortunately, R3 does not have a route for 172.16.16.6, so that route cannot be considered "best" by BGP.

There are two easy choices to solve the problem:

KEY POINT

- Make the eBGP neighbor's IP address reachable by advertising that subnet into the IGP.
- Use the **next-hop-self** option on the **neighbor** command that points to iBGP peers.

The first option typically can be easily implemented. Because many eBGP neighbors use interface IP addresses on their **neighbor** commands, the NEXT_HOP exists in a subnet directly connected to the AS. For example, R1 is directly connected to 172.16.16.0/24, so R1 could simply advertise that connected subnet into the IGP inside the AS.

However, this option might be problematic when loopback addresses are used for BGP neighbors. For example, if R1 had been configured to refer to R6's 6.6.6.6 loopback IP address, and it was working, R1 must have a route to reach 6.6.6.6. However, it is less likely that R1 would already be advertising a route to reach 6.6.6.6 into ASN 123.

The second option causes the router to change the NEXT_HOP PA to one of its own IP addresses—an address that is more likely to already be in the neighbor's IP routing table, which works well even if using loopbacks with an eBGP peer. Example 12-11 points out such a case, with R1 using the **neighbor next-hop-self** command, advertising itself (1.1.1.1) as the NEXT_HOP. As a result, R3 changes its choice of best routes, because R3 has a route to reach 1.1.1.1, overcoming the "NEXT_HOP unreachable" problem.

Example 12-11 points out how an iBGP peer can set NEXT_HOP to itself. However, it's also a good example of how BGP decides when to advertise routes to iBGP peers. The example follows this sequence, with the command output showing evidence of these events:

1. The example begins like the end of Example 12-10, with R1 advertising routes with R6 as the next hop, and with R3 not being able to use those routes as best routes.
2. Because R3's best routes are eBGP routes (through R4), R3 is allowed to advertise those routes to R2.
3. R1 then changes its configuration to use NEXT_HOP SELF.
4. R3 is now able to treat the routes learned from R1 as R3's best routes.
5. R3 can no longer advertise its best routes to these networks to R2, because the new best routes are iBGP routes.

Example 12-11 *R3 Advertises the 30s Networks to R2, and Then R3 Withdraws the Routes*

```
! (Step 1): At this point, R3 still believes its best route to all three prefixes
! in the 30s is through R4; as those are eBGP routes, R3 advertises all three
! routes to iBGP peer R2, as seen next.
R3# show ip bgp neighbor 2.2.2.2 advertised-routes
BGP table version is 7, local router ID is 3.3.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
```

Example 12-11 *R3 Advertises the 30s Networks to R2, and Then R3 Withdraws the Routes (Continued)*

```
   Network          Next Hop            Metric LocPrf Weight Path
*> 31.0.0.0         4.4.4.4                                0 45 678 ?
*> 32.0.0.0         4.4.4.4                                0 45 678 i
*> 32.1.1.0/24      4.4.4.4                                0 45 678 ?
Total number of prefixes 3
```

```
! (Step 2) R2 lists the number of prefixes learned from R3 next (3).
R2# show ip bgp summary ¦ begin Neighbor
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
1.1.1.1         4   123     212     210        7    0    0 03:27:59        3
3.3.3.3         4   123     213     211        7    0    0 03:28:00        3
```

```
! (Step 3) R1 now changes to use next-hop-self to peer R3.
R1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# router bgp 123
R1(config-router)# neigh 3.3.3.3 next-hop-self
```

```
! (Step 4) R3 now lists the routes through R1 as best, because the new
! NEXT_HOP is R1's update source IP address, 1.1.1.1, which is reachable by R3.
R3# show ip bgp
BGP table version is 10, local router ID is 3.3.3.3
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
*  31.0.0.0         4.4.4.4                                0 45 678 ?
*>i                 1.1.1.1             156160    100      0 678 ?
*  32.0.0.0         4.4.4.4                                0 45 678 i
*>i                 1.1.1.1                  0    100      0 678 i
*  32.1.1.0/24      4.4.4.4                                0 45 678 ?
*>i                 1.1.1.1             156160    100      0 678 ?
! (Step 5) First, note above that all three "best" routes are iBGP routes, as noted by the "i"
! immediately before the prefix. R3 only advertises "best" routes, with the added
! requirement that it must not advertise iBGP routes to other iBGP peers. As a
! result, R3 has withdrawn the routes that had formerly been sent to R2.
R3# show ip bgp neighbor 2.2.2.2 advertised-routes

Total number of prefixes 0
```

```
! The next command confirms on R2 that it no longer has any prefixes learned from
! R3.
R2# show ip bgp summary ¦ begin Neighbor
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
1.1.1.1         4   123     213     211        7    0    0 03:28:44        3
3.3.3.3         4   123     214     211        7    0    0 03:28:46        0
```

Summary of Rules for Routes Advertised in BGP Updates

The following list summarizes the rules dictating which routes a BGP router sends in its Update messages:

KEY POINT

- Send only the best route listed in the BGP table.
- To iBGP neighbors, do not advertise paths learned from other iBGP neighbors.
- To eBGP neighbors, do not advertise paths for which the neighbor's AS is already in the AS_PATH PA.
- Do not advertise suppressed or dampened routes.
- Do not advertise routes filtered via configuration.

The first two rules have been covered in some depth in this section. Chapter 13 covers the other three rules in more depth.

Building the IP Routing Table

So far, this chapter has explained how to form BGP neighbor relationships, how to inject routes into the BGP table, and how BGP routers choose which routes to propagate to neighboring routers. Part of that logic relates to how the BGP decision process selects a router's best route to each prefix, with the added restriction that the NEXT_HOP must be reachable before the route can be considered as a best route.

This section completes the last step in BGP's ultimate goal—adding the appropriate routes to the IP routing table. In its simplest form, BGP takes the already identified best BGP routes for each prefix and adds those routes to the IP routing table. However, there are some additional restrictions, mainly related to administrative distance (AD) (for eBGP and iBGP routes) and BGP synchronization (iBGP routes only). The sections that follow detail the exceptions.

Adding eBGP Routes to the IP Routing Table

Cisco IOS software uses simple logic when determining which eBGP routes to add to the IP routing table. The only two requirements are as follows:

KEY POINT

- The eBGP route in the BGP table is considered to be a "best" route.
- If the same prefix has been learned via another IGP or via static routes, the AD for BGP external routes must be lower than the ADs for other routing source(s).

By default, Cisco IOS considers eBGP routes to have AD 20, which gives eBGP routes a better (lower) AD than any other dynamic routing protocol's default AD (except for the AD 5 of EIGRP summary routes). The rationale behind the default is that eBGP-learned routes should never be

prefixes from within an AS. Under normal conditions, eBGP-learned prefixes should seldom be seen as IGP-learned routes as well, but when they are, the BGP route would win by default.

BGP sets the AD differently for eBGP routes, iBGP routes, and for local (locally injected) routes—with defaults of 20, 200, and 200, respectively. These values can be overridden in two ways, both consistent with the coverage of AD in Chapter 11:

- By using the **distance bgp** *external-distance internal-distance local-distance* BGP subcommand, which allows the simple setting of AD for eBGP-learned prefixes, iBGP-learned prefixes, and prefixes injected locally, respectively.
- By changing the AD using the **distance** {*ip-address* {*wildcard-mask*}} [*ip-standard-list* | *ip-extended-list*] BGP subcommand

Similar commands were covered in the Chapter 11 section "Preventing Suboptimal Routes by Setting the Administrative Distance." With BGP, the IP address and wildcard mask refer to the IP address used on the **neighbor** command for that particular neighbor, not the BGP RID or NEXT_HOP of the route. The ACL examines the BGP routes received from the neighbor, assigning the specified AD for any routes matching the ACL with a permit action.

Finally, a quick note is needed about the actual IP route added to the IP routing table. The route contains the exact same prefix, prefix length, and next-hop IP address as listed in the BGP table—even if the NEXT_HOP PA is an IP address that is not in a connected network. As a result, the IP forwarding process may require a recursive route lookup. Example 12-12 shows such a case on R3, where the three BGP routes each list a next hop of 1.1.1.1, which happens to be a loopback interface on R1. As you can see from Figure 12-4, R3 and R1 have no interfaces in common. The route to 1.1.1.1 lists the actual next-hop IP address to which a packet would be forwarded.

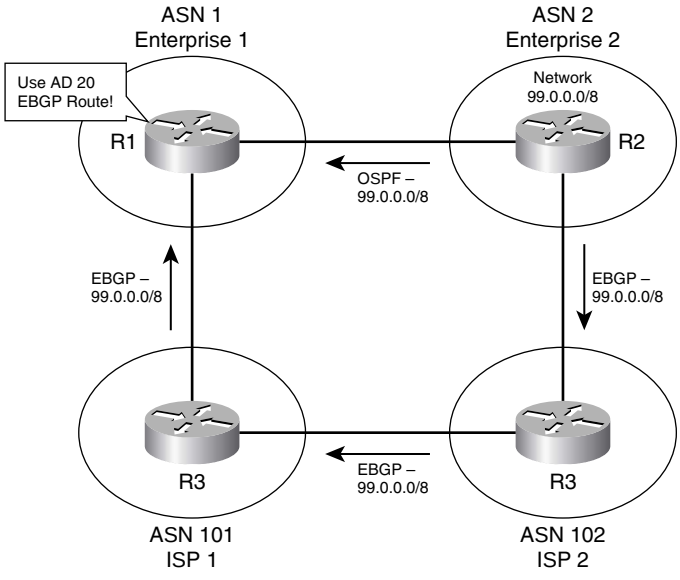
Example 12-12 *R3 Routes with Next-Hop 1.1.1.1, Requiring Recursive Route Lookup*

```
! Packets forwarded to 31.0.0.0/8 match the last route, with next-hop 1.1.1.1; R3
! then finds the route that matches destination 1.1.1.1 (the first route), finding
! the appropriate next-hop IP address and outgoing interface.
R3# show ip route | incl 1.1.1.1
D       1.1.1.1 [90/2809856] via 10.1.23.2, 04:01:44, Serial0/0/1
B       32.1.1.0/24 [200/156160] via 1.1.1.1, 00:01:00
B       32.0.0.0/8 [200/0] via 1.1.1.1, 00:01:00
B    31.0.0.0/8 [200/156160] via 1.1.1.1, 00:01:00
```

Backdoor Routes

Having a low default AD (20) for eBGP routes can cause a problem in some topologies. Figure 12-6 shows a typical case, in which Enterprise 1 uses its eBGP route to reach network 99.0.0.0 in Enterprise 2. However, the two enterprises want to use the OSPF-learned route via the leased line between the two companies.

Figure 12-6 *The Need for BGP Backdoor Routes*



R1 uses its eBGP route to reach 99.0.0.0 because eBGP has a lower AD (20) than OSPF (110). One solution would be to configure the **distance** command to lower the AD of the OSPF-learned route. However, BGP offers an elegant solution to this particular problem through the use of the **network backdoor** command. In this case, if R1 configures the **network 99.0.0.0 backdoor** router BGP subcommand, the following would occur:

- R1 would use the local AD (default 200) for the eBGP-learned route to network 99.0.0.0.
- R1 does not advertise 99.0.0.0 with BGP.

Given that logic, R1 can use a **network backdoor** command for each prefix for which R1 needs to use the private link to reach Enterprise 2. If the OSPF route to each prefix is up and working, R1 uses the OSPF (AD 110) route over the eBGP-learned (AD 200) route through the Internet. If the OSPF route is lost, the two companies can still communicate through the Internet.

Adding iBGP Routes to the IP Routing Table

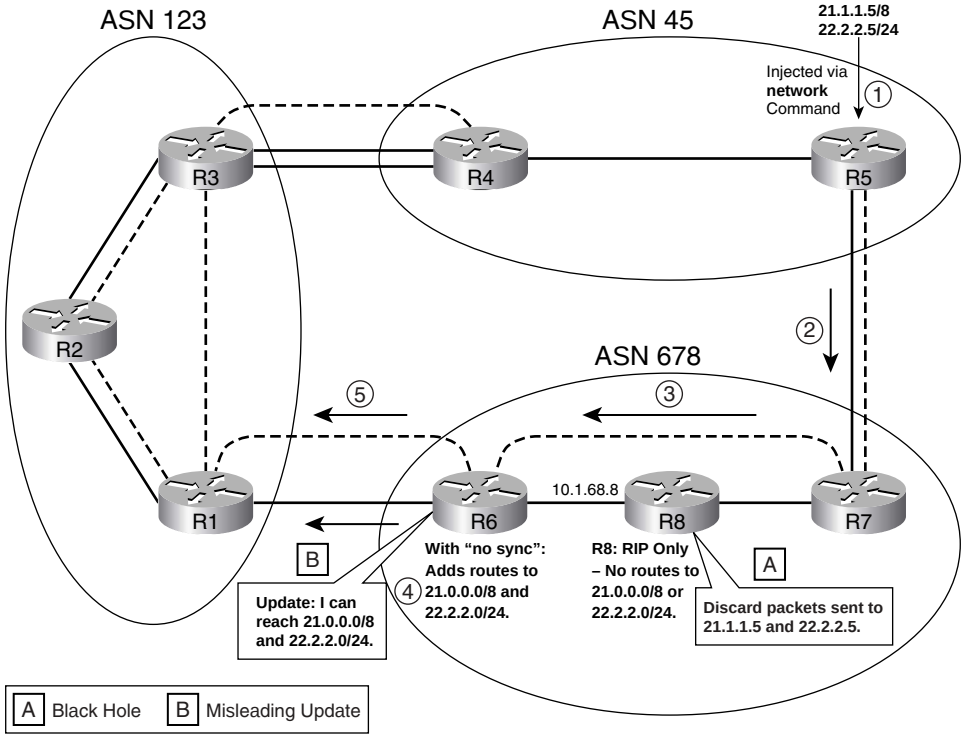
Cisco IOS has the same two requirements for adding iBGP routes to the IP routing table as it does for eBGP routes:

- The route must be the best BGP route.
- The route must be the best route (according to the AD) in comparison with other routing sources.

Additionally, for iBGP-learned routes, IOS considers the concept of *BGP synchronization*.

With BGP synchronization (often called *sync*) disabled using the **no synchronization** command, BGP uses the same logic for iBGP routes as it does for eBGP routes regarding which routes to add to the IP routing table. However, enabling BGP sync (with the **synchronization** BGP subcommand) prevents a couple of problems related to IP routing. Figure 12-7 shows the details of just such a problem. In this case, sync was inappropriately disabled in ASN 678, creating a black hole.

Figure 12-7 *Problem: Routing Black Hole Due to Not Using BGP Sync*



The following list takes a sequential view of what occurs within BGP in Figure 12-7:

1. R5 adds two prefixes (21.0.0.0/8 and 22.2.2.0/24) into its BGP table using two **network** commands.
2. R5 advertises the prefixes to R7, but does not redistribute the routes into its IGP.
3. R7 advertises the prefixes to R6.
4. R6, with synchronization disabled, considers the routes as "best," so R6 adds the routes to its routing table.
5. R6 also advertises the two prefixes to R1.

Two related problems (labeled A and B in the figure) actually occur in this case. The routing *black hole* occurs because R8 does not have a route to either of the prefixes advertised by BGP. R8 is not running BGP—a common occurrence for a router that does not directly connect to an eBGP peer. R7 did not redistribute those two prefixes into the IGP; as a result, R8 cannot route packets for those prefixes. R6, and possibly routers in AS 123, try to forward packets destined to the two prefixes through AS 678, but R8 discards the packets—hence the black hole.

The second related problem, labeled B, occurs at Step 5. R6 exacerbated the routing black-hole problem by advertising to another AS (AS 123) that it could reach the prefixes. R6 considers its routes to 21.0.0.0/8 and 22.2.2.0/24 as "best" routes in its BGP table, so R6 then advertises those routes to R1. Depending on the topology and PA settings, R1 could have considered these routes as its best routes—thereby sending packets destined for those prefixes into AS 678. (Assuming the configuration as shown in the previous examples, R1 would actually believe the 1 AS_PATH through R3 to AS 45 as the best path.)

The solutions to these problems are varied, but all the solutions result in the internal routers (for example, R8) learning the routes to these prefixes, thereby removing the black hole and removing the negative effect of advertising the route. The original solution to this problem involves the use of BGP synchronization, along with redistributing BGP routes into the IGP. However, two later solutions provide better options today:

- BGP route reflectors
- BGP confederations

The next several sections cover all of these options.

Using Sync and Redistributing Routes

BGP synchronization is best understood when considered in the context in which it was intended to be used—namely, in conjunction with the redistribution of BGP routes into the IGP. This method is seldom used by ISPs today, mainly because of the large number of routes that would be injected into the IGP. However, using BGP sync in conjunction with redistribution solves both problems related to the routing black hole.

KEY POINT

The key to understanding BGP sync is to know that redistribution solves the routing black-hole problem, and sync solves the problem of advertising a black-hole route to another AS. For example, to solve the routing black-hole problem, R7 redistributes the two prefixes into RIP (from Figure 12-7). R8 then has routes to those prefixes, solving the black-hole problem.

Sync logic on R6 controls the second part of the overall problem, regulating the conditions under which R6 advertises the prefixes to other eBGP peers (like R1). Sync works by controlling whether a BGP table entry can be considered "best"; keep in mind that a route in the BGP table

must be considered to be "best" before it can be advertised to another BGP peer. The BGP sync logic controls that decision as follows:

KEY POINT

Do not consider an iBGP route in the BGP table as "best" unless the exact prefix was learned via an IGP and is currently in the routing table.

Sync logic essentially gives a router a method to know whether the non-BGP routers inside the AS should have the ability to route packets to the prefix. Note that the route must be IGP-learned because a static route on R6 would not imply anything about what other routers (like R8) might or might not have learned. For example, using Figure 12-7 again, once R6 learns the prefixes via RIP, RIP will place the routes in its IP routing table. At that point, the sync logic on R6 can consider those same BGP-learned prefixes in the BGP table as candidates to be best routes. If chosen as best, R6 can then advertise the BGP routes to R1.

Example 12-13 shows the black hole occurring from R6's perspective, with sync disabled on R6 using the **no synchronization** BGP subcommand. Following that, the example shows R6's behavior once R7 has begun redistributing BGP routes into RIP, with sync enabled on R6.

Example 12-13 *Comparing the Black Hole (No Sync) and Solution (Sync)*

```
! R6 has a "best" BGP route to 21.0.0.0/8 through R7 (7.7.7.7), but a trace
! command shows that the packets are discarded by R8 (10.1.68.8).
R6# show ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*  21.0.0.0         172.16.16.1                            0 123 45 i
*>i                 7.7.7.7                  0    100      0 45 i
*  22.2.2.0/24      172.16.16.1                            0 123 45 i
*>i                 7.7.7.7                  0    100      0 45 i
R6# trace 21.1.1.5
Type escape sequence to abort.
Tracing the route to 21.1.1.5

  1 10.1.68.8 20 msec 20 msec 20 msec
  2 10.1.68.8 !H  *  !H
```

```
! R7 is now configured to redistribute BGP into RIP.
R7# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R7(config)# router rip
R7(config-router)# redist bgp 678 metric 3
```

```
! Next, R6 switches to use sync, and the BGP process is cleared.
R6# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R6(config)# router bgp 678
R6(config-router)# synchronization
R6(config-router)# ^Z
R6# clear ip bgp *
```

continues

Example 12-13 *Comparing the Black Hole (No Sync) and Solution (Sync) (Continued)*

```
! R6's BGP table entries now show "RIB-failure," a status code that can mean
! (as of some 12.2T IOS releases) that the prefix is known via an IGP. 21.0.0.0/8
! is shown to be included as a RIP route in R6's routing table. Note also that R6
! considers the BGP routes through R7 as the "best" routes; these are still
! advertised to R1.
R6# show ip bgp
BGP table version is 5, local router ID is 6.6.6.6
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
r  21.0.0.0         172.16.16.1                            0 123 45 i
r>i                 7.7.7.7                  0    100      0 45 i
r  22.2.2.0/24      172.16.16.1                            0 123 45 i
r>i                 7.7.7.7                  0    100      0 45 i
R6# show ip route ¦ incl 21.0.0.0
R    21.0.0.0/8 [120/4] via 10.1.68.8, 00:00:15, Serial0/0.8
! R6 considers the routes through R7 as the "best" routes; these are still
! advertised to R1, even though they are in a "RIB-failure" state.
R6# show ip bgp neighbor 172.16.16.1 advertised-routes | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
r>i21.0.0.0         7.7.7.7                  0    100      0 45 i
r>i22.2.2.0/24      7.7.7.7                  0    100      0 45 i
```

NOTE Sync includes an additional odd requirement when OSPF is used as the IGP. If the OSPF RID of the router advertising the prefix is a different number than the BGP router advertising that same prefix, then sync still does not allow BGP to consider the route to be the best route. OSPF and BGP use the same priorities and logic to choose their RIDs; however, when using sync, it makes sense to explicitly configure the RID for OSPF and BGP to be the same value on the router that redistributes from BGP into OSPF.

Disabling Sync and Using BGP on All Routers in an AS

A second method to overcome the black-hole issue is to simply use BGP to advertise all the BGP-learned prefixes to all routers in the AS. Because all routers know the prefixes, sync can be disabled safely. The downside is the introduction of BGP onto all routers, and the addition of iBGP neighbor connections between each pair of routers. (In an AS with *N* routers, N(N–1)/2 neighbor connections will be required.) With large autonomous systems, BGP performance and convergence time can degrade as a result of the large number of peers.

BGP needs the full mesh of iBGP peers inside an AS because BGP does not advertise iBGP routes (routes learned from one iBGP peer) to another iBGP peer. This additional restriction helps prevent routing loops, but it then requires a full mesh of iBGP peers—otherwise, only a subset of the iBGP peers would learn each prefix.

BGP offers two tools (confederations and route reflectors) that reduce the number of peer connections inside an AS, prevent loops, and allow all routers to learn about all prefixes. These two tools are covered next.

Confederations

An AS using BGP confederations, as defined in RFC 3065, separates each router in the AS into one of several confederation sub-autonomous systems (sub-autonomous systems). Peers inside the same sub-AS are considered to be *confederation iBGP peers*, and routers in different sub-autonomous systems are considered to be *confederation eBGP peers*.

Confederations propagate routes to all routers, without a full mesh of peers inside the entire AS. To do so, confederation eBGP peer connections act like true eBGP peers in some respects. In a single sub-AS, the confederation iBGP peers must be fully meshed, because they act exactly like normal iBGP peers—in other words, they do not advertise iBGP routes to each other. However, confederation eBGP peers act like eBGP peers in that they can advertise iBGP routes learned inside their confederation sub-AS into another confederation sub-AS.

Confederations prevent loops inside a confederation AS by using the AS_PATH PA. BGP routers in a confederation add the sub-autonomous systems into the AS_PATH as part of an AS_PATH segment called the AS_CONFED _SEQ. (The AS_PATH consists of up to four different components, called segments—AS_SEQ, AS_SET, AS_CONFED_ SEQ, and AS_CONFED_SET; see the earlier section titled “Manual Summaries and the AS_PATH Path Attribute” for more information on AS_SEQ and AS_SET.)

NOTE The terms *AS* and *sub-AS* refer to the concept of an autonomous system and sub-autonomous system. *ASN* and *sub-ASN* refer to the actual AS numbers used.

Just as the AS_SEQ and AS_SET components help prevent loops between autonomous systems, AS_CONFED_SEQ and AS_CONFED_SET help prevent loops within confederation autonomous systems. Before confederation eBGP peers can advertise an iBGP route into another sub-AS, the router must make sure the destination sub-AS is not already in the AS_PATH AS_CONFED_SEQ segment. For example, in Figure 12-8, the routers in sub-ASN 65001 learn some routes and then advertise those routes to sub-ASNs 65002 and 65003. Routers in these two sub-ASNs advertise the routes to each other. However, they never re-advertise the routes back to routers in sub-ASN 65001 due to AS_CONFED_SEQ, as shown in parentheses inside the figure.

Figure 12-8 *AS_PATH Changes in a Confederation*

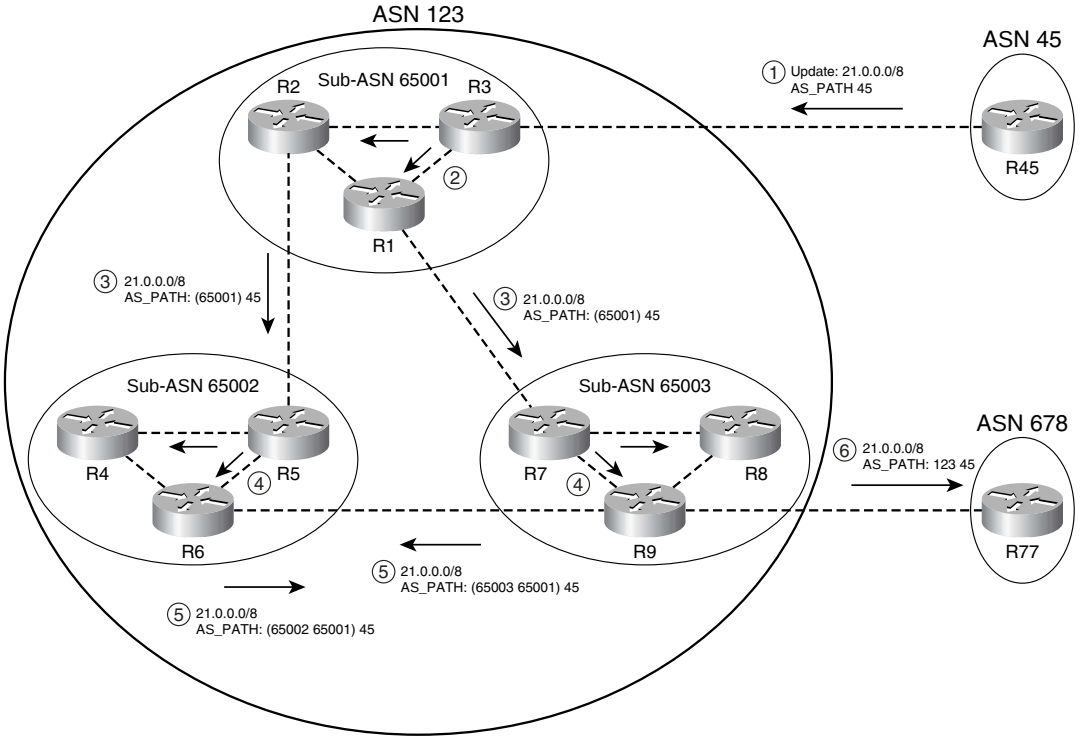


Figure 12-8 depicts a detailed example, with the steps in the following list matching the steps outlined in circled numbers in the figure:

1. 21.0.0.0/8 is injected by R45 and advertised via eBGP to AS 123. This route has an AS_PATH of 45.
2. R3 advertises the prefix via its two iBGP connections; however, due to iBGP rules inside the sub-AS, R1 and R2 do not attempt to advertise this prefix to each other.
3. Routers in sub-AS 65001 use eBGP-like logic to advertise 21.0.0.0/8 to their confederation eBGP peers, but first they inject their own sub-AS into the AS_PATH AS_CONFED_SEQ segment. (This part of the AS_PATH is displayed inside parentheses in the output of the **show ip bgp** command, as shown in the figure.)
4. The same process as in Step 2 occurs in the other two sub-autonomous systems, respectively.
5. R6 and R9 advertise the route to each other after adding their respective ASNs to the AS_CONFED_SEQ.
6. R9 advertises the prefix via a true eBGP connection after removing the sub-AS portion of the AS_PATH.

By the end of these steps, all the routers inside ASN 123 have learned of the 21.0.0.0/8 prefix. Also, ASN 678 (R77 in this case) learned of a route for that same prefix—a route that would work and would not have the black-hole effect. In fact, from ASN 678's perspective, it sees a route that appears to be through ASNs 123 and 45. Also note that routers in sub-AS 65002 and 65003 will not advertise the prefix back into sub-AS 65001 because AS 65001 is already in the confederation AS_PATH.

The choice of values for sub-ASNs 65001, 65002, and 65003 is not coincidental in this case. ASNs 64512 through 65535 are *private ASNs*, meant for use in cases where the ASN will not be advertised to the Internet or other autonomous systems. By using private ASNs, a confederation can hopefully avoid the following type of problem. Imagine that sub-AS 65003 instead used ASN 45. The AS_PATH loop check examines the entire AS_PATH. As a result, the prefixes shown in Figure 12-8 would never be advertised to sub-AS 45, and in turn would not be advertised to ASN 678. Using private ASNs would prevent this problem.

The following list summarizes the key points regarding confederations:

KEY POINT

- Inside a sub-AS, full mesh is required, because full iBGP rules are in effect.
- The confederation eBGP connections act like normal eBGP connections in that iBGP routes are advertised—as long as the AS_PATH implies that such an advertisement would not cause a loop.
- Confederation eBGP connections also act like normal eBGP connections regarding Time to Live (TTL), because all packets use a TTL of 1 by default. (TTL can be changed with the **neighbor ebgp-multihop** command.)
- Confederation eBGP connections act like iBGP connections in every other regard—for example, the NEXT_HOP is not changed by default.
- Confederation ASNs are not considered part of the length of the AS_PATH when a router chooses the best routes based on the shortest AS_PATH. (See Chapter 13 for more details on the BGP decision process.)
- Confederation routers remove the confederation ASNs from the AS_PATH in Updates sent outside the confederation; therefore, other routers do not know that a confederation was used.

Configuring Confederations

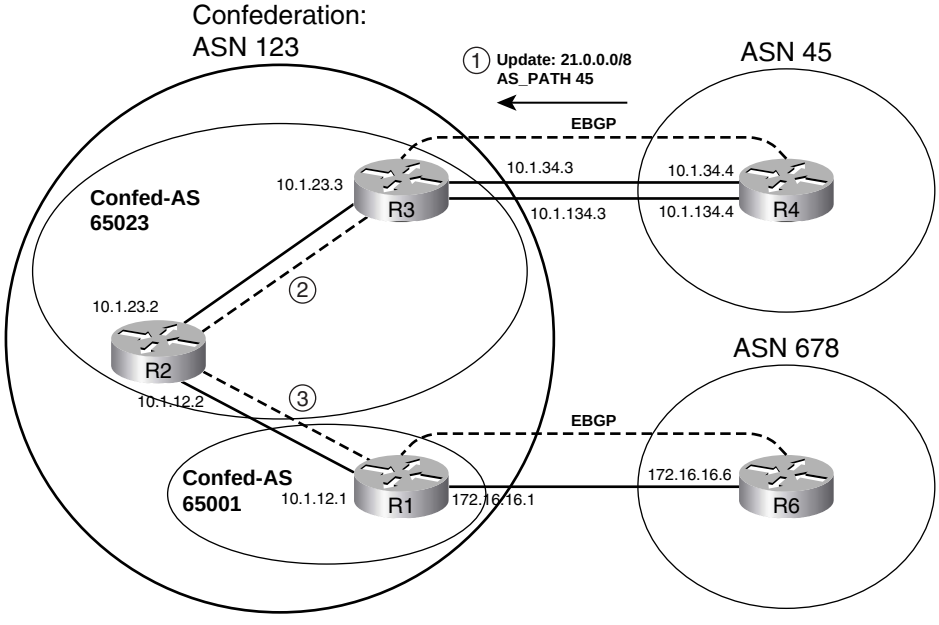
Configuring confederations requires only a few additional commands beyond those already covered in this chapter. However, migrating to use confederations can be quite painful. The problem is that the true ASN will no longer be configured on the **router bgp** command, but instead on the **bgp confederation identifier** BGP subcommand. So, BGP will simply be out of service on one or more routers while the migration occurs. Table 12-10 lists the key confederation commands, and their purpose.

Table 12-10 *BGP Subcommands Used for Confederations*

Purpose	Command
Define a router's sub-AS	**router bgp** *sub-as*
Define the true AS	**bgp confederation identifier** *asn*
To identify a neighboring AS as another sub-AS	**bgp confederation peers** *sub-asn*

Example 12-14 shows a simple configuration for the topology in Figure 12-9.

Figure 12-9 *Internetwork Topology with Confederations in ASN 123*



In this internetwork topology, R1 is in sub-AS 65001, with R2 and R3 in sub-AS 65023. In this case, R1 and R3 will not be neighbors. The following list outlines the sequence of events to propagate a prefix:

1. R3 will learn prefix 21.0.0.0/8 via eBGP from AS 45 (R4).
2. R3 will advertise the prefix via iBGP to R2.
3. R2 will advertise the prefix via confederation eBGP to R1.

Example 12-14 *Confederation Inside AS 123*

```
! R1 Configuration. Note the sub-AS in the router bgp command, and the true AS in
! the bgp confederation identifier command. Also note the neighbor ebgp-multihop
! command for confederation eBGP peer R2, as they are using loopbacks. Also, sync
```

Example 12-14 *Confederation Inside AS 123 (Continued)*

```
! is not needed now that the confederation has been created.
router bgp 65001
 no synchronization
 bgp router-id 111.111.111.111
 bgp confederation identifier 123
 bgp confederation peers 65023
 neighbor 2.2.2.2 remote-as 65023
 neighbor 2.2.2.2 ebgp-multihop 2
 neighbor 2.2.2.2 update-source Loopback1
 neighbor 2.2.2.2 next-hop-self
 neighbor 172.16.16.6 remote-as 678
```

```
! R2 Configuration. Note the bgp confederation peers 65023 command. Without it,
! R2 would think that neighbor 1.1.1.1 was a true eBGP connection, and remove
! the confederation AS_PATH entries before advertising to R1.
router bgp 65023
 no synchronization
 bgp confederation identifier 123
 bgp confederation peers 65001
 neighbor 1.1.1.1 remote-as 65001
 neighbor 1.1.1.1 ebgp-multihop 2
 neighbor 1.1.1.1 update-source Loopback1
 neighbor 3.3.3.3 remote-as 65023
 neighbor 3.3.3.3 update-source Loopback1
```

```
! R3 Configuration. Note that R3 does not need a bgp confederation peers command,
! as it does not have any confederation eBGP peers.
router bgp 65023
 no synchronization
 bgp log-neighbor-changes
 bgp confederation identifier 123
 neighbor 2.2.2.2 remote-as 65023
 neighbor 2.2.2.2 update-source Loopback1
 neighbor 2.2.2.2 next-hop-self
 neighbor 4.4.4.4 remote-as 45
 neighbor 4.4.4.4 ebgp-multihop 2
 neighbor 4.4.4.4 update-source Loopback1
```

```
! R1 has received the 21.0.0.0/8 prefix, with sub-AS 65023 shown in parentheses,
! and true AS 45 shown outside the parentheses. R1 has also learned the same
! prefix via AS 678 and R6. The route through the sub-AS is best because it is the
! shortest AS_PATH; the shortest AS_PATH logic ignores the confederation sub-autonmous systems.
R1# show ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 21.0.0.0         3.3.3.3                  0    100      0 (65023) 45 i
*                   172.16.16.6                            0 678 45 i
*> 22.2.2.0/24      3.3.3.3                  0    100      0 (65023) 45 i
*                   172.16.16.6                            0 678 45 i
```

continues

Example 12-14 *Confederation Inside AS 123 (Continued)*

```
! R6 shows its received update from R1, showing the removed sub-AS, and the
! inclusion of the true AS, AS 123.
R6# show ip bgp neighbor 172.16.16.1 received-routes | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
r  21.0.0.0         172.16.16.1                              0 123 45 i
r  22.2.2.0/24      172.16.16.1                              0 123 45 i
```

Route Reflectors

Route reflectors (RRs) achieve the same result as confederations—they remove the need for a full mesh of iBGP peers, allow all iBGP routes to be learned by all iBGP routers in the AS, and prevent loops. In an iBGP design using RRs, a partial mesh of iBGP peers is defined. Some routers are configured as RR servers; these servers are allowed to learn iBGP routes from their clients and then advertise them to other iBGP peers. The example in Figure 12-10 shows the key terms and some of the core logic used by an RR; note that only the RR server itself uses different logic, with clients and nonclients acting as normal iBGP peers.

Figure 12-10 *Basic Flow Using a Single RR, Four Clients, and Two Nonclients*

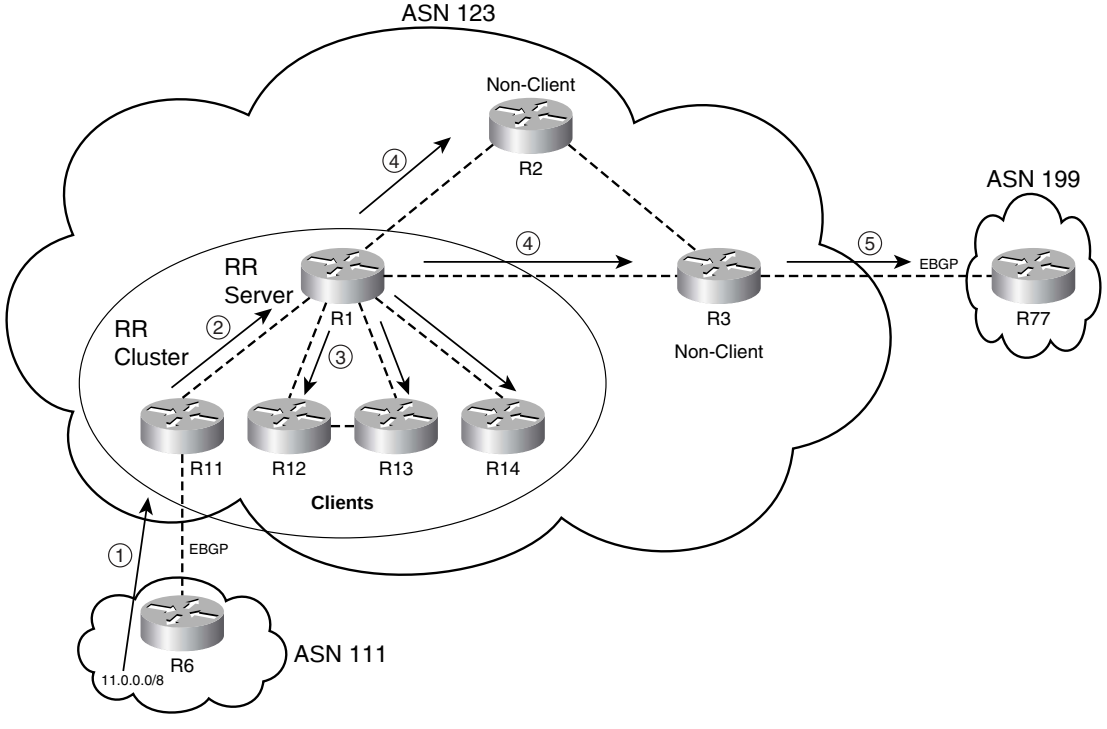


Figure 12-10 shows how prefix 11.0.0.0/8 is propagated through the AS, using the following steps:

1. R11 learns 11.0.0.0/8 using eBGP.
2. R11 uses normal iBGP rules and sends an Update to R1.
3. R1 reflects the routes by sending Updates to all other clients.
4. R1 also reflects the routes to all non-clients.
5. Nonclients use non-RR rules, sending an Update over eBGP to R77.

KEY POINT Only the router acting as the RR uses modified rules; the other routers (clients and non-clients) are not even aware of the RR, nor do they change their operating rules. Table 12-11 summarizes the rules for RR operation, which vary based on from what type of BGP peer the RR receives the prefix. The table lists the sources from which a prefix can be learned, and the types of other routers to which the RR will reflect the prefix information.

Table 12-11 *Types of Neighbors to Which Prefixes Are Reflected*

Location from Which a Prefix Is Learned	Are Routes Advertised to Clients?	Are Routes Advertised to Non-clients?
Client	Yes	Yes
Non-client	Yes	No
eBGP	Yes	Yes

The one case in which the RR does not reflect routes is when the RR receives a route from a nonclient, with the RR not reflecting that route to other nonclients. The perspective behind that logic is that RRs act like normal iBGP peers with nonclients and with eBGP neighbors—in other words, the RR does not forward iBGP-learned routes to other nonclient iBGP peers. The difference in how the RR behaves relates to when a client sends the RR a prefix, or when the RR decides to reflect a prefix to the clients.

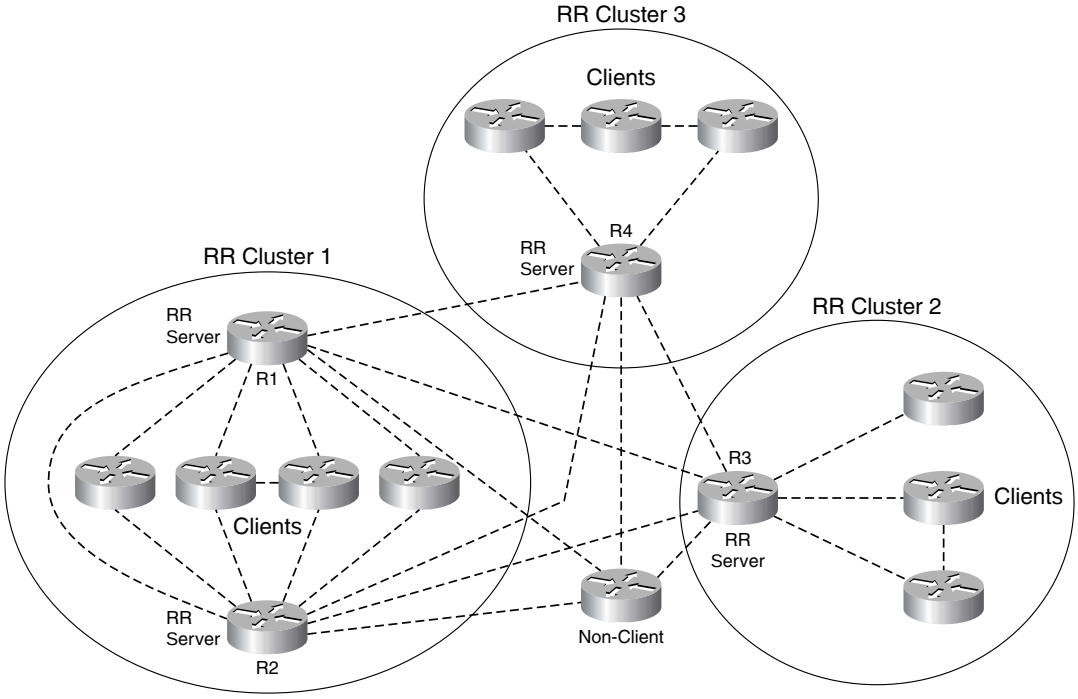
One (or more) RR servers, and their clients, create a single RR cluster. A BGP design using RRs can consist of:

- Clusters with multiple RRs in a cluster
- Multiple clusters, although using multiple clusters makes sense only when physical redundancy exists as well.

With multiple clusters, at least one RR from a cluster must be peered with at least one RR in each of the other clusters. Typically, all RRs are peered directly, creating a full mesh of RR iBGP peers among RRs. Also, if some routers are nonclients, they should be included in the full mesh of RRs.

Figure 12-11 shows the concept, with each RR fully meshed with the other RRs in other clusters, as well as with the nonclient.

Figure 12-11 *Multiple RR Clusters with Full Mesh Among RRs and Nonclients*



If you consider the logic summary in Table 12-11 compared to Figure 12-11, it appears that routing loops are not only possible but probable with this design. However, the RR feature uses several tools to prevent loops, as follows:

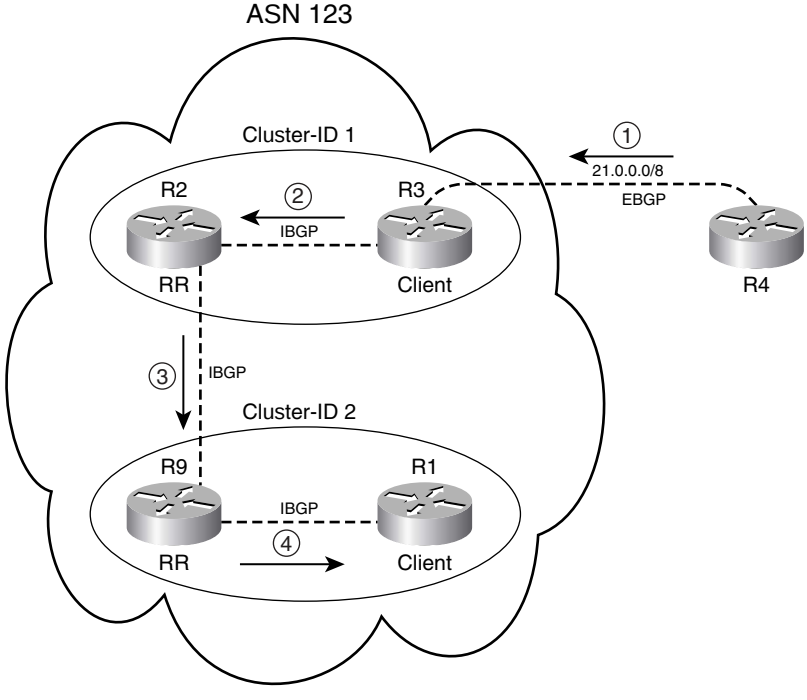
KEY POINT

- **CLUSTER_LIST**—RRs add their *cluster ID* into a BGP PA called the CLUSTER_LIST before sending an Update. When receiving a BGP Update, RRs discard received prefixes for which their cluster ID already appears. As with AS_PATH for confederations, this prevents RRs from looping advertisements between clusters.
- **ORIGINATOR_ID**—This PA lists the RID of the first iBGP peer to advertise the route into the AS. If a router sees its own BGP ID as the ORIGINATOR_ID in a received route, it does not use or propagate the route.
- **Only advertise the best routes**—RRs reflect routes only if the RR considers the route to be a "best" route in its own BGP table. This further limits the routes reflected by the RR. (It also has a positive effect compared with confederations in that an average router sees fewer, typically useless, redundant routes.)

Example 12-15 shows a simple example of using RRs. Figure 12-12 shows the modified AS 123 from the network of Figure 12-4, now with four routers. The design uses two clusters, with two RRs (R9 and R2) and two clients (R1 and R3). The following list outlines the sequence of events to propagate a prefix, as shown in Figure 12-12:

1. R3 learns prefix 21.0.0.0/8 via eBGP from AS 45 (R4).
2. R3 advertises the prefix via iBGP to R2 using normal logic.
3. R2, an RR, receiving a prefix from an RR client, reflects the route via iBGP to R9—a nonclient as far as R2 is concerned.
4. R9, an RR, receiving an iBGP route from a nonclient, reflects the route to R1, its RR client.

Figure 12-12 *Modified AS 123 Used in RR Example 12-15*



Example 12-12 *RR Configuration for AS 123, Two RRs, and Two Clients*

```
! R3 Configuration. The RR client has no overt signs of being a client; the
! process is completely hidden from all routers except RRs. Also, do not forget
! that one of the main motivations for using RRs is to allow sync to be disabled.
router bgp 123
 no synchronization
 neighbor 2.2.2.2 remote-as 123
```

continues

Example 12-12 *RR Configuration for AS 123, Two RRs, and Two Clients (Continued)*

```
 neighbor 2.2.2.2 update-source Loopback1
 neighbor 2.2.2.2 next-hop-self
 neighbor 4.4.4.4 remote-as 45
 neighbor 4.4.4.4 ebgp-multihop 255
 neighbor 4.4.4.4 update-source Loopback1
```

```
! R2 Configuration. The cluster ID would default to R2's BGP RID, but it has been
! manually set to "1," which will be listed as "0.0.0.1" in command output. R2
! designates 3.3.3.3 (R3) as a client.
router bgp 123
 no synchronization
 bgp cluster-id 1
 neighbor 3.3.3.3 remote-as 123
 neighbor 3.3.3.3 update-source Loopback1
 neighbor 3.3.3.3 route-reflector-client
 neighbor 9.9.9.9 remote-as 123
 neighbor 9.9.9.9 update-source Loopback1
```

```
! R9 Configuration. The configuration is similar to R2, but with a different
! cluster ID.
router bgp 123
 no synchronization
 bgp router-id 9.9.9.9
 bgp cluster-id 2
 neighbor 1.1.1.1 remote-as 123
 neighbor 1.1.1.1 update-source Loopback2
 neighbor 1.1.1.1 route-reflector-client
 neighbor 2.2.2.2 remote-as 123
 neighbor 2.2.2.2 update-source Loopback2
 no auto-summary
! The R1 configuration is omitted, as it contains no specific RR configuration,
! as is the case with all RR clients.
! The 21.0.0.0/8 prefix has been learned by R3, forwarded over iBGP as normal to
! R2. Then, R2 reflected the prefix to its only other peer, R9. The show ip bgp
! 21.0.0.0 command shows the current AS_PATH (45); the iBGP originator of the
! route (3.3.3.3), and the iBGP neighbor from which it was learned ("from
! 2.2.2.2"); and the cluster list, which currently has R2's cluster (0.0.0.1).
! The next output is from R9.
R9# show ip bgp 21.0.0.0
BGP routing table entry for 21.0.0.0/8, version 3
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
     2
  45
    3.3.3.3 (metric 2300416) from 2.2.2.2 (2.2.2.2)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Originator: 3.3.3.3, Cluster list: 0.0.0.1
```

```
! RR R9 reflected the prefix to its client (R1), as seen next. Note the changes
```

Example 12-12 *RR Configuration for AS 123, Two RRs, and Two Clients (Continued)*

```
! compared to R9's output, with iBGP route being learned from R9 ("from 9.9.9.9"),
! and the cluster list now including cluster 0.0.0.2, as added by R9.
R1# sho ip bgp 21.0.0.0
BGP routing table entry for 21.0.0.0/8, version 20
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Not advertised to any peer
  45
    3.3.3.3 (metric 2302976) from 9.9.9.9 (9.9.9.9)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Originator: 3.3.3.3, Cluster list: 0.0.0.2, 0.0.0.1
```

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most Cisco Press *Exam Certification Guides*, this book does not repeat information listed in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 12-12 lists some of the key RFCs for BGP.

Table 12-12 *Protocols and Standards for Chapter 12*

Topic	Standard
BGP-4	RFC 1771
BGP Confederations	RFC 3065
BGP Route Reflection	RFC 2796
MD5 Authentication	RFC 2385

Table 12-13 lists the BGP path attributes mentioned in this chapter, and describes their purpose. Chapter 13 lists additional PAs that are not covered in this chapter.

Table 12-13 *BGP PAs*

Path Attribute	Description	Characteristics
AS_PATH	Lists ASNs through which the route has been advertised	Well known Mandatory
NEXT_HOP	Lists the next-hop IP address used to reach an NLRI	Well known Mandatory
AGGREGATOR	Lists the RID and ASN of the router that created a summary NLRI	Optional Transitive
ATOMIC_AGGREGATE	Tags a summary NLRI as being a summary	Well known Discretionary
ORIGIN	Value implying from where the route was taken for injection into BGP; **i**(IGP), **e** (EGP), or **?** (incomplete information)	Well known Mandatory

Table 12-13 *BGP PAs (Continued)*

Path Attribute	Description	Characteristics
ORIGINATOR_ID	Used by RRs to denote the RID of the iBGP neighbor that injected the NLRI into the AS	Optional Nontransitive
CLUSTER_LIST	Used by RRs to list the RR cluster IDs in order to prevent loops	Optional Nontransitive

Table 12-14 *Summary: Methods to Introduce Entries into the BGP Table*

Method	Summary Description
network command	Advertises a route into BGP. Depends on the existence of the configured network/subnet in the IP routing table.
Redistribution	Takes IGP, static, or connected routes; metric (MED) assignment is not required.
Manual summarization	Requires at least one component subnet in the BGP table; options for keeping all component subnets, suppressing all from advertisement, or suppressing a subset from being advertised.
default-information originate	Requires a default route in the IP routing table, plus the **redistribute** command.
neighbor default-originate	With the optional route map, requires the route map to match the IP routing table with a permit action before advertising a default route. Without the route map, the default is always advertised.

Table 12-15 lists some of the most popular Cisco IOS commands related to the topics in this chapter.

Table 12-15 *Command Reference for Chapter 12*

Command	Command Mode and Description
aggregate-address *address mask* [**as-set**] [**summary-only**] [**suppress-map** *map-name*] [**advertise-map** *map-name*] [**attribute-map** *map-name*]	BGP mode; summarizes BGP routes, suppressing all/none/some of the component subnets
auto-summary	BGP mode; enables automatic summarization to classful boundaries of locally injected routes

continues

Table 12-15 *Command Reference for Chapter 12 (Continued)*

Command	Command Mode and Description
bgp client-to-client reflection	BGP mode; on by default, tells a RR server to reflect routes learned from a client to other clients
bgp cluster-id *cluster-id*	BGP mode; defines a nondefault RR cluster ID to a RR server
bgp confederation identifier *as-number*	BGP mode; for confederations, defines the ASN used for the entire AS as seen by other autonmous systems
bgp confederation peers *as-number* [*... as-number*]	BGP mode; for confederations, identifies which neighboring ASNs are in other confederation sub-autonmous systems
bgp log-neighbor-changes	BGP mode; on by default, it tells BGP to create log messages for significant changes in BGP operation
bgp router-id *ip-address*	BGP mode; defines the BGP router ID.
default-information originate	BGP mode; required to allow a static default route to be redistributed into BGP
default-metric *number*	BGP mode; sets the default metric assigned to routes redistributed into BGP; normally defaults to the IGP metric for each route
distance bgp *external-distance internal-distance local-distance*	BGP mode; defines the administrative distance for eBGP, iBGP, and locally injected BGP routes
neighbor {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*]	BGP mode; tells the router to add a default route to the BGP Update sent to this neighbor, under the conditions set in the optional route map
neighbor {*ip-address* | *peer-group-name*} **description** *text*	BGP mode; adds a descriptive text reference in the BGP configuration
neighbor {*ip-address* | *peer-group-name*} **ebgp-multihop** [*ttl*]	BGP mode; for eBGP peers, sets the TTL in packets sent to this peer to something larger than the default of 1
neighbor *ip-address* | *peer-group-name* **next-hop-self**	BGP mode; causes IOS to reset the NEXT_HOP PA to the IP address used as the source address of Updates sent to this neighbor
neighbor {*ip-address* | *peer-group-name*} **password** *string*	BGP mode; defines the key used in an MD5 hash of all BGP messages to this neighbor
neighbor *ip-address* **peer-group** *peer-group-name*	BGP mode; associates a neighbor's IP address as part of a peer group

Table 12-15 *Command Reference for Chapter 12 (Continued)*

Command	Command Mode and Description
neighbor *peer-group-name* **peer-group**	BGP mode; defines the name of a peer group
neighbor {*ip-address* | *peer-group-name*} **remote-as** *as-number*	BGP mode; defines the AS of the neighbor
neighbor {*ip-address* | *peer-group-name*} **shutdown**	BGP mode; administratively shuts down a neighbor, stopping the TCP connection
neighbor [*ip-address* | *peer-group-name*] **timers** *keepalive holdtime*	BGP mode; sets the two BGP timers, just for this neighbor
neighbor {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*	BGP mode; defines the source IP address used for BGP messages sent to this neighbor
network {*network-number* [**mask** *network-mask*] [**route-map** *map-tag*]	BGP mode; causes IOS to add the defined prefix to the BGP table if it exists in the IP routing table
router bgp *as-number*	Global command; defines the ASN and puts the user in BGP mode
synchronization	BGP mode; enables BGP synchronization
timers bgp *keepalive holdtime*	BGP mode; defines BGP timers for all neighbors
show ip bgp [*network*] [*network-mask*] [**longer-prefixes**] [**prefix-list** *prefix-list-name* | **route-map** *route-map-name*] [**shorter prefixes mask-length**]	Exec mode; lists details of a router's BGP table
show ip bgp injected-paths	Exec mode; lists routes locally injected into BGP
show ip bgp neighbors [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | {**paths** *regexp*} | **dampened-routes** | received prefix-filter]]	Exec mode; lists information about routes sent and received to particular neighbors
show ip bgp peer-group [*peer-group-name*] [**summary**]	Exec mode; lists details about a particular peer group
show ip bgp summary	Exec mode; lists basic statistics for each BGP peer

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

path attribute, BGP table, BGP Update, established, iBGP, eBGP, EGP, BGP, peer group, eBGP multihop, autonomous system, AS number, AS_PATH, ORIGIN, NLRI, NEXT_HOP, MULTI_EXIT_DISC, LOCAL_PREF, routing black hole, synchronization, confederation, route reflector, confederation identifier, sub-AS, route reflector server, route reflector client, route reflector nonclient, confederation AS, confederation eBGP, weight

Refer to the CD-based glossary to check your answers.

Further Reading

Routing TCP/IP, Volume II, by Jeff Doyle and Jennifer DeHaven Carrol

Cisco BGP-4 Command and Configuration Handbook, by William R. Parkhurst

Internet Routing Architectures, by Bassam Halabi

Troubleshooting IP Routing Protocols, by Zaheer Aziz, Johnson Liu, Abe Martey, and Faraz Shamim

Most every reference reached from Cisco's BGP support page at http://www.cisco.com/en/US/partner/tech/tk365/tk80/tsd_technology_support_sub-protocol_home.html. Requires a CCO username/password.

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- IP Routing
  - — BGP
  - — **show** and **debug** Commands

CHAPTER 13

BGP Routing Policies

This chapter examines the tools available to define BGP routing policies. A BGP routing policy defines the rules used by one or more routers to impact two main goals: filtering routes, and influencing which routes are considered the best routes by BGP.

BGP filtering tools are mostly straightforward, with the exception of AS_PATH filtering. AS_PATH filters use regular expressions to match the AS_PATH path attribute (PA), making the configuration challenging. Beyond that, most of the BGP filtering concepts are directly comparable to IGP filtering concepts, covered in earlier chapters.

The other topics in this chapter explain routing policies that focus on impacting the BGP decision process. The decision process itself is first outlined, followed by explanations of how each step in the process can be used to impact which routes are considered best by BGP.

"Do I Know This Already?" Quiz

Table 13-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 13-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Route Filtering and Route Summarization	1–4	
BGP Path Attributes and the BGP Decision Process	5–7	
Configuring BGP Policies	8–12	
BGP Communities	13–14	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. A BGP policy needs to be configured to filter all the /20 prefixes whose first two octets are 20.128. Which of the following answers would provide the correct matching logic for the filtering process, matching only the described subnets, and no others?

   a. **access-list 1 deny 20.128.0.0 0.0.255.255**

   b. **access-list 101 deny ip 20.128.0.0 0.0.255.255 host 255.255.240.0**

   c. **ip prefix-list 1 deny 20.128.0.0/16 eq 20**

   d. **ip prefix-list 2 deny 20.128.0.0/16 ge 20 le 20**

2. Router R1 has a working BGP implementation, advertising subnets of 1.0.0.0/8 to neighbor 2.2.2.2 (R2). A route map named fred has been configured on R1 to filter all routes in network 1.0.0.0. R1 has just added the **router bgp** subcommand **neighbor 2.2.2.2 route-map fred out**. No other commands have been used afterward. Which of the following answers, taken as the next step, would allow proper verification of whether the filter indeed filtered the routes?

   a. The **show ip bgp neighbor 2.2.2.2 advertised-routes** command on R1 will no longer list the filtered routes.

   b. The **show ip bgp neighbor 2.2.2.2 advertised-routes** command on R1 will reflect the filtered routes by listing them with a code of **r**, meaning "RIB failure."

   c. The filtering will not occur, and cannot be verified, until R1 issues a **clear ip bgp 2.2.2.2** command.

   d. None of the **show ip bgp** command options on R1 will confirm whether the filtering has occurred.

3. A router needs to match routes with AS_PATHs that include 333, as long as it is not the first ASN in the AS_PATH, while not matching AS_PATHs that include 33333. Which of the following syntactically correct commands could be a part of the complete configuration to match the correct AS_PATHs?

   a. **ip filter-list 1 permit ^.*_333_**

   b. **ip filter-list 2 permit .*333_**

   c. **ip filter-list 3 permit .*_333.*$**

   d. **ip filter-list 4 permit _333_$**

   e. **ip filter-list 5 permit ^.*_333_.*$**

4. R1 and R2 are working BGP peers. The following output of the **show ip bgp** command shows the entries learned by R1 from R2. It also shows some configuration that was added later on R1. After the appropriate **clear** command is used to make the new configuration take effect, which of the following entries should R1 have in its BGP table?

```
Network          Next Hop    Metric         LocPrf Weight Path
*>i11.10.0.0/16  2.2.2.2     4294967294        100      0 4 1 33333 10  200 44 i
*>i11.11.0.0/16  2.2.2.2     4294967294        100      0 4 1 33333 10  200 44 i
*>i11.12.0.0/16  2.2.2.2     4294967294        100      0 4 1 404   303 202    i
! New config shown next
router bgp 1
neighbor 2.2.2.2 distribute-list 1 in
access-list 1 permit 11.8.0.0 0.3.255.255
```

   a. 11.10.0.0/16
   b. 11.11.0.0/16
   c. 11.12.0.0/16
   d. 11.8.0.0/14
   e. None of the listed prefixes

5. Which of the following is true regarding BGP path attribute types?

   a. The BGP features using well-known attributes must be included in every BGP Update.
   b. Optional attributes do not have to be implemented by the programmers that are creating a particular BGP implementation.
   c. Nontransitive attributes cannot be advertised into another AS.
   d. Discretionary attributes contain sensitive information; Updates should be encoded with MD5 for privacy.

6. Which of the following items in the BGP decision tree occur after the check of the AS_PATH length?

   a. Best ORIGIN code
   b. LOCAL_PREF
   c. MED
   d. Whether the next hop is reachable

7. Which of the following steps in the BGP decision process consider a larger value to be the better value?

   a. ORIGIN

   b. LOCAL_PREF

   c. WEIGHT

   d. MED

   e. IGP metric to reach the next hop

8. Which of the following is not advertised to BGP neighbors at all?

   a. WEIGHT

   b. MED

   c. LOCAL_PREF

   d. ORIGIN

9. The following shows the output of the **show ip bgp** command on R1. Which BGP decision tree step determined which route was best? (Assume that the next hop IP address is reachable.)

```
Network          Next Hop  Metric      LocPrf Weight Path
*> 11.10.0.0/16  10.1.2.3  4294967294     100      0 4 1  33333 10 200 44 i
* i              2.2.2.2   4294967294     100      0 4 1  33333 10 200 44 i
* i              2.2.2.2   4294967294     100      0 4 1  404 505 303 202 i
```

   a. Largest Weight

   b. Best ORIGIN code

   c. Lowest MED

   d. Largest LOCAL_PREF

   e. Better neighbor type

   f. None of the answers is correct.

10. The following shows the output of the **show ip bgp** command on R1. Which BGP decision tree step determined which route was best? (Assume that the NEXT_HOP IP address is reachable.)

```
Network          Next Hop   Metric  LocPrf  Weight  Path
*  11.10.0.0/16  10.1.2.3         3     120      10  4 1  33333 10 200 44 ?
*>i              2.2.2.2          1     130      30  4 33333 10 200 44 i
* i              2.2.2.2          2     110      20  4 1  404 505 303 202 ?
```

a. Largest Weight.

b. Best ORIGIN code.

c. Lowest MED.

d. Largest LOCAL_PREF.

e. Better Neighbor Type.

f. None of the answers is correct.

11. The following exhibit lists commands that were typed using a text editor and later pasted into config mode on router R1. At the time, R1 had a working eBGP connection to several peers, including 3.3.3.3. R1 had learned of several subnets of networks 11.0.0.0/8 and 12.0.0.0/8 via neighbors besides 3.3.3.3. Once pasted, a **clear** command was issued to pick up the changes. Which of the following statements is true regarding the AS_PATH of the routes advertised by R1 to 3.3.3.3?

```
router bgp 1
 neighbor 3.3.3.3 route-map zzz out
ip prefix-list match11 seq 5 permit 11.0.0.0/8 le 32
route-map zzz permit 10
 match ip address prefix-list match11
 set as-path prepend 1 1 1
```

a. No changes would occur, because the configuration would be rejected due to syntax errors.

b. Routes to subnets inside network 11.0.0.0/8 would contain three consecutive 1s, but no more, in the AS_PATH.

c. Routes to subnets inside 11.0.0.0/8 would contain at least four consecutive 1s in the AS_PATH.

d. Routes to subnets inside 12.0.0.0/8 would have no 1s in the AS_PATH.

e. Routes to subnets inside 12.0.0.0/8 would have one additional 1 in the AS_PATH.

12. Which of the following must occur or be configured in order for BGP to mark multiple iBGP routes in the BGP table—entries for the exact same destination prefix/length—as the best routes?

    a. Inclusion of the **maximum-paths** *number* command under router BGP, with a setting larger than 1.

    b. Inclusion of the **maximum-paths ibgp** *number* command under router BGP, with a setting larger than 1.

    c. Multiple routes that tie for all BGP decision process comparisons up through checking a route's ORIGIN code.

    d. BGP cannot consider multiple routes in the BGP table for the exact same prefix as best routes.

13. Which of the following special BGP COMMUNITY values, when set, imply that a route should not be forwarded outside a confederation AS?

    a. LOCAL_AS

    b. NO_ADVERT

    c. NO_EXPORT

    d. NO_EXPORT_SUBCONFED

14. When BGP peers have set and sent COMMUNITY values in BGP Updates, which of the following is true?

    a. The BGP decision process adds a check for the COMMUNITY just before the check for the shortest AS_PATH length.

    b. The lowest COMMUNITY value is considered best by the BGP decision process, unless the COMMUNITY is set to one of the special reserved values.

    c. The COMMUNITY does not impact the BGP decision process directly.

    d. None of the other answers is correct.

Foundation Topics

Route Filtering and Route Summarization

This section focuses on four popular tools used to filter BGP routes:

- Distribution lists
- Prefix lists
- AS_PATH filter lists
- Route maps

Additionally, the **aggregate-address** command can be used to filter component subnets of a summary route. This section covers these five options. (Filtering using special BGP COMMUNITY values will be covered at the end of the chapter in the section titled "BGP Communities.")

The four main tools have the following features in common:

KEY POINT

- All can filter incoming and outgoing Updates, per neighbor or per peer group.
- Peer group configurations require Cisco IOS Software to process the routing policy against the Update only once, rather than once per neighbor.
- The filters cannot be applied to a single neighbor that is configured as part of a peer group; the filter must be applied to the entire peer group, or the neighbor must be reconfigured to be outside the peer group.
- Each tool's matching logic examines the contents of the BGP Update message, which includes the BGP PAs and network layer reachability information (NLRI).
- If a filter's configuration is changed, a **clear** command is required for the changed filter to take effect.
- The **clear** command can use the soft reconfiguration option to implement changes without requiring BGP peers to be brought down and back up.

The tools differ in what they can match in the BGP Update message. Table 13-2 outlines the commands for each tool and the differences in how they can match NLRI entries in an Update.

NOTE Throughout the book, the *wildcard mask* used in ACLs is abbreviated *WC mask*.

Table 13-2 *NLRI Filtering Tools*

KEY POINT

BGP Subcommand	Commands Referenced by neighbor Command	What Can Be Matched
neighbor distribute-list (standard ACL)	**access-list**, **ip access-list**	Prefix, with WC mask
neighbor distribute-list (extended ACL)	**access-list**, **ip access-list**	Prefix and prefix length, with WC mask for each
neighbor prefix-list	**ip prefix-list**	Exact or "first *N*" bits of prefix, plus range of prefix lengths
neighbor filter-list	**ip as-path access-list**	AS_PATH contents; all NLRI whose AS_PATHs are matched considered to be a match
neighbor route-map	**route-map**	Prefix, prefix length, AS_PATH, and/or any other PA matchable within a BGP route map

This section begins by covering filtering through the use of matching NLRI, distribute lists, prefix lists, and route maps. From there, it moves on to describe how to use BGP filter lists to match AS_PATH information to filter NLRI entries from routing updates.

Filtering BGP Updates Based on NLRI

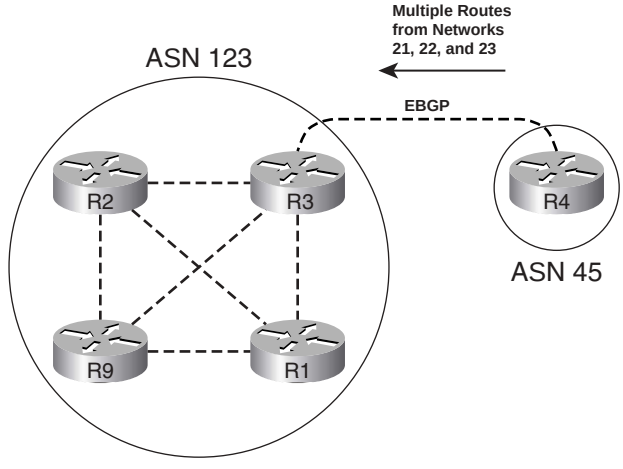
Most of the logic behind BGP distribution lists, prefix lists, and route maps has already been covered in previous chapters. For example, Chapter 11 explains the logic behind the **ip prefix-list** command, and Chapters 8 through 10 cover filtering in IGP routing protocols using the **distribute-list** command. This section shows some brief examples to cover the syntax when these methods are used with BGP, plus a few quirks unique to BGP.

One difference between BGP distribute lists and IGP distribute lists is that a BGP distribute list can use an extended ACL to match against both the prefix and the prefix length. When used with IGP filtering tools, ACLs called from distribute lists cannot match against the prefix length. Example 13-1 shows how an extended ACL matches the prefix with the source address portion of the ACL commands, and matches the prefix length (mask) using the destination address portion of the ACL commands.

The matching logic used by **prefix-list** and **route-map** commands works just the same for BGP as it does for IGPs. For example, both commands have an implied **deny** action at the end of the list, which can be overridden by matching all routes with the final entry in the **prefix-list** or **route-map** command.

Figure 13-1 shows the important portions of the network used for Example 13-1. The example shows a prefix list, distribute list, and a route map performing the same logic to filter based on NLRI. In this case, the four routers in AS 123 form a full mesh. R3 will learn a set of prefixes from AS 45, and then filter the same two prefixes (22.2.2.0/24 and 23.3.16.0/20) from being sent in Updates to each of R3's three neighbors—in each case using a different tool.

Figure 13-1 *iBGP Full Mesh in AS 123 with Routes Learned from AS 45*



Example 13-1 *Route Filtering on R3 with Route Maps, Distribution Lists, and Prefix Lists*

```
! R3 Configuration. Only the commands related to filtering are shown. Note that
! BGP Updates to R1 and R2 are filtered at this point; filtering to R9 will be
! added later in the example.
router bgp 123
 neighbor 1.1.1.1 route-map rmap-lose-2 out
 neighbor 2.2.2.2 distribute-list lose-2 out
! This ACL matches exactly for a prefix of 23.3.16.0 and 22.2.2.0, as well as
! exactly matching masks 255.255.240.0 and 255.255.255.0, respectively.
ip access-list extended lose-2
 deny   ip host 23.3.16.0 host 255.255.240.0
 deny   ip host 22.2.2.0 host 255.255.255.0
 permit ip any any
! The prefix list matches the exact prefixes and prefix lengths; the omission of
! any ge or le parameter means each line matches only that exact prefix. Also, the
! third line matches all prefixes, changing the default action to permit.
```

continues

Example 13-1 *Route Filtering on R3 with Route Maps, Distribution Lists, and Prefix Lists (Continued)*

```
ip prefix-list prefix-lose-2 seq 5 deny 22.2.2.0/24
ip prefix-list prefix-lose-2 seq 10 deny 23.3.16.0/20
ip prefix-list prefix-lose-2 seq 15 permit 0.0.0.0/0 le 32
! The route map refers to ACL lose-2, passing routes that are permitted
! by the ACL, and filtering all others. The two filtered routes are actually
! filtered by the implied deny clause at the end of the route map: Because the ACL
! matches those two prefixes with a deny action, they do not match clause 10 of the
! route map, and are then matched by the implied deny clause.
route-map rmap-lose-2 permit 10
 match ip address lose-2
! Next, R3 has seven prefixes, with the two slated for filtering highlighted.
R3# show ip bgp ¦ begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 21.0.0.0         4.4.4.4                                0 45 i
*> 22.2.2.0/24      4.4.4.4                                0 45 i
*> 23.3.0.0/20      4.4.4.4                                0 45 i
*> 23.3.16.0/20     4.4.4.4                                0 45 i
*> 23.3.32.0/19     4.4.4.4                                0 45 i
*> 23.3.64.0/18     4.4.4.4                                0 45 i
*> 23.3.128.0/17    4.4.4.4                                0 45 i
Total number of prefixes 7
! The next command shows what entries R3 will advertise to R1. Note that the
! correct two prefixes have been removed, with only five prefixes listed. The same
! results could be seen for the Update sent to R2, but it is not shown here.
R3# show ip bgp neighbor 1.1.1.1 advertised-routes | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 21.0.0.0         4.4.4.4                                0 45 i
*> 23.3.0.0/20      4.4.4.4                                0 45 i
*> 23.3.32.0/19     4.4.4.4                                0 45 i
*> 23.3.64.0/18     4.4.4.4                                0 45 i
*> 23.3.128.0/17    4.4.4.4                                0 45 i
Total number of prefixes 5
! Next, R3 adds an outbound prefix list for neighbor R9 (9.9.9.9). However,
! afterwards, R3 still believes it should send all seven prefixes to R9.
R3# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R3(config)# router bgp 123
R3(config-router)# neigh 9.9.9.9 prefix-list prefix-lose-2 out
R3(config-router)# ^Z
R3# show ip bgp neighbor 9.9.9.9 advertised-routes | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 21.0.0.0         4.4.4.4                                0 45 i
*> 22.2.2.0/24      4.4.4.4                                0 45 i
*> 23.3.0.0/20      4.4.4.4                                0 45 i
*> 23.3.16.0/20     4.4.4.4                                0 45 i
*> 23.3.32.0/19     4.4.4.4                                0 45 i
*> 23.3.64.0/18     4.4.4.4                                0 45 i
```

Example 13-1 *Route Filtering on R3 with Route Maps, Distribution Lists, and Prefix Lists (Continued)*

```
*> 23.3.128.0/17    4.4.4.4                                0 45 i
Total number of prefixes 7
! Instead of the clear ip bgp 9.9.9.9 command, which would close the BGP neighbor
! and TCP connection to R9, R3 uses the clear ip bgp 9.9.9.9 out or clear ip bgp * soft
! command to perform a soft reconfiguration. Now R3 filters the correct two prefixes.
R3# clear ip bgp 9.9.9.9 out
R3# show ip bgp neighbor 9.9.9.9 advertised-routes | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 21.0.0.0         4.4.4.4                                0 45 i
*> 23.3.0.0/20      4.4.4.4                                0 45 i
*> 23.3.32.0/19     4.4.4.4                                0 45 i
*> 23.3.64.0/18     4.4.4.4                                0 45 i
*> 23.3.128.0/17    4.4.4.4                                0 45 i
Total number of prefixes 5
```

Route Map Rules for NLRI Filtering

The overall logic used by route maps to filter NLRIs is relatively straightforward—the Update is compared to the route map and the route is filtered (or not) based on the first-matching clause. However, route maps can cause a bit of confusion on a couple of points; the next page or so points out some of the potential confusing points with regard to route maps when they are used to filter BGP routes.

Both the route map and any referenced ACL or prefix list have **deny** and **permit** actions configured, so it is easy to confuse the context in which they are used. The **route-map** command's action—either **deny** or **permit**—defines whether an NLRI is filtered (**deny**) or allowed to pass (**permit**). The **permit** or **deny** action in an ACL or prefix list implies whether an NLRI matches the **route map** clause (**permit** by the ACL/prefix list) or does not match (**deny** in the ACL/prefix list).

For example, **route-map rmap-lose-2 permit 10** from Example 13-1 matched all NLRIs except the two prefixes that needed to be filtered based on named ACL lose-2. The matched routes—all the routes that do not need to be filtered in this case—were then advertised, because clause 10 had a **permit** action configured. The route map then filtered the other two routes by virtue of the implied **deny all** logic at the end of the route map.

Alternately, the route map could have just as easily used a beginning clause of **route-map rmap-lose-2 deny 10**, with the matching logic only matching the two prefixes that needed to be filtered—in that case, the first clause would have filtered the two routes because of the **deny** keyword in the **route-map** command. Such a route map would then require a second clause that matched all routes, with a **permit** action configured. (To match all NLRI in a route map, simply omit the **match** command from that route map clause. In this case, adding just the command **route-map rmap-lose-2 20**, with no subcommands, would match all remaining routes and allow them to be advertised.)

Soft Reconfiguration

The end of Example 13-1 shows a BGP feature called *soft reconfiguration*. Soft reconfiguration allows a BGP peer to reapply its routing policies without closing a neighbor connection. To reapply the policies, Cisco IOS uses the **clear** command with either the **soft**, **in**, or **out** options, as shown in the following generic **clear** command syntax:

```
clear ip bgp {* | neighbor-address | peer-group-name} [soft [in | out]]
```

The **soft** option alone reapplies the policy configuration for both inbound and outbound policies, whereas the inclusion of the **in** or **out** keyword limits the reconfiguration to the stated direction.

Cisco IOS supports soft reconfiguration for sent Updates automatically, but BGP must be configured to support soft reconfiguration for inbound Updates. To support soft reconfiguration, BGP must remember the actual sent and received BGP Update information for each neighbor. The **neighbor** *neighbor-id* **soft-reconfiguration inbound** command causes the router to keep a copy of the received Updates from the specified neighbor. (IOS keeps a copy of sent Updates automatically.) With these Updates available, BGP can simply reapply the changed filtering policy to the Update without closing the neighbor connection.

KEY POINT Clearing the neighbor is required to pick up the changes to routing policies that impact Updates sent and received from neighbors. All such changes can be implemented using soft reconfiguration. However, for configuration changes that impact the local injection of routes into the BGP table, soft reconfiguration does not help. The reason is that soft reconfiguration simply reprocesses Updates, and features that inject routes into BGP via the **redistribute** or **network** commands are not injected based on Update messages.

Comparing BGP Prefix Lists, Distribute Lists, and Route Maps

Prefix lists and distribute lists both use their matching logic on the BGP Update's NLRI. However, a prefix list allows more flexible matching of the prefix length because it can match a range of prefixes that extends to a maximum length of less than 32. For example, the command **ip prefix-list test1 permit 10.0.0.0/8 ge 16 le 23** matches a range of prefix lengths, but the same logic using an ACL as a distribute list takes several more lines, or a tricky wildcard mask.

For many BGP filtering tasks, route maps do not provide any benefit over prefix lists, ACLs, and AS_PATH filter lists. If the desired policy is only to filter routes based on matching prefixes/lengths, a route map does not provide any additional function over using a distribute list or prefix list directly. Similarly, if the goal of the policy is to filter routes just based on matching with an AS_PATH filter, the route map does not provide any additional function as compared to calling an AS_PATH filter directly using the **neighbor filter-list** command.

However, only route maps can provide the following two functions for BGP routing policy configurations:

- Matching logic that combines multiple of the following: prefix/length, AS_PATH, or other BGP PAs
- The setting of BGP PA's for the purpose of manipulating BGP's choice of which route to use

Many of the features for manipulating the choice of best routes by BGP, as covered later in this chapter, use route maps for that purpose.

Filtering Subnets of a Summary Using the aggregate-address Command

Manual BGP route summarization, using the **aggregate-address** BGP router subcommand, provides the flexibility to allow none, all, or a subset of the summary's component subnets to be advertised out of the BGP table. By allowing some and not others, the **aggregate-address** command can in effect filter some routes. The filtering options on the **aggregate-address** command are as follows:

KEY POINT

- Filtering all component subnets of the summary from being advertised, by using the **summary-only** keyword
- Advertising all the component subnets of the summary, by *omitting* the **summary-only** keyword
- Advertising some and filtering other component subnets of the summary, by omitting the **summary-only** keyword and referring to a **route-map** using the **suppress-map** keyword

The logic behind the **suppress-map** option can be a little tricky. This option requires reference to a route map, with any component subnets matching a route map **permit** clause being *suppressed*—in other words, routes permitted by the route map are filtered and not advertised. The router does not actually remove the suppressed route from its local BGP table; however, it does suppress the advertisement of those routes.

Example 13-2 shows how the **suppress-map** option works, with a summary of 23.0.0.0/8, and a goal of allowing all component subnets to be advertised except 23.3.16.0/20.

Example 13-2 *Filtering Routes Using the* **aggregate-address suppress-map** *Command*

```
! The first command below lists all BGP routes in network 23.
R3# sh ip bgp neigh 1.1.1.1 advertised-routes | include 23
*> 23.3.0.0/20      4.4.4.4                                0 45 i
*> 23.3.16.0/20     4.4.4.4                                0 45 i
*> 23.3.32.0/19     4.4.4.4                                0 45 i
```

continues

Example 13-2 *Filtering Routes Using the* **aggregate-address suppress-map** *Command (Continued)*

```
*> 23.3.64.0/18     4.4.4.4                                0 45 i
*> 23.3.128.0/17    4.4.4.4                                0 45 i
*> 23.4.0.0/16      4.4.4.4                                0 45 678 i
! The ACL below matches 23.3.16.0/20 with a permit clause, and denies all other
! routes (default). The route-map uses a permit clause and references
! access-list permit-1. The logic means that the one route permitted by the ACL
! will be suppressed. Note also that the summary-only keyword was not used
! on the aggregate-address command, allowing the subnets to also be advertised.
ip access-list extended permit-1
 permit ip host 23.3.16.0 host 255.255.240.0
!
route-map suppress-1 permit 10
 match ip address permit-1
!
router bgp 123
 aggregate-address 23.0.0.0 255.0.0.0 as-set suppress-map suppress-1
! Below, R3 (after a clear ip bgp * soft command) no longer advertises the route.
R3# sh ip bgp neigh 1.1.1.1 advertised-routes ¦ include 23.3.16.0
R3#
! Note the "s" on the left side of the show ip bgp command output for the
! suppressed route. The route remains in the table; it is simply no longer
! advertised outside the router.
R3# sh ip bgp neigh 1.1.1.1 advertised-routes ¦ include 23
*> 23.3.0.0/20      4.4.4.4                                0 45 i
s> 23.3.16.0/20     4.4.4.4                                0 45 i
*> 23.3.32.0/19     4.4.4.4                                0 45 i
*> 23.3.64.0/18     4.4.4.4                                0 45 i
*> 23.3.128.0/17    4.4.4.4                                0 45 i
*> 23.4.0.0/16      4.4.4.4                                0 45 678 i
```

Filtering BGP Updates by Matching the AS_PATH PA

To filter routes by matching the AS_PATH PA, Cisco IOS uses AS_PATH filters. The overall configuration structure is very similar to BGP distribute lists and prefix lists, with the matching logic specified in a list, and the logic being applied with a **neighbor** command. The main two steps are as follows:

1. Configure the AS_PATH filter using the **ip as-path access-list** *number* {**permit** | **deny**} *regex* command.
2. Enable the AS_PATH filter using the **neighbor** *neighbor-id* **filter-list** *as-path-filter-number* {**in** | **out**} command.

Based on these commands, Cisco IOS examines the AS_PATH PA in the sent or received Updates for the stated neighbor. NLRI whose AS_PATHs match with a **deny** action are filtered.

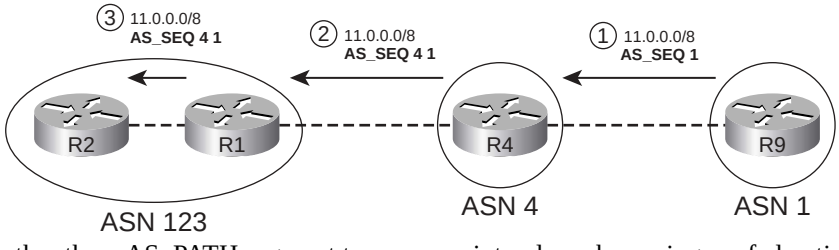
AS_PATH filters use *regular expressions* (abbreviated *regex*) to apply powerful matching logic to the AS_PATH. To match the AS_PATH contents, the regex need to match values, delimiters, and other special characters. For instance, the AS_PATH itself has several components, called *segments*, which, when present, require slightly different matching logic within a regex. The next few sections take a closer look at both regex and AS_PATHs, followed by some examples of using AS_PATH filters.

The BGP AS_PATH and AS_PATH Segment Types

RFC 1771 describes four types of AS_PATH segments held inside the AS_PATH PA (see Table 13-3). The most common segment is called AS_SEQUENCE, which is an ordered list of all the autonomous systems through which the route has passed. The AS_SEQUENCE segment lists the most recently added ASN as the first ASN; this value is also the leftmost entry when looking at **show** commands, and is considered to be the first ASN for the regex matching logic.

Because the most recently added ASN is the first ASN in the AS_SEQUENCE segment, the process of adding the ASN before advertising routes to eBGP peers is called *AS_PATH prepending*. For example, Figure 13-2 shows a sample network in which a route is injected inside AS 1, advertised to AS 4, and then advertised to AS 123.

Figure 13-2 *AS_PATH (AS_SEQUENCE) Prepending*



The other three AS_PATH segment types come into play when using confederations and route summarization, as described in Chapter 12. Table 13-3 lists and briefly describes all four types.

Table 13-3 *AS_PATH Segment Types*

KEY POINT

Component	Description	Delimiters Between ASNs	Character Enclosing the Segment
AS_SEQUENCE	An ordered list of ASNs through which the route has been advertised	Space	None
AS_SET	An unordered list of ASNs through which the route has been advertised	Comma	{}

continues

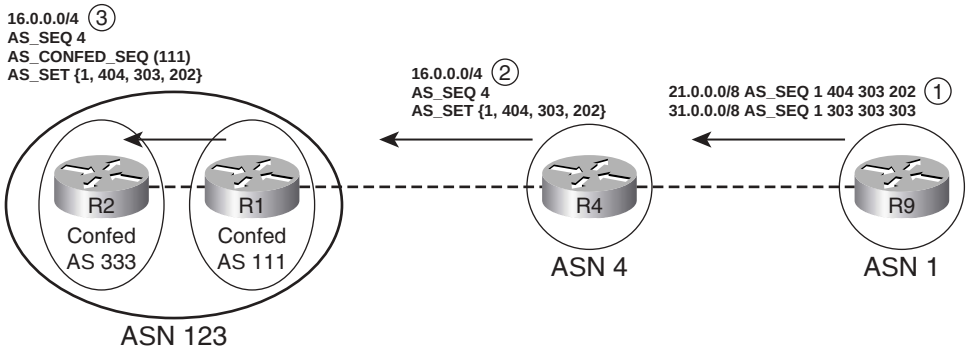
Table 13-3 *AS_PATH Segment Types (Continued)*

Component	Description	Delimiters Between ASNs	Character Enclosing the Segment
AS_CONFED_SEQ[1]	Like AS_SEQ, but holds only confederation ASNs	Space	()
AS_CONFED_SET[1]	Like AS_SET, but holds only confederation ASNs	Comma	{}

[1] Not advertised outside the confederation.

Figure 13-3 shows an example of AS_SET in which R4 summarizes some routes using the **aggregate-address… as-set** command. As a result of including the **as-set** keyword, R4 creates an AS_SET segment in the AS_PATH of the aggregate route. Note that the AS_SET segment is shown in brackets, and it is listed in no particular order. These facts are all important to the process of AS_PATH filtering.

Figure 13-3 *AS_SET and AS_CONFED_SEQ Example*



Also note the addition of the AS_CONFED_SEQ segment by R1 in Figure 13-3. Confederation ASNs are used to prevent loops inside the confederation; because these ASNs will be removed before advertising the route outside the full AS, the confederation ASNs are kept inside a different segment—the AS_CONFED_SEQ segment. Finally, if a route is aggregated inside a confederation, the AS_CONFED_SET segment holds the confederation ASNs with the same logic as used by the AS_SET segment type, but keeps them separate for easy removal before advertising the routes outside the confederation. Example 13-3 provides sample **show ip bgp** command output showing an AS_SET and AS_CONFED_SEQ. The output shows R2's AS_PATH for the route shown in Figure 13-3.

Example 13-3 *AS_PATH on R2: AS_CONFED_SEQ, AS_SEQUENCE, and AS_SET*

```
! The AS_CONFED_SEQ is (111), enclosed in parentheses. The AS_SEQUENCE only
! contains 4, with no enclosing characters. The AS_SET created by R4 when
! summarizing 16.0.0.0/4 is {1,404,303,202}, enclosed in brackets.
R2# show ip bgp | include 16.0.0.0
*> 16.0.0.0/4       10.1.14.4                0    100      0 (111) 4 {1,404,303,202} i
```

Using Regular Expressions to Match AS_PATH

A Cisco IOS AS_PATH filter has one or more configured lines, with each line requiring a regex. The logic is then applied as follows:

KEY POINT

1. The regex of the first line in the list is applied to the AS_PATH of each route.
2. For matched NLRI, the NLRI is passed or filtered based on that AS_PATH filter's configured **permit** or **deny** action.
3. For unmatched NLRI, Steps 1 and 2 are repeated, using the next line in the AS_PATH filter, analyzing all NLRI yet to be matched by this list.
4. Any NLRI not matched explicitly is filtered.

Regex contain literal strings as well as metacharacters. The *metacharacters* allow matching using wildcards, matches for different delimiters, and other special operations. Table 13-4 lists the regex metacharacters that are useful for IOS AS_PATH filters.

Table 13-4 *Regex Metacharacters Useful for AS_PATH Matching*

KEY POINT

Metacharacter	Meaning
^	Start of line
$	End of line
|	Logical OR applied between the preceding and succeeding characters[1]
_	Any delimiter: blank, comma, start of line, or end of line[2]
.	Any single character
?	Zero or one instances of the preceding character
*****	Zero or more instances of the preceding character
+	One or more instances of the preceding character

continues

Table 13-4 *Regex Metacharacters Useful for AS_PATH Matching (Continued)*

Metacharacter	Meaning
(*string***)**	Parentheses combine enclosed string characters as a single entity when used with ?, *, or +
[*string***]**	Creates a wildcard for which any of the single characters in the string can be used to match that position in the AS_PATH

[1] If preceded by a value in parentheses, the logic applies to the preceding string listed inside the parentheses, and not just to the preceding character.

[2] This character is an underscore.

When the regular expression is applied to a BGP route, Cisco IOS searches the AS_PATH for the first instance of the first item in the regex; from that point forward, it processes the rest of the AS_PATH sequentially. For example, consider two routes, one with an AS_PATH with only an AS_SEQ, set to 12 34 56, and another route with an AS_SEQ of 78 12 34 56. A regular expression of **12_34_56** matches both routes, because IOS looks for the first occurrence of AS 12, and then searches sequentially. However, a regular expression of **^12_34_56** would match only the first route. The second AS_PATH (78 12 34 56) would not match because the regex would immediately match on the ^ (start of line), then search sequentially—finding AS 78 next, which does not match the regex.

Although Table 13-4 provides a useful reference, Table 13-5 provides a number of examples of using these metacharacters, with explanations of what they match. Take special note of the wording in the explanations. Phrases like “ASN 303” and “ASN beginning with 303” differ in that the first phrase means exactly 303, and not 3031, 30342, and so on, whereas the second phrase would match any of these values.

Table 13-5 *Example AS_PATH Regex and Their Meanings*

KEY POINT

Example Regex	What Type of AS_PATH It Would Match
.*	All AS_PATHs (useful as a final match to change the default from **deny** to **permit**).
^$	Null (empty)—used for NLRIs originated in the same AS.
^123$	An AS_PATH with only one AS, ASN 123.
^123	An AS_PATH whose first ASN begins with or is 123; includes 123, 1232, 12354, and so on.
^123.	An AS_PATH whose first ASN is one of two things: a four-digit number that begins with 123, or a number that begins with ASN 123 and is followed by a delimiter before the next ASN. (It does not match an AS_PATH of only ASN 123, because the period does not match the end-of-line.)

Table 13-5 *Example AS_PATH Regex and Their Meanings (Continued)*

Example Regex	What Type of AS_PATH It Would Match
^123+	An AS_PATH whose first ASN begins with 123, with 1233, or is 12333. For example, it includes ASNs 1231 and 12331 because it does not specify what happens after the +.
^123+_	An AS_PATH whose first ASN is one of three numbers: 123, 1233, or 12333. It does not match 1231 and 12331, for example, because it requires a delimiter after the last **3**.
^123*	An AS_PATH whose first ASN begins with 12, 123, or 1233, or is 12333. Any character can follow these values, because the regex does not specify anything about the next character. For example, 121 would match because the * can represent 0 occurrences of “3”. 1231 would match with * representing 1 occurrence of 3.
^123*_	An AS_PATH whose first ASN begins with 12, 123, or 1233, or is 12333. It does not include matches for 121, 1231, and 12331, because the next character must be a delimiter.
^123?	An AS_PATH whose first ASN begins with either 12 or 123.
^123_45$	An AS_PATH with two autonomous systems, beginning with 123 and ending with 45.
^123_.*_45$	An AS_PATH beginning with AS 123 and ending in AS 45, with at least one other AS in between.
^123_.*45	An AS_PATH beginning with AS 123, with zero or more intermediate ASNs and delimiters, and ending with any AS whose last two digits are 45 (including simply AS 45).
(^123_45$)\|(^123_.*_45$)	An AS_PATH beginning with 123 and ending with AS 45, with zero or more other ASNs between the two.
^123_45$\|^123_.*_45$	(Note: this is the same as the previous example, but without the parentheses.) Represents a common error in attempting to match AS_PATHs that begin with ASN 123 and end with ASN 45. The problem is that the | is applied to the previous character ($) and next character (^), as opposed to everything before and after the |.
^123(_[0..9]+)*_45	Another way to match an AS_PATH beginning with 123 and ending with AS 45.
^{123	The AS_PATH begins with an AS_SET or AS_CONFED_SET, with the first three numerals of the first ASN being 123.
[(]303.*[)]	Find the AS_CONFED_SEQ, and match if the first ASN begins with 303.

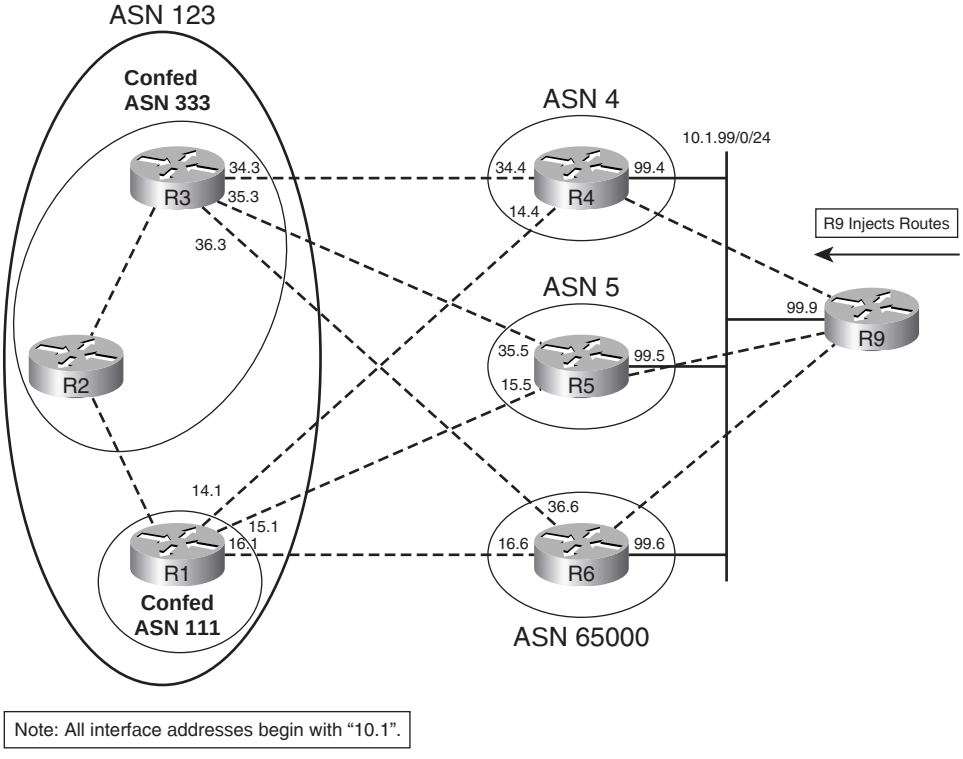
Example: Matching AS_PATHs Using AS_PATH Filters

NLRI filtering with AS_PATH filters uses two commands:

```
ip as-path access-list access-list-number {permit | deny} as-regexp
neighbor {ip-address | peer-group-name} filter-list access-list-number {in | out}
```

Figure 13-4 shows a sample internetwork used in many of the upcoming examples, two of which show the use of AS_PATH filtering.

Figure 13-4 *Network Used for AS_PATH Filter Examples*



Example 13-4 shows an AS_PATH filter in which routes are filtered going from R4 to R3. Filtering the outbound Update on R4 will be shown first, followed by filtering of inbound Updates on R3. In both cases, the goal of the filter is as follows:

> Filter routes in R4's BGP table whose ASN begins with AS 1, has three additional ASNs of any value, and ends with ASN 44.

The two NLRIs matching these criteria are 11.0.0.0/8 and 12.0.0.0/8.

Example 13-4 *AS_PATH Filtering of Routes Sent from R4 to R3*

```
! R4 learned its best routes to 11.0.0.0/8 and 12.0.0.0/8 from R9 (10.1.99.9), plus
! two other routers. Only the routes learned from R9, with NEXT_HOP 10.1.99.9,
! match the AS_PATH criteria for this example.
R4# show ip bgp
BGP table version is 9, local router ID is 4.4.4.4
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
*  11.0.0.0         10.1.14.1                              0 123 5 1 33333 10 200 44 i
*                   10.1.34.3                              0 123 5 1 33333 10 200 44 i
*>                  10.1.99.9                0             0 1 33333 10 200 44 i
*  12.0.0.0         10.1.14.1                              0 123 5 1 33333 10 200 44 i
*                   10.1.34.3                              0 123 5 1 33333 10 200 44 i
*>                  10.1.99.9                0             0 1 33333 10 200 44 i
! lines omitted for brevity
! R4 currently advertises four routes to R3, as shown next.
R4# show ip bgp neighbor 10.1.34.3 advertised-routes | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 11.0.0.0         10.1.99.9                0             0 1 33333 10 200 44 i
*> 12.0.0.0         10.1.99.9                0             0 1 33333 10 200 44 i
*> 21.0.0.0         10.1.99.9                0             0 1 404 303 202 i
*> 31.0.0.0         10.1.99.9                0             0 1 303 303 303 i
! R4's new AS_PATH filter is shown next. The first line matches AS_PATHs beginning
! with ASN 1, and ending in 44, with three ASNs in between. The second line matches all
! other AS_PATHs, with a permit action—essentially a permit all at the end. The
! list is then enabled with the neighbor filter-list command, for outbound Updates
! sent to R3 (10.1.34.3).
ip as-path access-list 34 deny ^1_.*_.*_.*_44$
ip as-path access-list 34 permit .*
router bgp 4
neighbor 10.1.34.3 filter-list 34 out
! After soft reconfiguration, R4 no longer advertises the routes to networks
! 11 and 12.
R4# clear ip bgp * soft
R4# show ip bgp neighbor 10.1.34.3 advertised-routes | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 21.0.0.0         10.1.99.9                0             0 1 404 303 202 i
*> 31.0.0.0         10.1.99.9                0             0 1 303 303 303 i
Total number of prefixes 2
```

```
! Not shown: R4's neighbor filter-list command is now removed, and soft
! reconfiguration used to restore the Updates to their unfiltered state.
! R3 lists its unfiltered received Update from R4. Note that R4's ASN was added
! by R4 before sending the Update.
```

continues

Example 13-4 *AS_PATH Filtering of Routes Sent from R4 to R3 (Continued)*

```
R3# show ip bgp neighbor 10.1.34.4 received-routes | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*  11.0.0.0         10.1.34.4                              0 4 1 33333 10 200 44 i
*  12.0.0.0         10.1.34.4                              0 4 1 33333 10 200 44 i
*  21.0.0.0         10.1.34.4                              0 4 1 404 303 202 i
*  31.0.0.0         10.1.34.4                              0 4 1 303 303 303 i
! R3 uses practically the same AS_PATH filter, except that it must look for ASN
! 4 as the first ASN.
ip as-path access-list 34 deny ^4_1_.*_.*_.*_44$
ip as-path access-list permit .*
router bgp 333
 neighbor 10.1.34.4 filter-list 34 in
! The show ip as-path-access-list command shows the contents of the list.
R3# show ip as-path-access-list 34
AS path access list 34
    deny ^4_1_.*_.*_.*_44$
    permit .*
! To test the logic of the regex, the show ip bgp command can be used, with the
! pipe (|) and the include option. That parses the command output based on the
! regex at the end of the show command. However, note that some things matchable
! using an AS_PATH filter are not in the show command output—for example, the
! beginning or end of line cannot be matched with a ^ or $, respectively.
! These metacharacters must be omitted for this testing trick to work.
R3# show ip bgp neighbor 10.1.34.4 received-routes | include 4_1_.*_.*_.*_44
*  11.0.0.0         10.1.34.4                              0 4 1 33333 10 200 44 i
*  12.0.0.0         10.1.34.4                              0 4 1 33333 10 200 44 i
! After a clear, it first appears that the routes were not filtered, as they
! still show up in the output below.
R3# clear ip bgp * soft
R3# show ip bgp neighbor 10.1.34.4 received-routes | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*  11.0.0.0         10.1.34.4                              0 4 1 33333 10 200 44 i
*  12.0.0.0         10.1.34.4                              0 4 1 33333 10 200 44 i
*  21.0.0.0         10.1.34.4                              0 4 1 404 303 202 i
*  31.0.0.0         10.1.34.4                              0 4 1 303 303 303 i
! However, R3 does not show the routes shown in the received Update from R4 in
! the BGP table; the routes were indeed filtered.
R3# show ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*  11.0.0.0         10.1.36.6                              0 65000 1 33333 10 200 44 i
* i                 10.1.15.5                0    100      0 (111) 5 1 33333 10 200 44 i
*>                  10.1.35.5                              0 5 1 33333 10 200 44 i
*  12.0.0.0         10.1.36.6                              0 65000 1 33333 10 200 44 i
* i                 10.1.15.5                0    100      0 (111) 5 1 33333 10 200 44 i
*>                  10.1.35.5                              0 5 1 33333 10 200 44 i
! lines omitted for brevity
```

The explanations in Example 13-4 cover most of the individual points about using filter lists to filter NLRI based on the AS_PATH. The example also depicts a couple of broader issues regarding the Cisco IOS BGP **show** commands:

KEY POINT

- The **show ip bgp neighbor** *neighbor-id* **advertised-routes** command displays the routes actually sent—in other words, this command reflects the effects of the filtering by omitting the filtered routes from the output.
- The **show ip bgp neighbor** *neighbor-id* **received-routes** command displays the routes actually received from a neighbor, never omitting routes from the output, even if the router locally filters the routes on input.
- Output filter lists are applied before the router adds its own ASN to the AS_PATH. (See Example 13-4's AS_PATH filter on R4 for an example.)

There are also a couple of ways to test regex without changing the routing policy. Example 13-4 showed one example using the following command:

```
show ip bgp neighbor 10.1.34.4 received-routes | include 4_1_.*_.*_.*_44
```

This command parses the entire command output using the regex after the **include** keyword. However, note that this command looks at the ASCII text of the command output, meaning that some special characters (like beginning-of-line and end-of-line characters) do not exist. For example, Example 13-4 left out the caret (^) in the regex, because the text output of the **show** command does not include a ^.

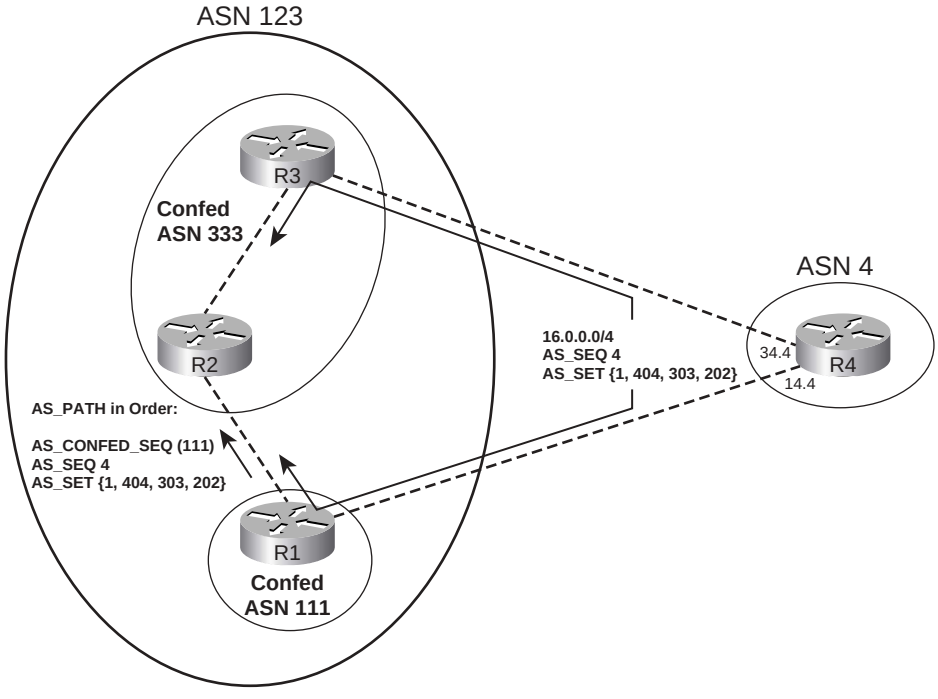
The other method to test a regex is to use the **show ip bgp regexp** *expression* command. This command parses the AS_PATH variables in a router's BGP table, including all special characters, allowing all aspects of the regex to be tested. However, the **regexp** option of the **show ip bgp** command is not allowed with the **received-routes** or **advertised-routes** option.

While Example 13-4 shows BGP using the **neighbor filter-list** command, the AS_PATH filter list can also be referenced in a route map using the **match as-path** *list-number* command. In that case, the route map then can be called using the **neighbor route-map** command.

Matching AS_SET and AS_CONFED_SEQ

Example 13-5 shows how to use a BGP filter list to match the AS_SET and AS_CONFED_SEQ segment types. Figure 13-5 depicts the specifics of the example. In this case, R4 summarizes 16.0.0.0/4, creating an AS_SET entry for the summary, and advertising it to R1 and R3. R1 and R3 in turn advertise the route to R2; R1's route includes an AS_CONFED_SEQ, because R1 and R2 are confederation eBGP peers.

Figure 13-5 *Generating AS_SET and AS_CONFED_SEQ*



Example 13-5 shows two different example filters, as follows:

- Filtering routes with ASN 303 anywhere inside the AS_SET
- Filtering based on the AS_CONFED_SEQ of only ASN 111 to begin the AS_PATH

Example 13-5 *AS_PATH Filtering of Routes Sent from R4 to R3*

```
! The next command shows R2's BGP table before filtering is enabled. R2 has five
! routes with AS_CONFED_SEQ of (111), all learned from R1. R2 also learned the
! same NLRI from R3, with the related AS_PATH not including the beginning
! AS_CONFED_SEQ of (111), because R3 is in the same confederation sub-AS as R2.
R2# sh ip bgp ¦ begin Network
   Network          Next Hop            Metric LocPrf Weight Path
* i11.0.0.0         10.1.35.5                0    100      0 5 1 33333 10 200 44 i
*>                  10.1.15.5                0    100      0 (111) 5 1 33333 10 200 44 i
* i12.0.0.0         10.1.35.5                0    100      0 5 1 33333 10 200 44 i
*>                  10.1.15.5                0    100      0 (111) 5 1 33333 10 200 44 i
* i16.0.0.0/4       10.1.34.4                0    100      0 4 {1,404,303,202} i
*>                  10.1.14.4                0    100      0 (111) 4 {1,404,303,202} i
* i21.0.0.0         10.1.34.4                0    100      0 4 1 404 303 202 i
*>                  10.1.15.5                0    100      0 (111) 5 1 404 303 202 i
```

Example 13-5 *AS_PATH Filtering of Routes Sent from R4 to R3 (Continued)*

```
* i31.0.0.0         10.1.34.4                0    100      0 4 1 303 303 303 i
*>                  10.1.15.5                0    100      0 (111) 5 1 303 303 303 i
! R2 will use AS_PATH access-list 1 to find routes that begin with AS_CONFED_SEQ
! of 111. Note that the "(" must be matched by enclosing it in square brackets, as
! the "(" itself and the ")" are metacharacters, and would otherwise be
! interpreted as a metacharacter. Without the "[(]" to begin the regex, the
! AS_PATH filter would not match.
R2# show ip as-path-access-list 1
AS path access list 1
    deny ^[(]111
    permit .*
! R2 filters incoming routes from both peers, and performs a soft reconfig.
R2(config)# router bgp 333
R2(config-router)# neigh 1.1.1.1 filter-list 1 in
R2(config-router)# neigh 3.3.3.3 filter-list 1 in
R2# clear ip bgp * soft
! Now all routes with AS_CONFED_SEQ of 111 beginning the AS_PATH are gone.
R2# sh ip bgp ¦ begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*>i11.0.0.0         10.1.35.5                0    100      0 5 1 33333 10 200 44 i
*>i12.0.0.0         10.1.35.5                0    100      0 5 1 33333 10 200 44 i
*>i16.0.0.0/4       10.1.34.4                0    100      0 4 {1,404,303,202} i
*>i21.0.0.0         10.1.34.4                0    100      0 4 1 404 303 202 i
*>i31.0.0.0         10.1.34.4                0    100      0 4 1 303 303 303 i
! Not shown—R2's switches to using AS_PATH filter-list 2 instead for peer R3
! only, and soft reconfiguration is applied.
! The next command shows the contents
! of the new filter for inbound Updates from R3. Because the "{" and "}" are not
! metacharacters, they can simply be typed directly into the regex. AS_PATH
! access-list 2 matches an AS_SET anywhere in the AS_PATH, as long as 303
! resides anywhere inside the AS_SET.
R2# show ip as-path-access-list 2
AS path access list 2
    deny {.*303.*}
    permit .*
! The next command is a test to show routes received by R2 from R3 that happen to
! have 303 anywhere in the AS_PATH. Remember, filtered routes are still
! displayed when viewing the BGP table with the received-routes option.
R2# show ip bgp neighbor 3.3.3.3 received-routes ¦ include 303
* i16.0.0.0/4       10.1.34.4                0    100      0 4 {1,404,303,202} i
* i21.0.0.0         10.1.34.4                0    100      0 4 1 404 303 202 i
* i31.0.0.0         10.1.34.4                0    100      0 4 1 303 303 303 i
! R2 has filtered the route with 303 in the AS_SET, but it did not filter the
! routes with 303 in the AS_SEQ.
R2# sh ip bgp ¦ include 10.1.34.4
* i21.0.0.0         10.1.34.4                0    100      0 4 1 404 303 202 i
* i31.0.0.0         10.1.34.4                0    100      0 4 1 303 303 303 i
```

NOTE While AS_SET and AS_CONFED_SET are both unordered lists, when applying regex logic, Cisco IOS uses the order listed in the output of the **show ip bgp** command.

BGP Path Attributes and the BGP Decision Process

BGP path attributes define different characteristics about the NLRI(s) associated with a PA. For example, the AS_PATH PA lists the ASNs through which the NLRI has been advertised. Some BGP PAs impact the *BGP decision process* by which a router chooses the best path among multiple known routes to the same NLRI. This chapter explains the BGP decision process and introduces several new PAs and other BGP features that impact that process.

Generic Terms and Characteristics of BGP PAs

Each BGP PA can be described as either a *well-known* or *optional* PA. These terms refer to whether a particular implementation of BGP software must support the PA (well known) or support for the PA is not required (optional).

Well-known PAs are either one of the following:

- **Mandatory**—The PA must be in every BGP Update
- **Discretionary**—The PA is not required in every BGP Update

These classifications relate not to the capabilities of a BGP implementation, but rather to whether a particular feature has been configured or used by default. For example, the ATOMIC_AGGREGATE PA is a well-known discretionary PA. That means that all implementations of BGP must understand this PA, but a particular router adds this PA only at its discretion, in this case by a router that creates a summary route. Conversely, the AS_PATH PA is a well-known mandatory PA, and as such must be included in every BGP Update.

BGP classifies optional PAs into one of two other categories, which relate to a router's behavior when the router's BGP implementation does not understand the PA:

- **Transitive**—The router should silently forward the PA to other routers without needing to consider the meaning of the PA.
- **Nontransitive**—The router should remove the PA so that it is not propagated to any peers.

Table 13-6 summarizes these classification terms and definitions.

Table 13-6 *Definitions of Path Attribute Classification Terms*

KEY POINT

Term	All BGP Software Implementations Must Support It	Must Be Sent in Each BGP Update	Silently Forwarded if Not Supported
Well-known mandatory	Yes	Yes	—
Well-known discretionary	Yes	No	—
Optional transitive	No	—	Yes
Optional nontransitive	No	—	No

The BGP PAs that have been mentioned so far in this book provide several good examples of the meanings behind the terms given in Table 13-6. Those PAs are summarized in Table 13-7, along with their characteristics.

Table 13-7 *BGP Path Attributes Covered So Far, and Their Characteristics*

Path Attribute	Description	Characteristics
AS_PATH	Lists ASNs through which the route has been advertised	Well-known mandatory
NEXT_HOP	Lists the next-hop IP address used to reach an NLRI	Well-known mandatory
AGGREGATOR	Lists the RID and ASN of the router that created a summary NLRI	Optional transitive
ATOMIC_AGGREGATE	Tags a summary NLRI as being a summary	Well-known discretionary
ORIGIN	Value implying from where the route was taken for injection into BGP; **i** (IGP), **e** (EGP), or **?** (incomplete information)	Well-known mandatory
ORIGINATOR_ID	Used by RRs to denote the RID of the iBGP neighbor that injected the NLRI into the AS	Optional nontransitive
CLUSTER_LIST	Used by RRs to list the RR cluster IDs in order to prevent loops	Optional nontransitive

Additions to BGP can be defined through the creation of new optional PAs, without requiring a new baseline RFC and a bump to a new version for BGP. The last two PAs in Table 13-7

list two such examples, both of which were added by RFC 1966 for the route reflectors feature.

The BGP Decision Process

The BGP decision process uses some of the PAs listed in Table 13-7, as well as several others. This section focuses on the decision process as an end to itself, with only brief explanations of new features or PAs. Following that, the text explains the details of some of the PAs that have not yet been covered in the book, as well as some other details that affect the BGP decision process.

When a BGP router learns multiple routes to the same NLRI, it must choose a single best route to reach that NLRI. BGP does not rely on a single concept like an IGP metric, but rather provides a rich set of tools that can be manipulated to affect the choice of routes. The following list defines the core of the BGP decision process to choose routes. Two additional tiebreaker steps are listed later in this section.

KEY POINT

0. **Is the NEXT_HOP reachable?**—Many texts, as well as RFC 1771, mention the fact that if a router does not have a route to the NEXT_HOP PA for a route, it should be rejected in the decision process.
1. **Highest administrative weight**—This is a Cisco-proprietary feature. The administrative weight can be assigned to each NLRI locally on a router, and the value cannot be communicated to another router. The higher the value, the better the route.
2. **Highest LOCAL_PREF PA**—This optional nontransitive PA can be set on a router inside an AS, and distributed inside the AS only. As a result, this feature can be used by all BGP routers in one AS to choose the same exit point from their AS for particular NLRI. The higher the value, the better the route.
3. **Locally injected routes**—Pick the route injected into BGP locally; if multiple routes exist, prefer ORIGIN I routes first, then ORIGIN E routes, and finally ORIGIN ? routes. (This step is seldom needed, and is sometimes omitted from other BGP references.)
4. **Shortest AS_PATH length**—The shorter the AS_PATH length, the better the route. The length calculation ignores both AS_CONFED_SET and AS_CONFED_SEQ, and treats an AS_SET as 1 ASN, regardless of the number of ASNs in the AS_SET. It counts each ASN in the AS_SEQUENCE as 1.
5. **ORIGIN PA**—IGP (I) routes are preferred over EGP (E) routes, which are in turn preferred over incomplete (?) routes.
6. **Smallest Multi-Exit Discriminator (MED) PA**—Traditionally, this PA allows an ISP with multiple peer connections to a customer AS to tell the customer AS which of the peer connections is best for reaching particular NLRI. The smaller the value, the better the route.
7. **Neighbor Type**—Routes learned via eBGP are preferred over iBGP-learned routes, with confederation eBGP equal to iBGP for this step.

8. **IGP metric for reaching the NEXT_HOP**—IGP metrics for each NLRI's NEXT_HOP are compared. The lower the value, the better the route.

Clarifications of the BGP Decision Process

The goal of this nine-step decision process is to determine the one best route to reach each NLRI. These steps do not attempt to find multiple equal routes, and install equal routes into the IP routing table, until a later step—even if the **maximum-paths** router subcommand has been configured to some number higher than the default of 1. The goal is to find the one best route to each NLRI.

First, you probably noticed that the list starts with Step 0 instead of Step 1. I debated whether to include Step 0 in the list at all. Certainly, the statement at Step 0 is true—however, one could argue that this concept is not related to choosing between multiple useful routes, because it is really a restriction as to which routes could be used, thereby being candidates to become the best route. I decided to include it in the list for a couple of reasons: it is prominently mentioned in the corresponding parts of RFC 1771, and it is part of the decision process listed in both *Internet Routing Architectures* (Halabi) and *Routing TCP/IP*, Volume II (Doyle and Carroll). It is an important point, and worth memorizing, but to help point out that some people might not even consider this logic as part of the BGP decision process, I numbered it as Step 0 to make it stand out.

KEY POINT BGP considers each decision step independently. If a step determines the best route for an NLRI, BGP does not bother with the remaining steps. For example, imagine that R1 has five routes to 9.0.0.0/10, two with AS_PATH length 3 and the others with AS_PATH length 5. The decision process did not determine a best route before reaching Step 4 (AS_PATH length). Step 4's logic can determine that two routes are better than the others because they have a shorter AS_PATH length. However, because this step does not produce the single winner, the process moves on to the next step, and *all five routes* are still considered as candidates to be the best route. Each step either produces a winner or moves on to the next step, examining all routes to that NLRI at the next step.

BGP applies this process to each unique NLRI. When overlapping NLRIs exist—for example, 130.1.0.0/16, 130.2.0.0/16, and 130.0.0.0/12—BGP attempts to find the best route for each specific prefix/prefix length.

Two Final Tiebreaker Steps in the BGP Decision Process

It is possible for BGP to fail to determine a best path to an NLRI using Steps 0 through 8, so BGP includes the following tiebreakers. These values would not typically be manipulated in a routing policy to impact the decision process.

KEY POINT

9. **Smallest advertising eBGP RID, or iBGP RID, with an exception**—If some routes are eBGP routes, the router prefers the route that was advertised by the eBGP router with the lowest RID. If only iBGP routes exist, the same logic is used but for iBGP peers. (All confederation peers are considered iBGP peers for this process.)

10. **Smallest neighbor ID**—For Step 9 to fail to produce a winning route, the router must have at least one other router with which it has multiple neighbor relationships. For this atypical case, the router now prefers the route advertised by the lowest neighbor ID, as listed in that router's **neighbor** commands.

NOTE The exception at Step 9 occurs when BGP already has a chosen best route to the NLRI but is now reapplying the decision process as a result of new information. For example, the router may have just learned of another route to reach the same NLRI. In that case, if the current best route is an eBGP route, the old best route remains the best route. This logic prevents eBGP route flaps.

NOTE For those of you more familiar with BGP, hopefully the lists describing the BGP decision process bring to mind the details you have learned in the past. For those of you less familiar with BGP, you might begin to feel a little overwhelmed by the details of the process. The lists are useful for study and memorization once you understand the background and details, which will be forthcoming in just a few pages. However, a few other general details need to be introduced before you get to the details at each step of the decision process. Hang in there!

Adding Multiple BGP Routes to the IP Routing Table

The BGP decision process has an impact on whether BGP adds multiple routes for a single NLRI to the IP routing table. The following statements summarize the logic:

KEY POINT

- If the best path for an NLRI is determined in Steps 0 through 8, BGP adds only one BGP route to the IP routing table—the best route, of course.
- If the best path for an NLRI is determined at Step 9 or 10, BGP considers placing multiple BGP routes into the IP routing table.
- Even if multiple BGP routes are added to the IP routing table, BGP still chooses only one route per NLRI as the best route; that best route is the only route to that NLRI that BGP will advertise to neighbors.

The section "The **maximum-paths** Command and BGP Decision Process Tiebreakers," later in this chapter, details the restrictions.

Mnemonics for Memorizing the Decision Process

Many people do not bother to memorize the BGP decision process steps. However, memorizing the list is very useful for both the CCIE Routing Switching written and lab exams. This section provides a set of mnemonic devices to aid you in memorizing the list. Please feel free to learn the mnemonic or skip to the next heading, at your discretion.

Table 13-8 is part of the practice effort to memorize the BGP decision tree.

Table 13-8 *BGP Decision Process Mnemonic: N WLLA OMNI*

Trigger Letter	Short Phrase	Which Is Better?
N	Next hop: reachable?	—
W	Weight	Bigger
L	LOCAL_PREF	Bigger
L	Locally injected routes	Locally injected is better than iBGP/eBGP learned
A	AS_PATH length	Smaller
O	ORIGIN	Prefer ORIGIN code I over E, and E over ?
M	MED	Smaller
N	Neighbor Type	Prefer eBGP over iBGP
I	IGP metric to NEXT_HOP	Smaller

The first mnemonic step is to memorize the nine trigger letters—single letters that, once memorized, should hopefully trigger your memory to recall some short phrase that describes the logic of each decision point. Of course, memorizing nine seemingly random letters is not easy. So, memorize them as three groups:

N
WLLA
OMNI

NOTE The nine letters are organized as shown here for several reasons. First, the single letter N, for Step 0, is purposefully separated from the other two groups because it can be argued that this step is not really part of the decision process. OMNI was separated because it is a commonly known English language prefix. And WLLA was just left over after designating OMNI.

After memorizing the trigger letter groups, you should exercise correlating the triggers to the short phrases listed in Table 13-8. (The CD contains memory-builder versions of the tables, including Table 13-8, which you can print and use for this practice if you like.) Simply write down the nine trigger letters, and exercise your memory by writing out the short phrase associated with each letter. I'd recommend practicing as the first thing you do when you pick up the book for your next

reading/study session, and do it for 5 minutes, and typically after a few rounds you will have it memorized.

Of these nine steps, I find also that most people have difficultly correlating the I to the phrase "IGP metric to reach the NEXT_HOP"; in case it helps, memorize also that the first and last of the nine items relate to NEXT_HOP. Based on that fact, and the fact that the trigger letter I implies IGP, maybe the two facts together may trigger your memory.

Once you can essentially re-create the first two columns of Table 13-8 from memory, memorize the fact that the first two quantitative decision points use bigger-is-better logic, and the rest use smaller-is-better logic. By doing so, you do not have to memorize which specific feature uses which type of logic—as long as you can write down the whole list, you can easily find the first two with quantitative comparisons (specifically Steps 1 and 2, WEIGHT and LOCAL_PREF, respectively).

Finally, Steps 9 and 10 are left to you to simply memorize. Remember that **maximum-paths** comes into play only if the first eight points do not determine a best route.

Configuring BGP Policies

BGP policies include route filters as well as tools that modify PAs and other settings that impact the BGP decision process. This section examines the Cisco IOS tools used to implement routing policies that impact the BGP decision process, covering the tools in the same order as the decision process.

Background: BGP PAs and Features Used by Routing Policies

Before getting into each individual step of the decision process, it is important to have a handy reference for the features the process manipulates, and the command output on routers that will reflect the changes made by each step. First, Table 13-9 summarizes the BGP PAs and other features used in the BGP decision process.

Table 13-9 *Proprietary Features and BGP Path Attributes that Affect the BGP Decision Process*

KEY POINT

PA/Other	Description	BGP PA Type
NEXT_HOP	Lists the next-hop IP address used to reach an NLRI.	Well-known mandatory
Weight[1]	Local Cisco-proprietary setting, not advertised to any peers. Bigger is better.	—
LOCAL_PREF	Communicated inside a single AS. Bigger is better; range 0 through $2^{32} - 1$.	Well-known discretionary

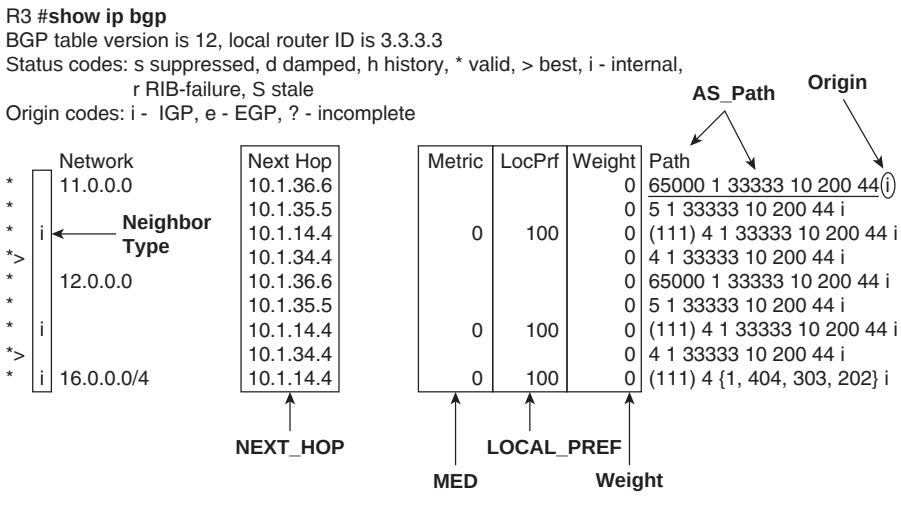
Table 13-9 *Proprietary Features and BGP Path Attributes that Affect the BGP Decision Process (Continued)*

PA/Other	Description	BGP PA Type
AS_PATH length	The number of ASNs in the AS_SEQ, plus 1 if an AS_SET exists.	Well-known mandatory
ORIGIN	Value implying the route was injected into BGP; I (IGP), E (EGP), or ? (incomplete information).	Well-known mandatory
MULTI_EXIT_ DISC (MED)	Multi-Exit Discriminator. Set and advertised by routers in one AS, impacting the BGP decision of routers in the other AS. Smaller is better.	Optional nontransitive
Neighbor Type[1]	The type of BGP neighbor from which a route was learned. Confederation eBGP is treated as iBGP for the decision process.	—
IGP metric to reach NEXT_HOP[1]	Smaller is better.	—
BGP RID[1]	Defines a unique identifier for a BGP router. Smaller is better.	—

[1] This value is not a BGP PA.

Next, Figure 13-6 shows an example of the **show ip bgp** command. Note that the locations of most of the variables used for the BGP decision process are given in the output.

Figure 13-6 *Locating Key BGP Decision Features in the* **show ip bgp** *Command*



The **show ip bgp** command lists most of the settings that impact the decision process, but it does not list the advertising router's RID, the IGP metric to reach the NEXT_HOP, or the neighbor ID that advertised the route. Two other commands do supply these three missing pieces of information. Example 13-6 shows the output of one of those commands, the **show ip bgp 16.0.0.0** command, which lists the advertising router's RID and neighbor ID. The IGP metric, of course, is given by the **show ip route** command.

Example 13-6 *Output of the* **show ip bgp 16.0.0.0** *Command on R3*

```
! Two routes to 16.0.0.0 are listed. The "from z.z.z.z" phrases identify the
! neighbor ID that advertised the route. The "(y.y.y.y)" output that follows lists
! the RID of that same router. Also, note that the
! output first identifies entry #2 as the best one, indicated by that entry (on the last
! line of output) also listing the word "best."
R3# sh ip bgp 16.0.0.0
BGP routing table entry for 16.0.0.0/4, version 8
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1          2
  (111) 4 {1,404,303,202}, (aggregated by 4 4.4.4.4), (received & used)
    10.1.14.4 (metric 3193856) from 2.2.2.2 (2.2.2.2)
      Origin IGP, metric 0, localpref 100, valid, confed-internal
  4 {1,404,303,202}, (aggregated by 4 4.4.4.4), (received & used)
    10.1.34.4 from 10.1.34.4 (4.4.4.4)
Origin IGP, metric 0, localpref 100, valid, external, best* i11.0.0.0
```

Armed with the BGP decision process steps, the definitions for the PAs that impact the process, and a good reference for where you need to look to see the values, the next several sections take a tour of the BGP decision process. Each successive heading examines the decision process steps, in sequence.

Step 0: NEXT_HOP Reachable

This decision step simply prevents BGP from making the poor choice of accepting a BGP route as best, even though that router cannot possibly forward packets to the next-hop router.

Routing policies do not typically attempt to change a NEXT_HOP address to impact a routing choice. However, the NEXT_HOP can be changed by using either the **neighbor** *neighbor-id* **next-hop-self** command (the default for eBGP peers) or the **neighbor** *neighbor-id* **next-hop-unchanged** command (the default for iBGP peers). If **next-hop-self** is used, the NEXT_HOP is set to the IP address used as the source of the BGP Update sent to that neighbor. If **next-hop-unchanged** is used, the NEXT_HOP is not changed.

Step 1: Administrative Weight

The *weight*, more fully titled *administrative weight*, allows a single router to examine inbound BGP Updates and decide which routes to prefer. The weight is not a BGP PA, but simply a Cisco-proprietary setting on a local router. In fact, it cannot be included in a BGP Update sent to another

router, because there is no place in the Update message to include the weight. Table 13-10 summarizes the key points regarding BGP weight.

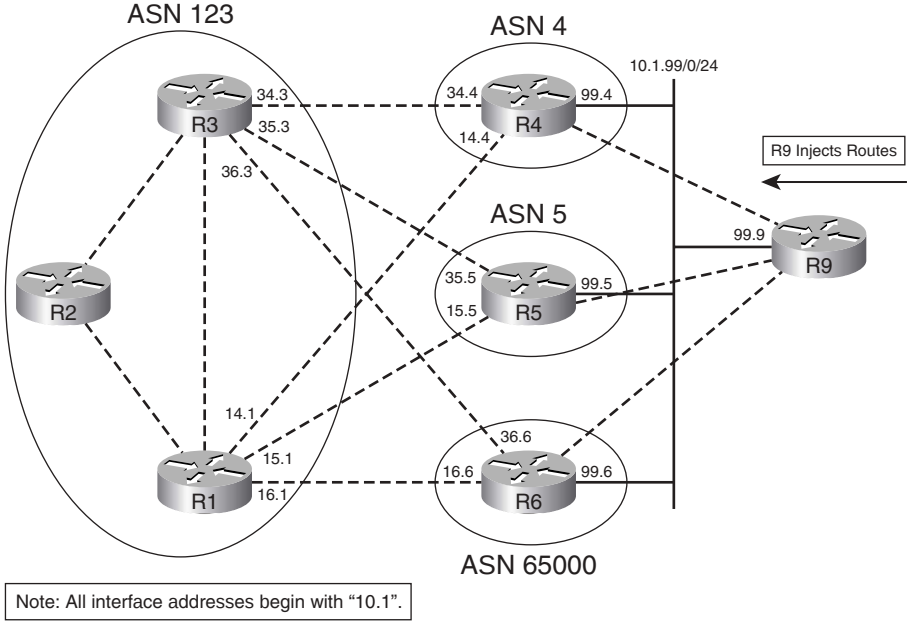
Table 13-10 *Key Features of Administrative Weight*

KEY POINT

Feature	Description
Is it a PA?	No; Cisco-proprietary feature
Purpose	Identifies a single router's best route
Scope	In a single router only
Default	0 for learned routes, 32,768 for locally injected routes
Changing the defaults	Not supported
Range	0 through 65,535 ($2^{16} - 1$)
Which is best?	Bigger values are better
Configuration	Via **neighbor route-map in** command or the **neighbor weight** command (if a route is matched by both commands, IOS uses weight specified in route map)

Figure 13-7 shows an updated version of Figure 13-4 that is used in the next example. Compared to Figure 13-4, Figure 13-7 shows the three routers in AS 123 as a full mesh of iBGP peers, with no confederations.

Figure 13-7 *Sample Network: AS 123 Without Confederations*



In Example 13-7, R1 sets the weight for NLRIs learned from R4, R5, and R6. The configuration shows both methods of configuring weight:

- Routes learned from R4 are set to weight 4 by using the **neighbor weight** command.
- Routes learned from R5 are set to weight 200 if ASN 200 is in the AS_PATH, by using a **route-map**.

Example 13-7 *Setting BGP Administrative Weight on R1*

```
! The commands below list only commands that were added to the existing R1
! configuration. All routes from R4 (10.1.14.4) will now be weight 4, and those
! matching clause 10 of the route-map, from R5 (10.1.15.5), will be weight 200.
router bgp 123
 neighbor 10.1.14.4 weight 4
 neighbor 10.1.15.5 route-map set-weight-200 in
! The AS_PATH ACL matches any AS_PATH that includes ASN 200. Note that the
! route-map requires a second permit clause with no match or set, otherwise all
! routes not matched by clause 10 will be filtered.
ip as-path access-list 5 permit _200_
!
route-map set-weight-200 permit 10
 match as-path 5
 set weight 200
!
route-map set-weight-200 permit 20
! The changes are reflected below. Note also that both networks 11 and 12 have
! weights of 200, so those routes were chosen as the best paths.
R1# sh ip bgp ¦ begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*  11.0.0.0         10.1.14.4       4294967294             4 4 1 33333 10 200 44 i
* i                 10.1.36.6       4294967294    100      0 65000 1 33333 10 200 44 i
*                   10.1.16.6       4294967294             0 65000 1 33333 10 200 44 i
*>                  10.1.15.5       4294967294           200 5 1 33333 10 200 44 i
*  12.0.0.0         10.1.14.4       4294967294             4 4 1 33333 10 200 44 i
* i                 10.1.36.6       4294967294    100      0 65000 1 33333 10 200 44 i
*                   10.1.16.6       4294967294             0 65000 1 33333 10 200 44 i
*>                  10.1.15.5       4294967294           200 5 1 33333 10 200 44 i
```

Of particular importance in the example is the fact that the route map includes clause 20, with a **permit** action and no **match** or **set** commands. The **neighbor route-map** command creates an implied filtering decision. Any route matched by a **permit** clause in the route map is implied to be allowed through, and routes matched by a **deny** clause will be filtered. Route maps use an implied **deny all** at the end of the route map for any unmatched routes. By including a final clause with just a **permit** keyword, the route map changes to use **permit all** logic, thereby passing all routes.

Step 2: Highest Local Preference (LOCAL_PREF)

The BGP LOCAL_PREF PA allows routers in an AS with multiple exit points to choose which exit point is used to reach a particular NLRI. To do so, the router that is the desired exit point sets the LOCAL_PREF for its eBGP route for that NLRI to a relatively high value, then advertises that route via iBGP. The other routers in the same AS can learn of multiple routes to reach the NLRI, but they will choose the route with the higher LOCAL_PREF as the best route.

Table 13-11 summarizes the key points regarding LOCAL_PREF.

Table 13-11 *Key Features of LOCAL_PREF*

KEY POINT

Feature	Description
PA?	Yes, well-known discretionary
Purpose	Identifies the best exit point from the AS to reach the NLRI
Scope	Throughout the AS in which it was set, including confederation sub-ASs
Default	100
Changing the default	Using the **bgp default local-preference <0-4294967295>** BGP subcommand
Range	0 through 4,294,967,295 ($2^{32} - 1$)
Which is best?	Higher values are better
Configuration	Via **neighbor route-map** command; **in** option is required for Updates from an eBGP peer

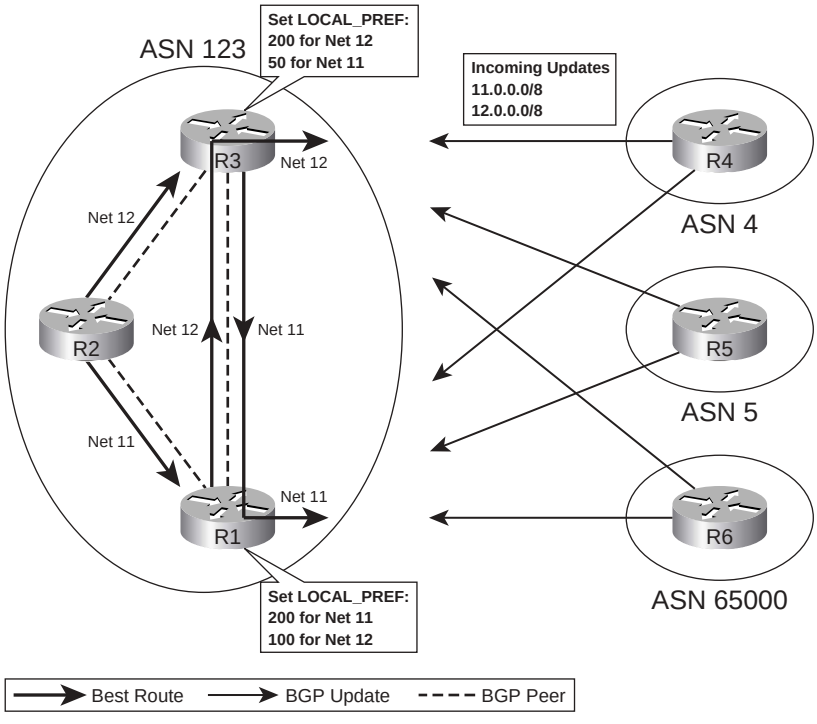
Figure 13-8 shows a typical example of using LOCAL_PREF. In this case, the engineers for AS 123 want to use R1 to forward packets to 11.0.0.0/8, but use R3 to forward packets to 12.0.0.0/8. If either route fails, the other router should be used instead.

Example 13-8 shows the configuration used on R1 and R3 to implement the following routing policy:

- AS 123 routers should use R1 to reach 11.0.0.0/8.
- AS 123 routers should use R3 to reach 12.0.0.0/8.
- R1 can use any of its three routes to reach 11.0.0.0/8, and R3 can use any of its three routes to reach 12.0.0.0/8.

To meet these design goals, R1 and R3 will set LOCAL_PREF to values higher than the default of 100.

Figure 13-8 *Typical Use of LOCAL_PREF to Influence Exit Point from AS 123.*



Example 13-8 *LOCAL_PREF Directing Packets for 11/8 Out R1 and Packets for 12/8 Out R3*

```
! R1 Config—only the relevant configuration is shown. The same route-map is
! called for incoming Updates from R4, R5, and R6. Note that the route-map
! includes a permit clause 20 with no match or set commands to permit
! any routes not specified in clause 10 to pass without changes. The route-map
! allows the LOCAL_PREF for 12.0.0.0/8 to default (100).
router bgp 123
 neighbor 10.1.14.4 route-map 11-high-12-default in
 neighbor 10.1.15.5 route-map 11-high-12-default in
 neighbor 10.1.16.6 route-map 11-high-12-default in
!
access-list 11 permit 11.0.0.0
!
route-map 11-high-12-default permit 10
 match ip address 11
 set local-preference 200
!
route-map 11-high-12-default permit 20
```

```
! R3 Config—Same general concept as R1, but the 12.0.0.0/8 route is assigned
```

Example 13-8 *LOCAL_PREF Directing Packets for 11/8 Out R1 and Packets for 12/8 Out R3 (Continued)*

```
! LOCAL_PREF 200, and 11.0.0.0/8 is assigned LOCAL_PREF 50.
router bgp 123
 neighbor 10.1.34.4 route-map 11-low-12-high in
 neighbor 10.1.35.5 route-map 11-low-12-high in
 neighbor 10.1.36.6 route-map 11-low-12-high in
!
access-list 11 permit 11.0.0.0
access-list 12 permit 12.0.0.0
!
route-map 11-low-12-high permit 10
 match ip address 12
 set local-preference 200
!
route-map 11-low-12-high permit 20
 match ip address 11
 set local-preference 50
!
route-map 11-low-12-high permit 30
! R3 now shows the LOCAL_PREF values. R3's best route to 12.0.0.0 is the one it
! learned from R4 (10.1.34.4). Its best route to 11.0.0.0 is the only one of the
! 4 routes with LOCAL_PREF 200—the one learned from R1. Note also that the
! administrative weights are all tied at 0; otherwise, BGP might have chosen a
! different best route.
R3# show ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*  11.0.0.0         10.1.36.6       4294967294     50      0 65000 1 33333 10 200 44 i
*>i                 10.1.14.4       4294967294    200      0 4 1 33333 10 200 44 i
*                   10.1.35.5       4294967294     50      0 5 1 33333 10 200 44 i
*                   10.1.34.4       4294967294     50      0 4 1 33333 10 200 44 i
*  12.0.0.0         10.1.36.6       4294967294    200      0 65000 1 33333 10 200 44 i
*                   10.1.35.5       4294967294    200      0 5 1 33333 10 200 44 i
*>                  10.1.34.4       4294967294    200      0 4 1 33333 10 200 44 i
R3# show ip bgp 11.0.0.0
! lines omitted for brevity
  4 1 33333 10 200 44, (received & used)
    10.1.14.4 (metric 2681856) from 1.1.1.1 (1.1.1.1)
      Origin IGP, metric 4294967294, localpref 200, valid, internal, best
! lines omitted for brevity
```

```
! Because R3's best route to 11.0.0.0/8 is through R1, R3 does not advertise that
! iBGP route to R2. Similarly, R1's best route to 12.0.0.0/8 is through R3, so R1
! does not advertise its best route to 12.0.0.0/8, again because it is an iBGP
! route. As a result, R2 receives only one route to each of the two networks.
R2# show ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*>i11.0.0.0         10.1.14.4       4294967294    200      0 4 1 33333 10 200 44 i
*>i12.0.0.0         10.1.34.4       4294967294    200      0 4 1 33333 10 200 44 i
```

This example does meet the stated design goals, but note that one design goal states that it does not matter which of the three eBGP routes R1 and R3 use to reach their assigned prefixes. Interestingly, R1 did not choose its best BGP route to network 11.0.0.0/8 based on LOCAL_PREF, nor did R3 choose its best route to 12.0.0.0/8 based on LOCAL_PREF. Note that R1 and R3 had three routes that tied based on LOCAL_PREF. In this case, their decisions happened to fall all the way to Step 9—the lowest advertising BGP RID. As a result, R3 chose the route through R4 (RID 4.4.4.4) instead of R5 (RID 5.5.5.5) or R6 (RID 6.6.6.6).

Had R1 or R3 wanted to impact which of the three eBGP routers to use to reach their respective NLRI, the route map could have been changed to match routes from each neighbor, and set the LOCAL_PREF to different high values. For example, the LOCAL_PREF could be set to 204, 205, and 206 for R4, R5, and R6, respectively, thereby making R3 choose to use the route through R6 if 12.0.0.0/8 was learned from each of the three eBGP peers. To match, the **match ip next-hop** or **match ip route-source** command could be used, or a different route map could simply be used per neighbor.

Step 3: Choose Between Locally Injected Routes Based on ORIGIN PA

This logic step is seldom used, but it is a valid part of the BGP decision process. To appreciate why it is so seldom needed, consider the following: BGP assigns a weight of 32,768 to routes locally injected into BGP. As a result, a router would have already picked a locally injected route as best because of to its high weight.

Two general cases can occur that cause a router to use the logic for this step. The first case is unlikely. A router must locally inject an NLRI, learn the same NLRI from a neighbor, and use an inbound **route-map** to set the weight of that received NLRI to the same value as the locally injected route. That only occurs in lab experiments.

The second case occurs when a router attempts to inject routes locally via multiple methods, and the same NLRI is injected from two different sources. For example, imagine that R1 injects a route to network 123.0.0.0/8 due to both a **network 123.0.0.0** command and a **redistribute connected** command. Both routes would have default weights of 32,768, and both would default to the same LOCAL_PREF. The two routes would then be compared at this step, with the ORIGIN code determining which route is best.

The logic for the second (and only likely) case to use this step in the decision process can be reduced to the following:

> When the same NLRI is locally injected into BGP from multiple methods, pick the route with the better ORIGIN PA.

The only hard part is memorizing the ORIGIN codes, and that “I” is better than “E” is better than “?”.

Step 4: Shortest AS_PATH

Routers can easily determine the shortest AS_PATH length by using a few rules that define how to account for all four parts of the AS_PATH—the AS_SEQ, AS_SET, AS_CONFED_SEQ, and AS_CONFED_SET. Additionally, routing policies can change the number of ASNs in the AS_PATH. Table 13-12 summarizes the key points regarding AS_PATH length.

Table 13-12 *Features that Impact the Total Number of ASs in the AS_PATH Length Calculation*

KEY POINT

Feature	Description
AS_SET	Regardless of actual length, it counts as a single ASN.
Confederations	AS_CONFED_SEQ and AS_CONFED_SET do not count at all in the calculation.
aggregate-address command	If the component subnets have different AS_PATHs, the summary route has only the local AS in the AS_SEQ; otherwise, the AS_SEQ contains the AS_SEQ from the component subnets. Also, the presence/absence of the **as-set** command option determines whether the AS_SET is included.
neighbor remove-private-as command	Used by a router attached to a private AS (64512–65535), causing the router to remove the private ASN used by the neighboring AS.
neighbor local-as no-prepend command	Allows a router to use a different AS than the one on the **router bgp** command; with the **no-prepend** option, the router does not prepend any ASN when sending eBGP Updates to this neighbor.
AS_PATH prepending	Using a **neighbor route-map** in either direction, the route-map can use the **set as-path prepend** command to prepend one or more ASNs into the AS_SEQ.
bgp bestpath as-path ignore command	Removes the AS_PATH length step from the decision tree for the local router.

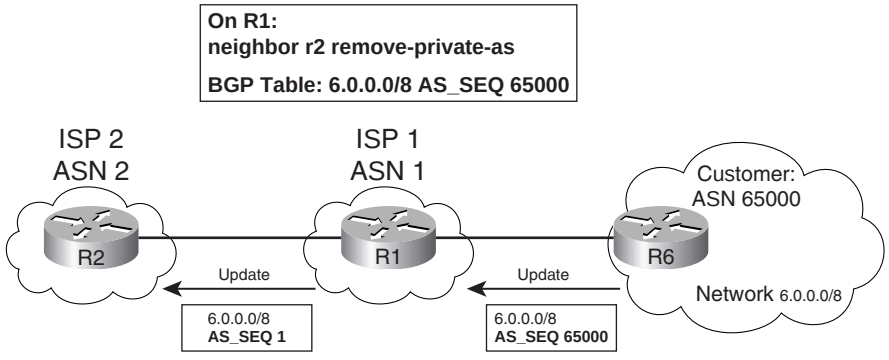
The typical logic at this step simply requires the router to calculate the number of ASNs in the AS_SEQ, and add 1 if an AS_SET exists. However, the table mentions several other features that impact what ASNs are used, and whether an eBGP peer adds an ASN. These additional features are covered next before moving on to Step 5 of the BGP decision process.

Removing Private ASNs

Private ASNs (64,512–65,535) should not be used in AS_PATHs advertised into the Internet beyond a single ISP. One purpose of this private range is to conserve the ASN space by assigning private ASNs to customers that only connect to that single ISP. Then, the ISP can simply remove the private ASN before advertising any routes for that customer outside its network.

Figure 13-9 shows the typical case for using a private AS. While the concept is relatively simple, the configuration details can be a bit surprising.

Figure 13-9 *Typical Use of Private ASNs and the* **neighbor remove-private-as** *Command*



Following Figure 13-9, right to left, here are the key points:

- R6, inside the private AS, does not require any special commands.
- R1, acting as a router in the sole ISP to which ASN 65000 is connected, lists private AS 65000 in its BGP table for any routes learned from R6.
- R1 needs the command **neighbor R2 remove-private-as** under router BGP, telling R1 to remove any private ASNs from AS_PATHs advertised to router R2.

Cisco IOS has several restrictions regarding whether a private AS is removed as a protection against causing routing loops:

- Private ASNs can be removed only at the point of sending an eBGP Update.
- If the current AS_SEQ contains both private and public ASNs, the private ASNs will not be removed.
- If the ASN of the eBGP peer is in the current AS_PATH, the private ASNs will not be removed, either.

This feature works with confederations as well, with the same restrictions being applied to the AS_CONFED_SEQ.

AS_PATH Prepending and Route Aggregation

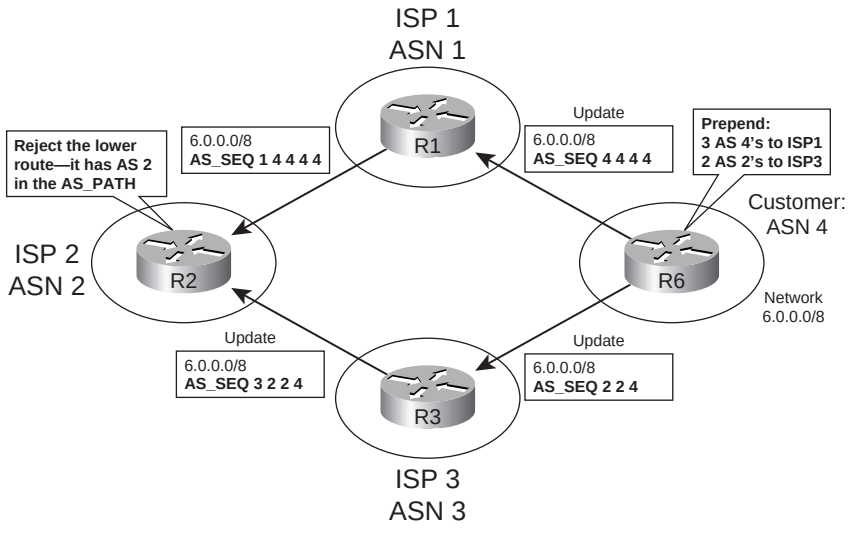
The concept and motivation behind the AS_PATH prepend feature is simple—impact the AS_PATH length decision step by increasing the length of the AS_PATH. To do so, a router simply configures a route map, refers to it with a **neighbor route-map** command, with the route map using the **set as-path prepend** *asn1 asn2…* command. As a result, the route map prepends additional ASNs to the AS_SEQUENCE.

Any ASN can be prepended, but in practice, it makes the most sense to prepend the local router's ASN. The reason is that prepending some other ASN prevents that route from being advertised into

that AS—a scenario that might not be intended. Also, if the AS_PATH needs to be lengthened by more than one ASN, the **set** command can repeat the same ASN multiple times, as shown in Example 13-9.

Figure 13-10 shows a design depicting the use of AS_PATH prepending. R6 correctly prepends its own ASN 4 for routes advertised to R1 (in the top part of the figure). R6 also causes problems by prepending ASN 2 for the route sent to R3 (in the lower part of the figure).

Figure 13-10 *Options for Prepending ASNs*



While AS_PATH prepending lengthens the AS_PATH, route aggregation may actually decrease the AS_PATH length. Route aggregation (summarization) with the BGP **aggregate-address** command impacts the AS_PATH length in a couple of ways:

- The router checks the component subnets' AS_PATH AS_SEQ values. If all the component subnets' AS_SEQ values are identical, the aggregate route uses that same AS_SEQ.
- If the component subnets' AS_SEQ values differ at all, the aggregating router uses a null AS_SEQ for the aggregate. (When advertised to an eBGP peer, the router does prepend its local ASN, as normal.) Of course, this process shortens the AS_PATH length.

Additionally, the **aggregate-address** command with the **as-set** option may lengthen the AS_PATH length calculation as well. When a router uses this command with the **as-set** option, and the aggregate empties out the AS_SEQ as described in the previous paragraph, the router adds an AS_SET segment to the AS_PATH. (Conversely, if the aggregate does not empty the AS_SEQ, the router does not create the AS_SET, as it is not needed for loop prevention in that case.) The AS_SET includes all ASNs of all component subnets.

The BGP AS_PATH length calculation counts the entire AS_SET as 1, regardless of the actual length.

Example 13-9 shows examples of both AS_PATH prepending and route aggregation on the AS_PATH length. In the example, the familiar network of Figure 13-7 is used. The following features are used in the example:

- R4 prepends route 11.0.0.0/8 with three additional ASN 4s, using an outbound route-map, before advertising routes into AS 123.
- R4 and R5 both summarize 16.0.0.0/4, but R4 uses the **as-set** option and R5 does not.

As a result of the second item, R3 learns both summaries, but treats the summary from R5 as better, because the AS_SET in R4's route counts as 1 ASN in the AS_PATH length calculation.

Example 13-9 *AS_PATH Prepending and an Examination of Route Summarization and AS_PATH Length*

```
! R4's configuration shows the route-map called add3-4s for its neighbor commands
! for R1 (10.1.14.1) and R3 (10.1.34.3). The route-map matches 11.0.0.0/8,
! prepending three additional ASN 4s. The normal process of prepending the local AS
! before advertising over eBGP peer connection adds the 4th instance of ASN 4. As
! usual, the route-map needs a null final clause with a permit so that the rest
! of the routes are not affected.
router bgp 4
 aggregate-address 16.0.0.0 240.0.0.0 as-set
 neighbor 10.1.14.1 route-map add3-4s out
 neighbor 10.1.34.3 route-map add3-4s out
!
ip prefix-list match11 seq 5 permit 11.0.0.0/8
!
route-map add3-4s permit 10
 match ip address prefix-list match11
 set as-path prepend 4 4 4
!
route-map add3-4s permit 20
```

```
! Below, first focus on 11.0.0.0/8. The highlighted route with NEXT_HOP 10.1.34.4
! (R4) has four consecutive 4s in the AS_PATH, showing the effects of the prepending
! on R4. The route through 10.1.35.5 ends up being best based on the tiebreaker
! at Step 9.
! Next, look at 16.0.0.0/4. The route through 10.1.34.4 is considered
! to be AS_PATH length 2, but the length through 10.1.35.5 is only 1. The
! route to 16.0.0.0/4 through NEXT_HOP 10.1.35.5 is chosen over the route through
! 10.1.15.5 because it is eBGP, versus iBGP for the route through 10.1.15.5.
R3# show ip bgp ¦ begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*  11.0.0.0         10.1.36.6       4294967294             0 65000 1 33333 10 200 44 i
* i                 10.1.16.6       4294967294             0 65000 1 33333 10 200 44 i
*                   10.1.34.4       4294967294             0 4 4 4 4 1 33333 10 200 44 i
*>                  10.1.35.5       4294967294             0 5 1 33333 10 200 44 i
*  16.0.0.0/4       10.1.34.4       4294967294             0 4 {1,404,303,202} ?
*>                  10.1.35.5       4294967294             0 5 i
* i                 10.1.15.5       4294967294             0 5 i
```

Step 5: Best ORIGIN PA

The well-known mandatory BGP ORIGIN PA characterizes a route based on how it was injected into BGP. The ORIGIN is either IGP (i), EGP (e), or incomplete (?).

The actual BGP decision process for the ORIGIN code is quite simple. First, an ORIGIN of EGP (e) should not occur today, because EGP is not even supported in current IOS revisions. So, the logic reduces to the following:

KEY POINT

If the set of routes to reach a single NLRI includes only one route of ORIGIN code IGP (i), and all the others as incomplete (?), then the route with ORIGIN i is the best route.

BGP routing policies may set the ORIGIN code explicitly by using the **set origin** route map subcommand, although the earlier steps in the BGP decision process are typically better choices for configuring BGP policies. BGP determines the ORIGIN code based on the method used to inject the routes, along with the options used with the **aggregate-address** command.

Chapter 12's section titled "The ORIGIN Path Attribute" describes more detail about the ORIGIN PA and how NLRI are assigned an ORIGIN code.

Step 6: Smallest Multi-Exit Discriminator

The purpose of the MED (or MULTI_EXIT_DISC) is to allow routers in one AS to tell routers in a neighboring AS how good a particular route is. In fact, because of how MED works, it is often called the *BGP metric*, even though it is not close to the top of the BGP decision process. Figure 13-11 shows a classic case for the use of MED, where a customer has two links connecting it to a single ISP. The ISP, aware of its best routes for 11.0.0.0/8 and 12.0.0.0/8, can set MED so that the customer routes packets to the eBGP peer that is closest to the destination network.

The ISP, knowing its best route to reach 11.0.0.0/8 is through R5, configures R5 to set a low MED for that prefix, and R7 to set a higher MED for the same prefix. As a result, the BGP routers in customer AS 123 choose the route through the top peer connection. The customer could then use the route through the lower link to R7 if the top connection failed.

Figure 13-11 shows the classic topology—a customer using a single ISP, but with multiple links to the ISP. Many customers want redundant ISPs as well, or at least connections to multiple autonomous systems controlled by a single, large ISP. MED can also be used in such cases, but it requires the multiple ISPs or multiple ASs in the same ISP to use the same policy when determining the MED values to set. For example, two ISPs might agree that because one ISP has more link bandwidth to a certain range of BGP prefixes, that ISP will set a lower MED for those NLRIs.

Figure 13-11 *Typical Use of MED*

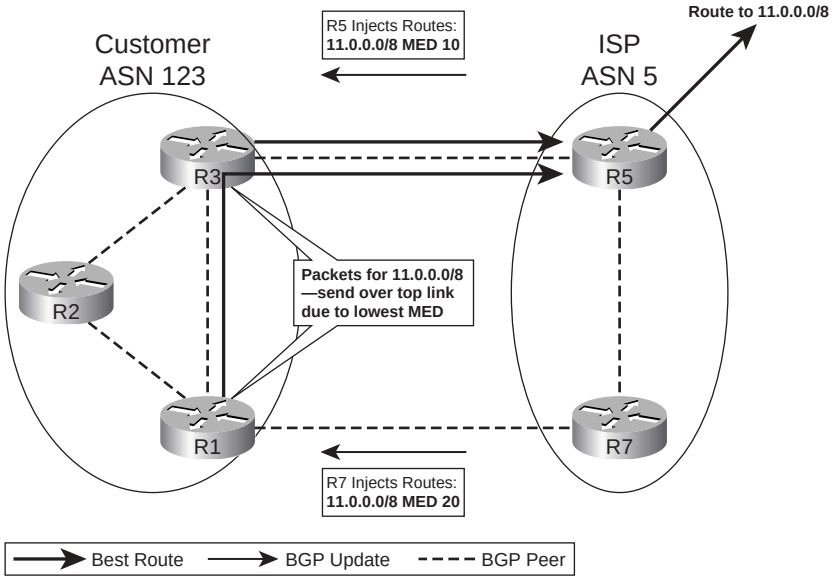


Table 13-13 summarizes the key points regarding MED.

Table 13-13 *Key Features of MED*

KEY POINT

Feature	Description
Is it a PA?	Yes, optional nontransitive
Purpose	Allows an AS to tell a neighboring AS the best way to forward packets into the first AS
Scope	Advertised by one AS into another, propagated inside the AS, but not sent to any other ASs
Default	0
Changing the default	Using the **bgp bestpath med missing-as-worst** BGP subcommand; sets it to the maximum value
Range	0 through 4,294,967,295 ($2^{32} - 1$)
Which is best?	Smaller is better
Configuration	Via **neighbor route-map out** command, using the **set metric** command inside the route-map

Configuring MED: Single Adjacent AS

Example 13-10 shows an example MED configuration that matches Figure 13-11, with R5 and R7 setting the MED for 11.0.0.0/8 to 10 and 20, respectively.

Example 13-10 *Classical MED Example Between Two ASs*

```
! The pertinent R5 configuration follows. R5 simply matches 11.0.0.0/8 and sets
! the metric to 10. The route-map includes a default permit any clause at the end
! to avoid affecting other routes.
router bgp 5
 neighbor 10.1.35.3 route-map set-med out
!
ip prefix-list 11 seq 5 permit 11.0.0.0/8
!
route-map set-med permit 10
 match ip address prefix-list 11
 set metric 10
route-map set-med permit 20
! R7's configuration is not shown, but it is basically the same regarding the
! setting of the MED. However, R7 sets the MED to 20.
```

```
! R1 lists routes with R5 (10.1.35.5) and R7 (10.1.17.7) as NEXT_HOP; the route
! through R5 is best due to the lower MED.
R1# show ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*>i11.0.0.0         10.1.35.5               10    100      0 5 1 33333 10 200 44 i
*                   10.1.17.7               20             0 5 1 33333 10 200 44 i
*> 12.0.0.0         10.1.35.5                              0 5 1 33333 10 200 44 i
* i                 10.1.17.7                0    100      0 5 1 33333 10 200 44 i
```

```
! R3 sees only the MED 10 route. R1's best route to NEXT_HOP 10.1.35.5 is through
! R3, so R1 did not advertise its best route to 11.0.0.0/8 to iBGP peer R3.
R3# show ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 11.0.0.0         10.1.35.5               10             0 5 1 33333 10 200 44 i
*> 12.0.0.0         10.1.35.5                              0 5 1 33333 10 200 44 i
* i                 10.1.17.7                0    100      0 5 1 33333 10 200 44 i
```

KEY POINT It is important that both R5 and R7 set the MED for 11.0.0.0/8. If R5 had set MED to 10, and R7 had done nothing, the router through R7 would have been the best route. R1 and R3 would have used their assumed default setting of 0 for MED for the route through R1 and R7, and, as with IGP metrics, smaller is better with MED. A better default for MED can be set by using the **bgp bestpath med missing-as-worst** BGP subcommand, which resets a router's default MED to the largest possible MED value, instead of the lowest. Note that it is important that all routers in the same AS either use the default of 0 or configure this command; otherwise, routing choices will be affected.

Configuring MED: Multiple Adjacent Autonomous Systems

By default, a Cisco router ignores MED when the multiple routes to a single NLRI list different neighboring ASNs. This default action makes sense—normally you would not expect two different neighboring ISPs to have chosen to work together to set MEDs. To override this default and consider the MED in all cases, a router needs to configure the **bgp always-compare-med** BGP subcommand. If used on one router, all routers inside the same AS should also use the **bgp always-compare-med** command, or routing loops may result.

Additionally, some Cisco documents imply that the internal BGP decision process for the MED may be different depending on the order of the entries in the BGP table. Interestingly, BGP lists the table entries from newest (most recently learned) to oldest in the output of the **show ip bgp** and **show ip bgp** *prefix* commands. Depending on that order, in some cases in which the competing routes for the same NLRI have different MEDs from different autonomous systems, the order of the entries impacts the final choice of the best route. In part, the difference results from the fact that Cisco IOS (by default) processes the list sequentially—which means it processes the first pair of routes (newest), picks the best of those two, then compares that one with the next newest, and so on.

Cisco solved this nondeterministic behavior for MED processing problem by creating an alternative process for analyzing and making the MED decision. With this new process, BGP processes the routes per adjacent AS, picking the best from each neighboring AS, and then comparing those routes. This logic provides a deterministic choice based on MED—in other words, it removes the possibility of BGP picking a different route based on the order of the routes in the BGP table. To enable this enhanced logic, add the **bgp deterministic-med** command to the routers in the same AS. In fact, Cisco recommends this setting for all new BGP implementations.

The Scope of MED

The MED PA is not intended to be advertised outside the AS that heard the MED in an incoming BGP Update. Typically, and as shown in the examples in this section, the MED can be set in an outbound route map by a router in one AS to influence the BGP decision process in another AS. So, the MED value is set by routers in one AS, and learned by routers in another AS. However, after reaching the other AS, the MED is advertised inside the AS, but not outside the AS. For example, in Figure 13-11, R5 and R7 set the MED, and advertise it into AS 123. However, if routers in AS 123 had any other eBGP connections to other ASNs, they would advertise the NLRI, but they would not include the MED value.

MED can also be set via inbound route maps, although that is not the intended design with which to use MED. When setting MED via an inbound route map, the MED is indeed set. The router can advertise the MED to iBGP peers. However, the MED is still not advertised outside the local AS.

Step 7: Prefer Neighbor Type eBGP over iBGP

This step is rather simple, and needs very little elucidation. Keeping in mind that the goal is a single best route for each NLRI, this decision point simply looks to see if a single eBGP route exists. If so, that route is chosen. If multiple eBGP routes exists, this decision point cannot determine the best route.

Interestingly, BGP uses this decision point frequently when two or more enterprise routers connect to the same ISP. Each border BGP router in the enterprise receives the same prefixes with the same AS_PATH lengths from the ISP, and then these border BGP routers advertise these routes to their iBGP peers. So, each enterprise border router knows of one eBGP route to reach each prefix, and one or more iBGP routes to the same prefix learned from that enterprise's other border routers. With no routing policies configured, the routes tie on all decision points up to this one, including AS_PATH length, because all the prefixes were learned from the same neighboring ISP. The decision process reaches this step, at which point the one eBGP route is picked as the best route.

Step 8: Smallest IGP Metric to the NEXT_HOP

This step again requires little explanation. The router looks for the route that would be used to reach the NEXT_HOP listed in each BGP table entry for a particular prefix. It is mentioned here just to complete the list.

The maximum-paths Command and BGP Decision Process Tiebreakers

The goal of the BGP decision tree is to find the one best BGP route to each NLRI, from that router's perspective. That router then considers only its best routes for advertising to other routers, restricting those routes based on AS_PATH loop prevention and routing policy configuration. That router also attempts to add that best route, and that best route only, to its IP routing table. In fact, as long as another routing source has not found a route to the same prefix, with a better administrative distance, the best BGP route is placed into that router's routing table.

If BGP has not chosen a best route for a particular NLRI after Steps 0 through 8, then multiple routes tie for being the best route. At this point, BGP needs to make two important decisions:

- **Which route is best**—BGP uses two tiebreakers, discussed next, to determine which route is best.
- **Whether to add multiple BGP routes for that NLRI to the IP routing table**—BGP considers the setting of the **maximum-paths** command to make this decision, as described after the discussion of Steps 9 and 10.

Even if BGP adds to the IP routing table multiple BGP routes to the same prefix, it still picks only one as the best route in the BGP table.

Step 9: Lowest BGP Router ID of Advertising Router (with One Exception)

The first tiebreaker is to pick the route with the lowest RID. The logic is actually two steps, as follows:

1. Examine the eBGP routes only, picking the route advertised by the router with the lowest RID.
2. If only iBGP routes exist, pick the route advertised by the router with the lowest RID.

These straight-forward rules are followed in some cases, but not in some others. The exception to this rule occurs when BGP already has a best route to the NLRI, but it has learned new BGP information from other routers, including a new BGP route to reach a previously known prefix. The router then applies its BGP decision process again to decide whether to change its opinion of which route is best for that NLRI. If the decision process does not determine a best route by this step, this step uses the following default logic:

> If the existing best route is an eBGP route, do not replace the existing best route, even if the new route has a smaller RID.

The reasoning is that replacing the route could result in route flaps, so keeping the same route is fine. This behavior can be changed so that the lowest RID is always used, by configuring the **bgp bestpath compare-routerid** BGP subcommand. Note that this exception only applies to eBGP routes; if the currently best route is an iBGP route, the decision is simply based on the lowest advertising router's RID.

Step 10: Lowest Neighbor ID

If Step 9 did not break the tie, then the router has at least two **neighbor** commands that point to the same router, and that router happens to have the lowest RID of all current neighbors advertising the NLRI in question. Typically, if redundancy exists between two routers, the configuration uses loopback interfaces, a single **neighbor** command, and the **neighbor ebgp-multihop** command if the neighbor is an eBGP neighbor. However, using a pair (or more) of **neighbor** commands pointing to a single neighboring router is a valid configuration option; this final tiebreaker provides a way to break ties for this case.

At this point, the router looks at the IP addresses on the **neighbor** commands corresponding to all the neighbors from which the route was received, and it picks the lowest neighbor IP address. Note that, as usual, it considers all routes again at this step, so it may not pick the neighboring router with the lowest RID at this point.

The BGP maximum-paths Command

BGP defaults the **maximum-paths** command to a setting of 1—in other words, only the BGP best route in the BGP table could possibly be added to the IP routing table. However, BGP will consider adding multiple entries to the IP routing table, for the same NLRI, under certain conditions—conditions that differ based on whether the best route is an eBGP route or an iBGP route.

First, consider eBGP routes. The following rules determine if and when a router will add multiple eBGP routes to the IP routing table for a single NLRI:

KEY POINT

1. BGP must have had to use a tiebreaker (Step 9 or 10) to determine the best route.
2. The **maximum-paths** *number* command must be configured to something larger than the default of 1.
3. Only eBGP routes whose adjacent ASNs are the same ASN as the best route are considered as candidates.
4. If more candidates exist than that called for with the **maximum-paths** command, the tiebreakers of Steps 9 and 10 determine the ones to use.

Although the list is detailed, the general idea is that the router can trust multiple routes, but only if the packets end up in the same adjacent AS. Also, BGP must restrict itself to not use multipath if the best route was found via Steps 0 through 8 of the decision process, because forwarding based on another route could cause loops.

Next, consider iBGP routes. The rules for iBGP have some similarities with eBGP, and a few differences, as follows:

KEY POINT

1. Same rule as eBGP rule 1.
2. The **maximum-paths ibgp** *number* command defines the number of possible IP routes, instead of the **maximum-paths** *number* command used for eBGP.
3. Only iBGP routes with differing NEXT_HOP settings are considered as candidates.
4. Same rule as eBGP rule 4.

The rationale is similar to eBGP with regard to most of the logic. Additionally, it does not help to add multiple IP routes if the NEXT_HOP settings are equal, so BGP performs that additional check.

Finally, the **maximum-paths eibgp** *number* command seemingly would apply to both iBGP and eBGP routes. However, this command only applies when MPLS is in use. Table 13-14 summarizes the key commands related to BGP multipath.

Table 13-14 *BGP* **maximum-paths** *Command Options*

KEY POINT

Command	Conditions for Use
maximum-paths *number*	eBGP routes only
maximum-paths ibgp *number*	iBGP routes only
maximum-paths eibgp *number*	Both types, but MPLS only

BGP Communities

The BGP COMMUNITY PA provides a mechanism by which to group routes so that routing policies can be applied to all the routes with the same community. By marking a set of routes with the same COMMUNITY string, routers can look for the COMMUNITY string and then make policy decisions—like setting some PA that impacts the BGP decision process, or simply filtering the routes. BGP communities are powerful in that they allow routers in one AS to communicate policy information to routers that are one or more autonomous systems distant. In fact, because the COMMUNITY PA is an optional transitive PA, it can pass through autonomous systems that do not even understand the COMMUNITY PA, and then still be useful at another downstream AS.

Figure 13-12 shows an example of one way in which communities can be used. The goal with this design is to have the engineers in ASNs 4 and 5 work together to decide which of them has the best route to reach each prefix, and then somehow tell the routers in ASN 123. That may sound familiar—that is exactly the motivation behind using MED, as shown in Figure 13-11. However, MED is relatively far into the BGP decision process, even after shortest AS_PATH. A better design might be to set the COMMUNITY PA, and then let the routers in ASN 123 react to the COMMUNITY string and set LOCAL_PREF based on that value, because LOCAL_PREF is considered early in the BGP decision process.

Figure 13-12 *Using COMMUNITY to Augment Routing Policies*

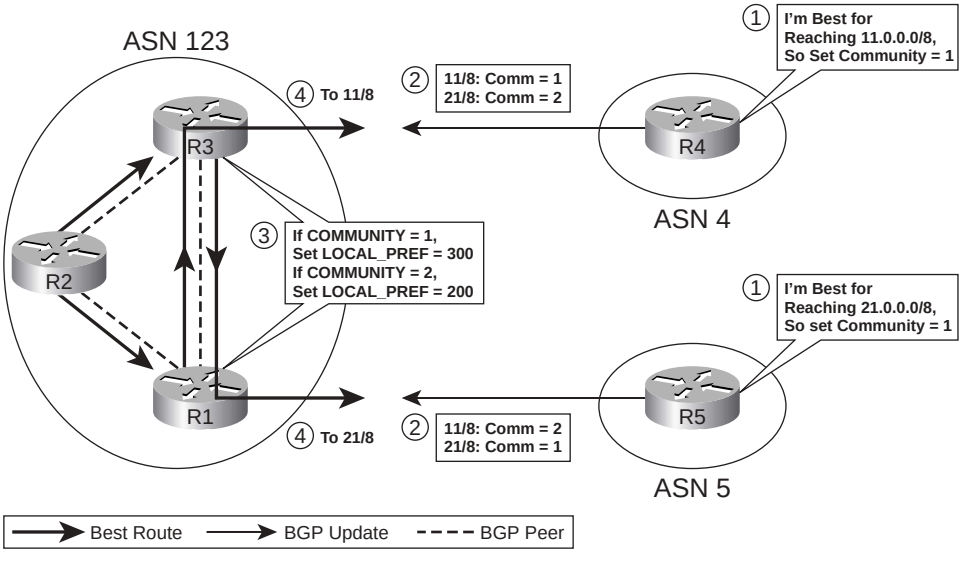


Figure 13-12 depicts the following steps:

1. The engineers at AS 4 and AS 5 agree as to which prefixes are best reached by each AS.

2. They then configure outbound route maps on their respective neighbor connections to AS 123, setting COMMUNITY to 1 for routes for which they are the best path, and setting COMMUNITY to 2 for some other routes.
3. R1 and R3 receive the Updates, match the NLRI based on the COMMUNITY, and set LOCAL_PREF to a large value for routes whose COMMUNITY was set to 1.
4. The LOCAL_PREF settings impact the BGP choice for the best routes.

This design includes several advantages over some of the options covered earlier in the chapter. It includes the best aspects of using LOCAL_PREF, helping AS 123 decide which neighboring AS to use to reach each prefix. However, it puts the choice of which routes should be reached through AS 4 and AS 5 into the hands of the folks running AS 4 and AS 5. If the AS 4 or AS 5 topology changes, link speeds increase, or other changes occur, the route maps that set the COMMUNITY in AS 4 and AS 5 can be changed accordingly. No changes would be required inside AS 123, because it already simply looks at the COMMUNITY string. Assuming that AS 123 is an enterprise, and AS 4 and AS 5 are ISPs, the ISPs can make one set of changes and impact the routing choices of countless customers.

Example 13-11 shows the configuration matching the scenario of Figure 13-12. The configuration follows mostly familiar commands and reasoning, with two additional features:

- R4 and R5 (AS 4 and AS 5) must use the **neighbor send-community** BGP subcommand, which tells BGP to include the COMMUNITY PA in the Update. Without that command, the Update does not even include the COMMUNITY PA.
- R1 and R3 (AS 123) need to match NLRI based on the received COMMUNITY values, so they must configure *community lists* that match the COMMUNITY, by using the **ip community-list** command.

Example 13-11 *Setting COMMUNITY, and Reacting to COMMUNITY to Set LOCAL_PREF*

```
! R4 must add the neighbor send-community command, otherwise it will not include
! the COMMUNITY PA in Updates sent to R3. The route-map matches 11/8, and sets
! COMMUNITY to 1, and matches 21/8 and sets COMMUNITY to 2.
router bgp 4
 neighbor 10.1.34.3 send-community both
 neighbor 10.1.34.3 route-map comm out
!
ip prefix-list 11 seq 5 permit 11.0.0.0/8
ip prefix-list 21 seq 5 permit 21.0.0.0/8
!
route-map comm permit 10
 match ip address prefix-list 11
 set community 1
!
```

continues

Example 13-11 *Setting COMMUNITY, and Reacting to COMMUNITY to Set LOCAL_PREF (Continued)*

```
route-map comm permit 20
 match ip address prefix-list 21
 set community 2
!
route-map comm permit 30
```

```
! R5 has essentially the same configuration, except that R5 sets COMMUNITY to 1
! for 21/8 and to 2 for 11/8—the opposite of R4.
router bgp 5
 neighbor 10.1.15.1 send-community
 neighbor 10.1.15.1 route-map comm out
!
ip prefix-list 11 seq 5 permit 11.0.0.0/8
ip prefix-list 21 seq 5 permit 21.0.0.0/8
!
route-map comm permit 10
 match ip address prefix-list 11
 set community 2
!
route-map comm permit 20
 match ip address prefix-list 21
 set community 1
!
route-map comm permit 30
```

```
! R3 Config: Next, R3 matches on the received COMMUNITY strings and sets
! LOCAL_PREF using a route-map called react-to-comm. The only way to match the
! COMMUNITY is to refer to an ip community-list, which then has the matching
! parameters.
router bgp 123
 neighbor 10.1.34.4 route-map react-to-comm in
!
ip community-list 1 permit 1
ip community-list 2 permit 2
!
route-map react-to-comm permit 10
 match community 1
 set local-preference 300
!
route-map react-to-comm permit 20
 match community 2
 set local-preference 200
!
route-map react-to-comm permit 30
! Not shown—R1 Config. R1's config matches R3's in every way, except for the
! fact that the inbound route-map is applied for the neighbor command pointing
! to R5 (10.1.15.5).
! R3 chooses its best path to 11/8 with NEXT_HOP of R4 (10.1.34.4), as a result
! of R3's assignment of LOCAL_PREF 300, which in turn was a result of the Update
```

Example 13-11 *Setting COMMUNITY, and Reacting to COMMUNITY to Set LOCAL_PREF (Continued)*

```
! from R4 listing 11/8 as COMMUNITY 1. R3's best route to 12/8 points to NEXT_HOP
! R5 (10.1.15.1), which happens to point back through R1, because R1 received an
! Update from R5 for 21/8 listing COMMUNITY 1, and then set LOCAL_PREF to 300.
R3# show ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 11.0.0.0         10.1.34.4       4294967294    300      0 4 1 33333 10 200 44 i
* i12.0.0.0         10.1.15.5       4294967294    100      0 5 1 33333 10 200 44 i
*>                  10.1.34.4       4294967294             0 4 1 33333 10 200 44 i
*>i21.0.0.0         10.1.15.5       4294967294    300      0 5 1 404 303 202 i
*                   10.1.34.4       4294967294    200      0 4 1 404 303 202 i
! R3 now lists its BGP table entries that have COMMUNITY settings that include
! 1 or 2. Note that both commands only list the routes learned directly from R4.
! If R1 had configured a neighbor 3.3.3.3 send-community command, R3 would have
! additional entries using COMMUNITY strings 1 and 2. However, for this design,
! the COMMUNITY strings do not need to be advertised to iBGP peers inside AS 123,
! as R1 and R3 have already reacted to the communities to set the LOCAL_PREF.
R3# show ip bgp community 1
BGP table version is 37, local router ID is 3.3.3.3
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
*> 11.0.0.0         10.1.34.4       4294967294    300      0 4 1 33333 10 200 44 i
R3# show ip bgp community 2 ¦ begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*  21.0.0.0         10.1.34.4       4294967294    200      0 4 1 404 303 202 i
! The COMMUNITY can be seen with the show ip bgp prefix command, as seen below.
! Note that the route learned from R1 (1.1.1.1) does not list a COMMUNITY, as R1
! did not configure a neighbor 3.3.3.3 send-community command.
R3# show ip bgp 21.0.0.0
BGP routing table entry for 21.0.0.0/8, version 35
Paths: (3 available, best #1, table Default-IP-Routing-Table)
Multipath: eBGP
  Advertised to update-groups:
     2
  5 1 404 303 202, (received & used)
    10.1.15.5 (metric 2681856) from 1.1.1.1 (1.1.1.1)
      Origin IGP, metric 4294967294, localpref 300, valid, internal, best
  4 1 404 303 202
    10.1.34.4 from 10.1.34.4 (4.4.4.4)
      Origin IGP, metric 4294967294, localpref 200, valid, external
      Community: 2
  4 1 404 303 202, (received-only)
    10.1.34.4 from 10.1.34.4 (4.4.4.4)
      Origin IGP, metric 4294967294, localpref 100, valid, external
      Community: 2
```

Matching COMMUNITY with Community Lists

Cisco originally created communities as a proprietary feature, treating the 32-bit COMMUNITY as a decimal value (as shown in Example 13-11). When the COMMUNITY PA was added to the BGP standard RFC 1997, the 32-bit COMMUNITY was formatted as AA:NN, where AA is a 16-bit number to potentially represent an ASN, and NN represents a value as set by that ASN. However, the COMMUNITY PA remained a 32-bit number.

Cisco routers can use either the original format or the RFC 1997 format for the COMMUNITY PA. By default, **show** commands list the decimal value; to use the AA:NN format, you should configure the global command **ip bgp-community new-format**. Also, the **set** command, as used with route maps, can use either the old decimal format or the newer AA:NN format; however, the absence or presence of the **ip bgp-community new-format** command dictates whether the output of a **show route-map** command lists the values as decimal or as AA:NN, respectively. For this reason, in practice it makes sense to choose and use a single format, typically the newer format today.

The COMMUNITY PA also supports multiple entries. For example, the **set community 10 20 30** command, applied within a route map, would actually create a COMMUNITY with all three values. In that case, any existing COMMUNITY value would be replaced with 10, 20, and 30. However, the **set community 10 20 30 additive** command would add the values to the existing COMMUNITY string.

As a result of the multi-entry COMMUNITY, and as a result of the literal ":" inside the COMMUNITY string when using the new format, Cisco IOS requires some more sophisticated matching capabilities as compared with IP ACLs. For example, community lists can list multiple values on the same **ip community-list** command; to match such a command, the COMMUNITY must include all the values. (The COMMUNITY values are unordered, so the order in which the values are listed in the community list does not matter.) Also, extended community lists (numbered 100–199) allow matching of the COMMUNITY PA with regular expressions. Table 13-15 summarizes some of the key points related to community lists.

Table 13-15 *Comparing Standard and Extended Community List*

KEY POINT

Feature	Standard	Extended
List numbers	1–99	100–99
Can match multiple communities in a single command?	Yes	Yes
Can match the COMMUNITY PA with regular expressions	No	Yes
More than 16 lines in a single list?	No	Yes

Example 13-12 shows a few example community lists just to show the logic. In the example, R4 has set multiple COMMUNITY values for prefixes 11/8 and 12/8. The **show ip bgp community-list**

list-number command is then used to show whether a match would be made. This command lists the entries of the BGP table that match the associated COMMUNITY PA, much like the **show ip bgp regex** command examines the AS_PATH PA.

Example 13-12 *Example of Matching with IP Community Lists*

```
R3# show ip community-list
Community standard list 2
     permit 0:1234
Community standard list 3
     permit 0:1212 8:9
Community (expanded) access list 111
    permit 0:12.*
! 11/8's COMMUNITY string is listed next, followed by 12/8's COMMUNITY string.
R3# show ip bgp 11.0.0.0 | include Community
      Community: 0:1212 0:1234 8:9 8:12 12:9 12:13
R3# show ip bgp 12.0.0.0 | include Community
      Community: 0:1212 8:12 8:13
! List 2 should match only 11/8, and not 12/8, as only 11/8 has 0:1234 as one of
! the values.
R3# show ip bgp community-list 2 | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 11.0.0.0         10.1.34.4       4294967294             0 4 1 33333 10 200 44 i
! Both 11/8 and 12/8 match the 0:1212 listed in list 3, but list 3 has two
! values configured. The list uses a logical AND between the entries, and only
! 11/8 has matching values for both communities.
R3# show ip bgp community-list 3 | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 11.0.0.0         10.1.34.4       4294967294             0 4 1 33333 10 200 44 i
! List 111 matches any COMMUNITY string with one entry beginning with 0:12,
! followed by any additional characters. 11/8 matches due to the 0:1234, and 12/8
! matches due to the 0:1212. COMMUNITY values 0:12, 0:123, and other would also
! have matched.
R3# show ip bgp community-list 111 | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 11.0.0.0         10.1.34.4       4294967294             0 4 1 33333 10 200 44 i
*> 12.0.0.0         10.1.34.4       4294967294             0 4 1 33333 10 200 44 i
```

Removing COMMUNITY Values

In some cases, a routing policy may need to remove one string from the COMMUNITY PA, or even delete the entire COMMUNITY PA. This can be accomplished with a route map as well, using the **set** command. Removing the entire COMMUNITY is relatively simple: include the **set community none** command in a **route-map** clause, and all routes matched by that clause will have their COMMUNITY PA removed. For example, Example 13-11 lists a **route-map react-to-comm** route map on each router. In that design, once the received COMMUNITY string on R1 and R3 was used to match the correct routes and set the LOCAL_PREF values, the COMMUNITY

PA was no longer needed. The revised route map in Example 13-13 simply removes the COMMUNITY at that point.

Example 13-13 *Removing the Entire COMMUNITY PA Once It Is No Longer Needed*

```
route-map react-to-comm permit 10
 match community 1
 set local-preference 300
 set community none
!
route-map react-to-comm permit 20
 match community 2
 set local-preference 200
 set community none
!
route-map react-to-comm permit 30
```

A route map can also remove individual COMMUNITY strings by using the **set comm-list** *community-list-number* **delete** command. This command tells the route map to match routes based on the community list, and then delete the COMMUNITY strings listed in the community list. (The referenced community list can contain only one COMMUNITY string per **ip community-list** command in this case.)

Filtering NLRI Using Special COMMUNITY Values

Routers can use route maps to filter NLRI from being added to the BGP table, or from being sent in Updates to other routers. These route maps can match a BGP route's COMMUNITY by using the **match community** {*standard-list-number* | *expanded-list-number* | *community-list-name* [**exact**]} command, which in turn references a community list.

Additionally, BGP includes several reserved values for the COMMUNITY PA that allow route filtering to occur, but with less effort than is required with community lists and route maps. These special COMMUNITY values, once set, affect the logic used by routers when making decisions about to which BGP peers they will advertise the route. The values are listed in Table 13-16.

Table 13-16 *COMMUNITY Values Used Specifically for NLRI Filtering*

KEY POINT

Name	Value	Meaning
NO_EXPORT	FFFF:FF01	Do not advertise outside this AS. It can be advertised to other confederation autonomous systems.
NO_ADVERT	FFFF:FF02	Do not advertise to any other peer.
LOCAL_AS[1]	FFFF:FF03	Do not advertise outside the local confederation sub-AS.

[1] LOCAL_AS is the Cisco term; RFC 1997 defines this value as NO_EXPORT_SUBCONFED.

A route with COMMUNITY NO_EXPORT is not advertised outside an AS. This value can be used to prevent an AS from being a transit AS for a set of prefixes. For example, a router in AS 1 could advertise an eBGP route into AS 2 with NO_EXPORT set. The routers inside AS 2 would then advertise the route inside AS 2 only. By not advertising the route outside AS 2, AS 2 cannot become a transit AS for that prefix. Note that the routers inside AS 2 do not have to configure a route map to prevent the route from exiting AS 2. However, the iBGP peers inside AS 2 must enable COMMUNITY using the **neighbor send-community** command.

The LOCAL_AS COMMUNITY value performs a similar function as NO_EXPORT, put just inside a single confederation sub-AS.

The NO_ADVERT COMMUNITY string may seem a bit unusual at first glance. However, it allows one router to advertise a prefix to a peer, with the intent that the peer will not advertise the route.

Finally, there are a few operational considerations to note regarding these COMMUNITY values. First, a router receiving any of these special communities can match them using an **ip community-list** command with obvious keywords for all three values. Additionally, a router can use a route map to match and then remove these COMMUNITY strings—in effect, ignoring the dictate to limit the scope of advertisement of the routes. Finally, routes with these settings can be seen with commands like **show ip bgp community no-export**, with similar options NO_ADVERT and LOCAL_AS.

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most Cisco Press *Exam Certification Guides*, this book does not repeat information listed in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 13-17 lists some of the RFCs for BGP whose concepts were covered in this chapter.

Table 13-17 *Protocols and Standards for Chapter 13*

Topic	Standard
BGP-4	RFC 1771
The NOPEER Community	RFC 3765
BGP Route Reflection	RFC 1966
BGP Communities	RFC 1997

Table 13-18 lists some of the relevant Cisco IOS commands related to the topics in this chapter.

Table 13-18 *Command Reference for Chapter 13*

Command	Command Mode and Description
bgp always-compare-med	BGP mode; tells the router to compare MED even if the neighboring ASNs are different
bgp bestpath med confed	BGP mode; tells the router to consider MED for choosing routes through different confederation sub-ASs
bgp bestpath med missing-as-worst	BGP mode; resets the default MED from 0 to the maximum ($2^{32} - 1$)
bgp default local-preference *number*	BGP mode; sets the default LOCAL_PREF value
bgp deterministic-med	BGP mode; tells IOS to process MED logic based on neighboring AS, rather than on the order in which the routes were learned
bgp maxas-limit *number*	BGP mode; tells the router to discard routes whose AS_PATH length exceeds this setting

Table 13-18 *Command Reference for Chapter 13 (Continued)*

Command	Command Mode and Description
clear ip bgp {***** | *neighbor-address* | *peer-group-name*} [**soft** [**in** | **out**]]	EXEC mode; clears the BGP process, or neighbors, optionally using soft reconfiguration
distribute-list *acl-number* | **prefix** *list-name* **in** | **out**	BGP mode; defines a BGP distribution list (ACL or prefix list) for filtering routes
ip as-path access-list *access-list-number* {**permit** | **deny**} *as-regexp*	Global config; creates entries in AS_PATH access lists used in matching existing AS_PATH values
ip bgp-community new-format	Global config; tells IOS to display and interpret the COMMUNITY PA in the RFC 1997 format, AA:NN
ip community-list {*standard* | **standard** *list-name* {**deny** | **permit**} [*community-number*] [*AA*:*NN*] [**internet**] [**local-AS**] [**no-advertise**] [**no-export**]} | {*expanded* | **expanded** *list-name* {**deny** | **permit**} *regexp*}	Global config; creates entries in a community list used in matching existing COMMUNITY values
maximum-paths *number*	BGP mode; sets the number of eBGP routes that can be added to the IP routing table
maximum-paths eibgp *number* [**import** *number*]	BGP mode; sets the number of eBGP and iBGP routes that can be added to the IP routing table when using MPLS
maximum-paths ibgp *number*	BGP mode; sets the number of iBGP routes that can be added to the IP routing table
neighbor {*ip-address* | *peer-group-name*} **distribute-list** {*access-list-number* | *expanded-list-number* | *access-list-name* | *prefix-list-name*} {**in** | **out**}	BGP mode; identifies a distribute list used to filter NLRI being sent to or received from the neighbor
neighbor {*ip-address* | *peer-group-name*} **filter-list** *access-list-number* {**in** | **out**}	BGP mode; identifies an AS_PATH access list used to filter NLRI by matching the AS_PATH PA
neighbor {*ip-address* | *peer-group-name*} **local-as** *as-number* [**no-prepend**]	BGP mode; defines an alternate ASN to be prepended in the AS_PATH of sent eBGP Updates, instead of the ASN listed in the **router bgp** command
neighbor {*ip-address* | *peer-group-name*} **prefix-list** {*prefix-list-name* | *clns-filter-expr-name* | *clns-filter-set-name*} {**in** | **out**}	BGP mode; identifies an IP prefix list used to filter NLRI being sent to or received from the neighbor

continues

Table 13-18 *Command Reference for Chapter 13 (Continued)*

Command	Command Mode and Description
neighbor {*ip-address* | *peer-group-name*} **remove-private-as**	BGP mode; used with eBGP peers, removes any private ASNs from the AS_PATH under certain conditions
neighbor {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}	BGP mode; defines a route map and direction for applying routing policies to BGP Updates
neighbor *ip-address* **route-reflector-client**	BGP mode; used on the RR server, identifies a neighbor as an RR client
neighbor {*ip-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**]	BGP mode; causes the router to include the COMMUNITY PA in Updates sent to this neighbor
neighbor {*ip-address* | *peer-group-name*} **soft-reconfiguration** [**inbound**]	BGP mode; enables soft reconfiguration of Updates
neighbor {*ip-address* | *peer-group-name*} **unsuppress-map** *route-map-name*	BGP mode; allows a router to identify previously suppressed routes and no longer suppress them
neighbor {*ip-address* | *peer-group-name*} **weight** *number*	BGP mode; sets the BGP weight for all routes learned from the neighbor
network *ip-address* **backdoor**	BGP mode; identifies a network as a backdoor route, considering it to have the same administrative distance as iBGP routes
show ip bgp quote-regexp *regexp*	EXEC mode; displays BGP table entries whose AS_PATH PA is matched by the stated regex
show ip bgp regexp *regexp*	EXEC mode; displays BGP table entries whose AS_PATH PA is matched by the stated regex
show ip community-list [*standard-community-list-number* | *extended-community-list-number* | *community-list-name*] [**exact-match**]	EXEC mode; lists the contents of configured IP community lists
show ip bgp community *community-number* [**exact**]	EXEC mode; lists BGP table entries that include the listed COMMUNITY
show ip bgp filter-list *access-list-number*	EXEC mode; lists the contents of AS_PATH access lists

Table 13-19 lists the route-map **match** and **set** commands pertinent to defining BGP routing policies.

Table 13-19 *Route-Map* **match** *and* **set** *Commands for BGP*

Command	Function
match as-path *path-list-number*	References an **ip as-path access-list** command to examine the AS_PATH
match community {*standard-list-number* | *expanded-list-number* | *community-list-name* [**exact**]}	References an **ip community-list** command to examine the COMMUNITY PA
match ip address {*access-list-number* [*access-list-number . . .* | *access-list-name . . .*]	References an IP access list to match based on NLRI
match ip address prefix-list *prefix-list-name* [*prefix-list-name . . .*]}	References an IP prefix list to match based on NLRI
match tag *tag-value* [*. . . tag-value*]	Matches a previously set route tag
set as-path prepend *as-path-string*	Adds the listed ASNs to the beginning of the AS_PATH
set comm-list *community-list-number* | *community-list-name* **delete**	Removes individual strings from the COMMUNITY PA as matched by the referenced community list
set community {*community-number* [**additive**] [*well-known-community*] | **none**}	Sets, replaces, adds to, or deletes the entire COMMUNITY
set ip next-hop *ip-address* [*. . . ip-address*] [**peer-address**]	With the peer address option, resets the NEXT_HOP PA to be the sender's IP address used to send Updates to a neighbor
set local-preference *number-value*	Sets the LOCAL_PREF PA
set metric *metric-value*	Inbound only; sets the MULTI_EXIT_DISC PA
set origin {**igp** | **egp** *as-number* | **incomplete**}	Sets the ORIGIN PA value
set weight *number*	Sets the proprietary administrative weight value

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

NLRI, soft reconfiguration, AS_PATH access list, AS_PATH prepending, regular expression, AS_SEQUENCE, AS_SET, AS_CONFED_SET, AS_CONFED_SEQ, well-known mandatory, well-known discretionary, optional transitive, optional nontransitive, AS_PATH, NEXT_HOP, AGGREGATOR, ATOMIC AGGREGATE, ORIGINATOR_ID, CLUSTER_LIST, ORIGIN, administrative weight, LOCAL_PREF, AS_PATH length, MULTI_EXIT_DISC (MED), Neighbor Type, BGP decision process, private AS, COMMUNITY, LOCAL_AS, NO_EXPORT, NO_ADVERT, NO_EXPORT_SUBCONFED

Further Reading

- *Routing TCP/IP*, Volume II, by Jeff Doyle and Jennifer DeHaven Carrol
- *Cisco BGP-4 Command and Configuration Handbook*, by William R. Parkhurst
- *Internet Routing Architectures*, by Bassam Halabi
- *Troubleshooting IP Routing Protocols*, by Zaheer Aziz, Johnson Liu, Abe Martey, and Faraz Shamim

- Most every reference reached from Cisco's BGP support page at http://www.cisco.com/en/US/partner/tech/tk365/tk80/tsd_technology_support_sub-protocol_home.html. Requires a CCO username/password.
- For the oddities of BGP table sequence impacting the MED-related best path choice, refer to the following Cisco resource: http://www.cisco.com/en/US/partner/tech/tk365/technologies_tech_note09186a0080094925.shtml

Part IV: Quality of Service

Chapter 14 Classification and Marking

Chapter 15 Congestion Management and Avoidance

Chapter 16 Shaping and Policing

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- Quality of Service (QoS)
  - — Traffic Classification

CHAPTER 14

Classification and Marking

The goal of classification and marking tools is to simplify the classification process of other QoS tools by performing complicated classification steps as few times as possible. For instance, a classification and marking tool might examine the source IP address of packets, incoming Class of Service (CoS) settings, and possibly TCP or UDP port numbers. Packets matching all those fields may have their IP Precedence (IPP) or DiffServ Code Points (DSCPs) field marked with a particular value. Later, other QoS tools—on the same router/switch or a different one—can simply look for the marked field when making a QoS decision, rather than having to perform the detailed classification again before taking the desired QoS action.

"Do I Know This Already?" Quiz

Table 14-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 14-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Fields that Can Be Marked for QoS Purposes	1–4	
Cisco Modular QoS CLI	5–7	
Classification and Marking Tools	8–9	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. According to the DiffServ RFCs, which PHB defines a set of three DSCPs in each service class, with different drop characteristics for each of the three DSCP values?
   a. Expedited Forwarding
   b. Class Selector
   c. Assured Forwarding
   d. Multi-class-multi-drop

2. Which of the following are true about the location of DSCP in the IP header?
   a. High-order 6 bits of ToS byte/DS field
   b. Low-order 6 bits of ToS byte
   c. Middle 6 bits of ToS byte
   d. Its first 3 bits overlap with IP Precedence
   e. Its last 3 bits overlap with IP Precedence

3. Imagine that a packet is marked with DSCP CS3. Later, a QoS tool classifies the packet. Which of the following classification criteria would match the packet, assuming the marking had not been changed from the original CS3 marking?
   a. Match on DSCP CS3
   b. Match on precedence 3
   c. Match on DSCP AF32
   d. Match on DSCP AF31
   e. Match on DSCP decimal 24

4. Imagine that a packet is marked with AF31. Later, a QoS tool classifies the packet. Which of the following classification criteria would match the packet, assuming the marking had not been changed from the original AF31 marking?
   a. Match on DSCP CS3
   b. Match on precedence 3
   c. Match on DSCP 24
   d. Match on DSCP 26
   e. Match on DSCP 28

5. Examine the following output from a router that shows a user adding configuration to a router. Which of the following statements is true about the configuration?

```
Router(config)# class-map fred
Router(config-cmap)# match dscp EF
Router(config-cmap)# match access-group 101
```

   a. Packets that match both DSCP EF and ACL 101 will match the class.

   b. Packets that match either DSCP EF or ACL 101 will match the class.

   c. Packets that match ACL 101 will match the class, because the second **match** command replaces the first.

   d. Packets will only match DSCP EF because the first match exits the class map.

6. Router R1 is configured with the following three class maps. Which class map(s) would match an incoming frame whose CoS field is set to 3, IP Precedence is set to 2, and DSCP is set to AF21?

```
class-map match-all c1
 match cos 3 4
class-map match-any c2
 match cos 2 3
 match cos 1
class-map match-all c3
 match cos 3 4
 match cos 2
```

   a. c1

   b. c2

   c. c3

   d. All of these answers are correct.

7. Examine the following example of commands typed in configuration mode to create a class map. Assuming that the **class fred** command was used inside a policy map, and the policy map was enabled on an interface, which of the following would be true with regard to packets classified by the class map?

```
Router(config)# class-map fred
Router(config-cmap)# match ip dscp ef
Router(config-cmap)# match ip dscp af31
```

   a. Match packets with both DSCP EF and AF31

   b. Match packets with either DSCP EF or AF31

   c. Match all packets that are neither EF or AF31

   d. Match no packets

   e. Match packets with precedence values of 3 and 5

8. The **service-policy output fred** command is found in router R1's configuration under Frame Relay subinterface s0/0.1. Which of the following could be true about this CB Marking policy map?

   a. The policy map can classify packets using class maps that match based on the DE bit.

   b. The policy map can refer to class maps that match based on DSCP.

   c. The policy map can set CoS.

   d. The policy map can set CLP.

   e. The policy map can set DE.

9. Which of the following is true regarding the listed configuration steps?

```
Router(config)# class-map barney
Router(config-cmap)# match protocol http url "this-here.jpg"
Router(config-cmap)# policy-map fred
Router(config-pmap)# class barney
Router(config-pmap-c)# set dscp af21
Router(config-pmap-c)# interface fa0/0
Router(config-if)# service-policy output fred
```

   a. If not already configured, the **ip cef** global command is required.

   b. The configuration does not use NBAR because the **match nbar** command was not used.

   c. The **service-policy** command would be rejected because **match protocol** is not allowed as an output function.

   d. None of these answers is correct.

Foundation Topics

This chapter has three major sections. The chapter begins by examining the fields that can be marked by the classification and marking (C&M) tools. Next, the chapter covers the mechanics of the Cisco IOS Modular QoS CLI (MQC), which is used by all the IOS QoS tools that begin with the words "Class-Based." Finally, the C&M tools are covered, with most of the content focused on the most important C&M tool, Class-Based Marking (CB Marking).

Fields That Can Be Marked for QoS Purposes

The IP header, LAN trunking headers, Frame Relay header, and ATM cell header all have at least one field that can be used to perform some form of QoS marking. This section lists and defines those fields, with the most significant coverage focused on the IP header IP Precedence (IPP) and Differentiated Services Code Point (DSCP) fields.

IP Precedence and DSCP Compared

The IP header is defined in RFC 791, including a 1-byte field called the Type of Service (ToS) byte. The ToS byte was intended to be used as a field to mark a packet for treatment with QoS tools. The ToS byte itself was further subdivided, with the high-order 3 bits defined as the *IP Precedence* (*IPP)* field. The complete list of values from the ToS byte's original IPP 3-bit field, and the corresponding names, is provided in Table 14-2.

Table 14-2 *IP Precedence Values and Names*

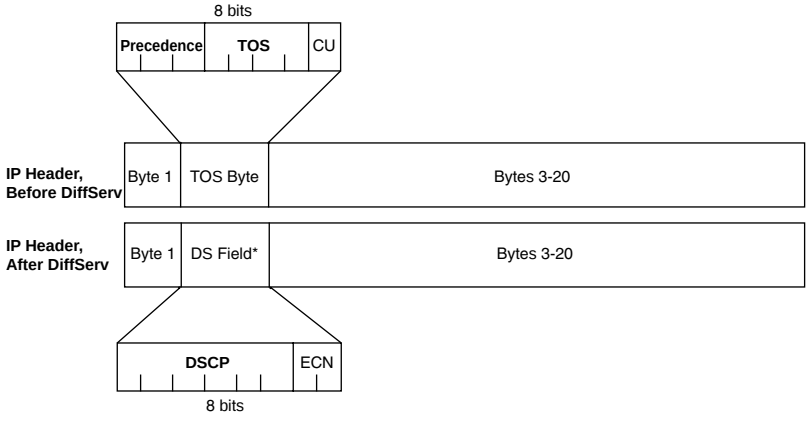
KEY POINT

Name	Decimal Value	Binary Value
Routine	Precedence 0	**000**
Priority	Precedence 1	**001**
Immediate	Precedence 2	**010**
Flash	Precedence 3	**011**
Flash Override	Precedence 4	**100**
Critic/Critical	Precedence 5	**101**
Internetwork Control	Precedence 6	**110**
Network Control	Precedence 7	**111**

Bits 3 through 6 of the ToS byte included flag fields that were toggled on or off to imply a particular QoS service. The final bit (bit 7) was not defined in RFC 791. The flags were not used very often, so in effect, the ToS byte's main purpose was to hold the 3-bit IPP field.

A series of RFCs collectively called *Differentiated Services (DiffServ)* came along later. DiffServ needed more than 3 bits to mark packets, so DiffServ standardized a redefinition of the ToS byte. The ToS byte itself was renamed the *Differentiated Services (DS) field*, and IPP was replaced with a 6-bit field (high-order bits 0–5) called the *Differentiated Services Code Point (DSCP)* field. Later, RFC 3168 defined the low-order 2 bits of the DS field for use with the QoS *Explicit Congestion Notification (ECN)* feature. Figure 14-1 shows the ToS byte's format with the pre-DiffServ and post-DiffServ definition of the field.

Figure 14-1 *IP ToS Byte and DS Field Compared*



C&M tools often mark DSCP or IPP because the IP packet remains intact as it is forwarded throughout an IP network. The other possible marking fields reside inside Layer 2 headers, which means the headers are discarded when forwarded by a Layer 3 process. Thus, the latter cannot be used to carry QoS markings beyond the current hop.

DSCP Settings and Terminology

Several DiffServ RFCs suggest a set of values to use in the DSCP field and an implied meaning for those settings. For instance, RFC 2598 defines a DSCP of decimal 46, with a name *Expedited Forwarding (EF)*. According to that RFC, packets marked as EF should be given queuing preference so that they experience minimal latency, but the packets should be policed to prevent them from taking over a link and preventing any other types of traffic from exiting an interface during periods when this high-priority traffic reaches or exceeds the interface bandwidth. These suggested settings, and the associated QoS behavior recommended when using each setting, are called *Per-Hop Behaviors (PHBs)* by DiffServ. (The particular example listed in this paragraph is called the Expedited Forwarding PHB.)

The Class Selector PHB and DSCP Values

IPP overlaps with the first 3 bits of the DSCP field because the DS field is simply a redefinition of the original ToS byte in the IP header. Because of this overlap, RFC 2475 defines a set of DSCP values and PHBs, called *Class Selector (CS)* PHBs, that provide backward compatibility with IPP. A C&M feature can set a CS DSCP value, and if another router or switch just looks at the IPP field, the value will make sense from an IPP perspective. Table 14-3 lists the CS DSCP names and values, and the corresponding IPP values and names.

Table 14-3 *Default and Class Selector DSCP Values*

KEY POINT

DSCP Class Selector Names	Binary DSCP Values	IPP Binary Values	IPP Names
Default/CS0*	**000**000	000	Routine
CS1	**001**000	001	Priority
CS2	**010**000	010	Immediate
CS3	**011**000	011	Flash
CS4	**100**000	100	Flash Override
CS5	**101**000	101	Critic/Critical
CS6	**110**000	110	Internetwork Control
CS7	**111**000	111	Network Control

*The terms "CS0" and "Default" both refer to a binary DSCP of 000000, but most Cisco IOS commands allow only the keyword "default" to represent this value.

Besides defining eight DSCP values and their text names, the CS PHB also suggests a simple set of QoS actions that should be taken based on the CS values. The CS PHB simply states that packets with larger CS DSCPs should be given better queuing preference than packets with lower CS DSCPs.

The Assured Forwarding PHB and DSCP Values

The *Assured Forwarding (AF)* PHB (RFC 2597) defines four classes for queuing purposes, along with three levels of drop probability inside each queue. To mark packets and distinguish into which of four queues a packet should be placed, along with one of three drop priorities inside each queue, the AF PHB defines 12 DSCP values and their meanings. The names of the AF DSCPs conform to the following format:

AF*xy*

where *x* implies one of four queues (values 1 through 4), and *y* implies one of three drop priorities (values 1 through 3).

The AF PHB suggests that the higher the value of *x* in the DSCP name AF*xy*, the better the queuing treatment a packet should get. For example, packets with AF11 DSCPs should get worse queuing treatment than packets with AF23 DSCP values. Additionally, the AF PHB suggests that the higher the value of *y* in the DSCP name AF*xy*, the worse the drop treatment for those packets. (Treating a packet worse for drop purposes means that the packet has a higher probability of being dropped.) For example, packets with AF11 DSCPs should get better drop treatment than packets with AF23 DSCP values. Table 14-4 lists the names of the DSCP values, the queuing classes, and the implied drop likelihood.

Table 14-4 *Assured Forwarding DSCP Values—Names, Binary Values, and Decimal Values*

KEY POINT

Queue Class	Low Drop Probability	Medium Drop Probability	High Drop Probability
	Name/Decimal/Binary	**Name/Decimal/Binary**	**Name/Decimal/Binary**
1	AF11 / 10 / 001010	AF12 / 12 / 001100	AF13 / 14 / 001110
2	AF21 / 18 / 010010	AF22 / 20 / 010100	AF23 / 22 / 010110
4	AF31 / 26 / 011010	AF32 / 28 / 011100	AF33 / 30 / 011110
5	AF41 / 34 / 100010	AF42 / 36 / 100100	AF43 / 38 / 100110

The text AF PHB names do not follow the "bigger-is- better" logic in all cases. For example, the name AF11 represents a decimal value of 10, and the name AF13 represents a decimal DSCP of 14. However, AF11 is "better" than AF13, because AF11 and AF13 are in the same queuing class, but AF11 has a lower probability of being dropped than AF13.

The binary version of the AF DSCP values shows the patterns of the values. The first 3 bits of the binary DSCP values imply the queuing class (bits 0 through 2), and the next 2 bits (bits 3 and 4) imply the drop preference. As a result, queuing tools that operate only on IPP can still react to the AF DSCP values, essentially making the AF DSCPs backward compatible with non-DiffServ nodes for queuing purposes.

KEY POINT

NOTE To convert from the AF name to the decimal equivalent, you can use a simple formula. If you think of the AF values as AF*xy*, the formula is:

$8x + 2y$ = decimal value

For example, AF41 gives you a formula of (8 * 4) + (2 * 1) = 34.

The Expedited Forwarding PHB and DSCP Values

RFC 2598 defines the *Expedited Forwarding (EF)* PHB, which was described briefly in the introduction to this section. This RFC defines a very simple pair of PHB actions:

- Queue EF packets so that they get scheduled quickly, to give them low latency.

- Police the EF packets so that they do not consume all bandwidth on the link or starve other queues.

The DSCP value defined for EF is named EF, with decimal value 46, binary value 101110.

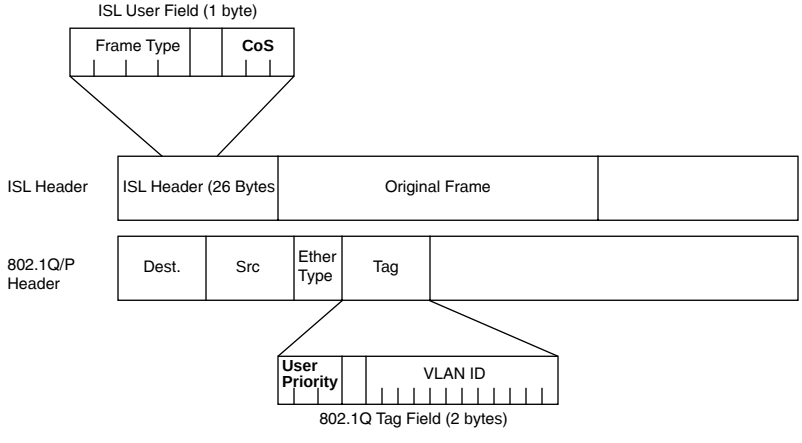
Non-IP Header Marking Fields

As IP packets pass through an internetwork, the packet is encapsulated in a variety of other headers. In several cases, these other headers have QoS fields that can be used for classification and marking.

Ethernet LAN Class of Service

Ethernet supports a 3-bit QoS marking field, but the field only exists when the Ethernet header includes either an 802.1Q or ISL trunking header. IEEE 802.1Q defines its QoS field as the 3 most-significant bits of the 2-byte *Tag Control* field, calling the field the *user-priority bits*. ISL defines the 3 least-significant bits from the 1-byte *User* field, calling this field the *Class of Service (CoS)*. Generally speaking, most people (and most IOS commands) refer to these fields as *CoS*, regardless of the type of trunking. Figure 14-2 shows the general location of the CoS field inside ISL and 802.1P headers.

Figure 14-2 *LAN CoS Fields*



WAN Marking Fields

Frame Relay and ATM support a single bit that can be set for QoS purposes, but these single bits are intended for a very strict use related to drop probability. Frames or cells with these bits set to 1 are considered to be better candidates to be dropped than frames or cells without the bit set to 1. Named the Frame Relay *Discard Eligibility (DE)* bit and the ATM *Cell Loss Priority (CLP)* bit, these bits can be set by a router, or by an ATM or Frame Relay switch. Router and switch drop

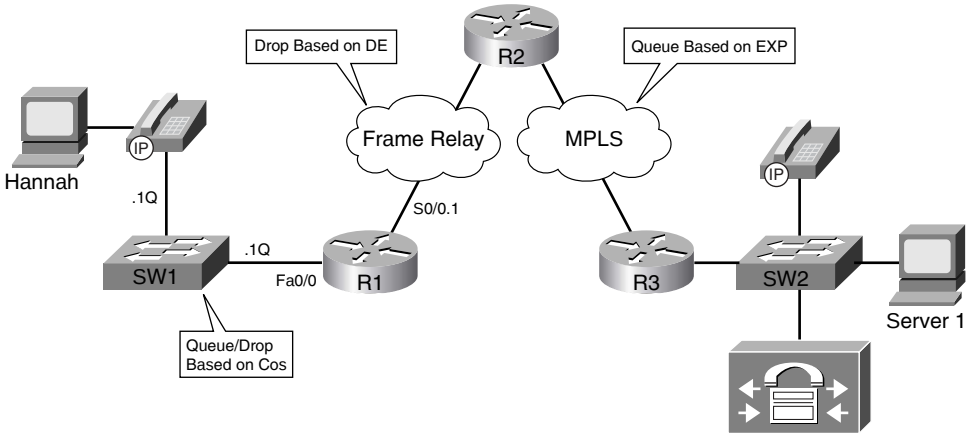
features can then be configured to more aggressively drop frames and cells that have the DE or CLP bit set, respectively.

MPLS defines a 3-bit field called the *MPLS Experimental (EXP)* bit that is intended for general QoS marking. Often, C&M tools are used on the edge of MPLS networks to remap DSCP or IPP values to MPLS Experimental bit values to provide QoS inside the MPLS network.

Locations for Marking and Matching

Figure 14-3 shows a sample network, with notes about the locations of the QoS fields.

Figure 14-3 *Sample Network Showing Non-IP Markable QoS Fields*



In such a network, the IPP and DSCP inside the IP packet remain intact from end to end. However, some devices may not be able to look at the IPP or DSCP fields, and some may find it more convenient to look at some other header field. For instance, an MPLS Label Switch Router (LSR) inside the MPLS cloud may be configured to make QoS decisions based on the 3-bit MPLS EXP field in the MPLS label, but unable to look at the encapsulated IP header and DSCP field. In such cases, QoS tools may need to be configured on edge devices to look at the DSCP and then mark a different field.

The non-IP header markable fields exist in only parts of the network. As a result, those fields can be used for classification or marking only on the appropriate interfaces. The rules for where these fields (CoS, DE, CLP, EXP) can be used are as follows:

KEY POINT

- **For classification**—On ingress only, and only if the interface supports that particular header field
- **For marking**—On egress only, and only if the interface supports that particular header field

For example, if CB Marking were to be configured on R1's fa0/0.1 802.1Q subinterface, it could classify incoming frames based on their CoS values, and mark outgoing frames with a CoS value. However, on ingress, it could not mark CoS, and on egress, it could not classify based on CoS. Similarly, on that same fa0/0.1 subinterface, CB Marking could neither classify nor mark based on a DE bit, CLP bit, or MPLS EXP bits, because these headers never exist on Ethernet interfaces.

Table 14-5 summarizes the QoS marking fields.

Table 14-5 *Marking Field Summary*

KEY POINT

Field	Location	Length
IP Precedence (IPP)	IP header	3 bits
IP DSCP	IP header	6 bits
DS field	IP header	1 byte
ToS byte	IP header	1 byte
CoS	ISL and 802.1Q header	3 bits
Discard Eligible (DE)	Frame Relay header	1 bit
Cell Loss Priority (CLP)	ATM cell header	1 bit
MPLS Experimental	MPLS header	3 bits

Cisco Modular QoS CLI

For many years and over many IOS releases, Cisco added QoS features and functions, each of which used its own separate set of configuration and exec commands. Eventually, the number of different QoS tools and different QoS commands got so large that QoS configuration became a big chore. Cisco created the *Modular QoS CLI (MQC)* to help resolve these problems, by defining a common set of configuration commands to configure many QoS features in a router or switch.

MQC is not a totally new CLI, different from IOS configuration mode, for configuring QoS. Rather, it is a method of categorizing IOS classification, marking, and related actions into logical groupings to unify the command-line interface. MQC defines a new set of configuration commands—commands that are typed in using the same IOS CLI, in configuration mode. However, once you understand MQC, you typically need to learn only one new command to know how to configure any additional MQC-based QoS tools. You can identify MQC-based tools by the name of the tool; they all begin with the phrase "Class-Based" (abbreviated CB for this discussion). These tools include CB Marking, CB Weighted Fair Queuing (CBWFQ), CB Policing, CB Shaping, and CB Header Compression.

The Mechanics of MQC

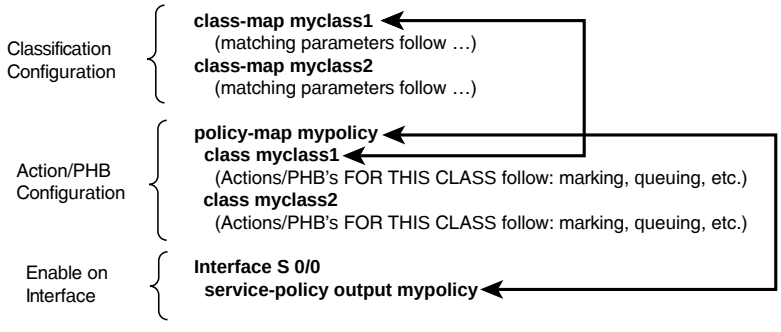
MQC separates the classification function of a QoS tool from the action (PHB) that the QoS tool wants to perform. To do so, there are three major commands with MQC, with several subordinate commands:

- The **class-map** command defines the matching parameters for classifying packets into service classes.
- The PHB actions (marking, queuing, and so on) are configured under a **policy-map** command.
- The policy map is enabled on an interface by using a **service-policy** command.

Figure 14-4 shows the general flow of commands.

Figure 14-4 *MQC Commands and Their Correlation*

KEY POINT



In Figure 14-4, the network's QoS policy calls for treating packets in one of two categories, called *QoS service classes*. (The actual types of packets that are placed into each class are not shown, to keep the focus on the general flow of how the main commands work together.) Classifying packets into two classes calls for the use of two **class-map** commands. Each **class-map** command would be followed by a **match** subcommand, which defines the actual parameters that are compared to the frame/packet header contents to match packets for classification.

For each class, some QoS action (PHB) needs to be performed; this action is configured using the **policy-map** command. Under a single policy map, multiple classes can be referenced; in Figure 14-4, the two classes myclass1 and myclass2. Inside the single policy called mypolicy, under each of the two classes myclass1 and myclass2, you can configure separate QoS actions. For instance, you could apply different markings to packets in myclass1 and myclass2 at this point. Finally, when the **service-policy** command is applied to an interface, the QoS features are enabled either inbound or outbound on that interface.

The next section takes a much closer look at packet classification using class maps. Most of the discussion of policy maps will be included when specifically covering CB Marking configuration later in the chapter.

Classification Using Class Maps

MQC-based tools classify packets using the **match** subcommand inside an MQC class map. The following list details the rules surrounding how class maps work for matching and classifying packets:

KEY POINT

- The **match** command has many options for matching packets, including QoS fields, ACLs, and MAC addresses. (See Table 14-10 in the "Foundation Summary" section for a reference.)
- Class-map names are case sensitive.
- The **match protocol** command means that IOS uses Network Based Application Recognition (NBAR) to perform that match.
- The **match any** command matches any packet—in other words, any and all packets.

Example 14-1 shows a simple CB Marking configuration, with comments focused on the classification configuration. Note that the names and logic match Figure 14-4.

Example 14-1 *Basic CB Marking Example*

```
! CEF is required for CB Marking. Without it, the class map and policy map
! configuration would be allowed, but the service-policy command would be rejected.
ip cef
! The first class map matches all UDP/RTP packets with UDP ports between 16384 and
! 32767 (the 2nd number is added to the first to get the end of the range.) The
! second class map matches any and all packets.
class-map match-all msclass1
  match ip rtp 16384 16383
class-map match-all myclass2
  match any
! The policy map calls each of the two class maps for matching. The set command
! implies that the PHB is marking, meaning that this is a CB Marking config.
policy-map mypolicy
  class myclass1
   set dscp EF
  class myclass2
   set dscp default
! The policy map processes packets leaving interface fa0/0.
interface Fastethernet0/0
 service-policy output mypolicy
```

With Example 14-1, each packet leaving interface fa0/0 will match one of the two classes. Because the policy map uses a **set dscp** command in each class, and all packets happen to match either myclass1 or myclass2, each packet will leave the interface marked either with DSCP EF (decimal 46) or default (decimal 0). (If the matching logic was different and some packets match neither myclass1 nor myclass2, those packets would not be marked, and would retain their existing DSCP values.)

Using Multiple match Commands

In some cases, a class map may need to examine multiple items in a packet to decide whether the packet should be part of that class. Class maps can use multiple **match** commands, and even nest class maps inside other class maps, to achieve the desired combination of logic. The following list summarizes the key points regarding these more complex matching options:

KEY POINT

- Up to four (CoS and IPP) or eight (DSCP) values can be listed on a single **match cos**, **match precedence**, or **match dscp** command, respectively. If any of the values are found in the packet, the statement is matched.
- If a class map has multiple **match** commands in it, the **match-any** or **match-all** (default) parameter on the **class-map** command defines whether a logical OR or a logical AND (default) is used between the **match** commands, respectively.
- The **match class** *name* command refers to another class map by name, nesting the named class map's matching logic; the **match class** *name* command is considered to match if the referenced **class-map** also results in a match.

Example 14-2 shows several examples of this more complicated matching logic, with notations inside the example of what must be true for a class map to match a packet.

Example 14-2 *Complex Matching with Class Maps*

```
! class-map example1 uses match-all logic (default), so this class map matches
! packets that are permitted by ACL 102, and that also have an IP precedence of 5.
class-map match-all example1
  match access-group 102
  match precedence 5
! class-map example2 uses match-any logic, so this class map matches packets that
! are permitted by ACL 102, or have DSCP AF21, or both.
class-map match-any example2
  match access-group 102
  match dscp AF21
! class-map example3 matches no packets, due to a common mistake—the two match
! commands use a logical AND between them due to the default match-all argument, meaning
! that a single packet must have DSCP 0 and DSCP 1, which is impossible. class-map example4
! shows how to correctly match either DSCP 0 or 1.
class-map match-all example3
  match dscp 0
  match dscp 1
!
class-map match-any example4
  match dscp 0 1
! class-map i-am-nesting refers to class-map i-am-nested through the match class
! i-am-nested command. The logic is explained after the example.
class-map match-all i-am-nested
  match access-group 102
```

Example 14-2 *Complex Matching with Class Maps (Continued)*

```
  match precedence 5
!
class-map match-any i-am-nesting
  match class i-am-nested
  match cos 5
```

The trickiest part of Example 14-2 is how the class maps can be nested, as shown at the end. **class-map i-am-nesting** uses OR logic between its two **match** commands, meaning "I will match if the CoS is 5, or if **class-map i-am-nested** matches the packet, or both." When combined with the match-all logic of the **i-am-nested** class map, the logic matches the following packets/frames:

> Packets that are permitted by ACL 102, AND marked with precedence 5
> or
> frames with CoS 5

Classification Using NBAR

NBAR classifies packets that are normally difficult to classify. For instance, some applications use dynamic port numbers, so a statically configured **match** command, matching a particular UDP or TCP port number, simply could not classify the traffic. NBAR can look past the UDP and TCP header, and refer to the host name, URL, or MIME type in HTTP requests. (This deeper examination of the packet contents is sometimes called *deep packet inspection*.) NBAR can also look past the TCP and UDP headers to recognize application-specific information. For instance, NBAR allows recognition of different Citrix application types, and allows searching for a portion of a URL string.

NBAR itself can be used for a couple of different purposes. Independent of QoS features, NBAR can be configured to keep counters of traffic types and traffic volume for each type. For QoS, NBAR can be used by CB Marking to match difficult-to-match packets. Whenever the MQC **match protocol** command is used, IOS is using NBAR to match the packets. Table 14-6 lists some of the more popular uses of the **match protocol** command and NBAR.

Table 14-6 *Popular Fields Matchable by CB Marking Using NBAR*

Field	Comments
RTP audio versus video	RTP uses even-numbered UDP ports from 16,384 to 32,768. The odd-numbered port numbers are used by RTCP for call control traffic. NBAR allows matching the even-numbered ports only, for classification of voice payload into a different service class from that used for voice signaling.
Citrix applications	NBAR can recognize different types of published Citrix applications.

continues

Table 14-6 *Popular Fields Matchable by CB Marking Using NBAR (Continued)*

Field	Comments
Host name, URL string, MIME type	NBAR can also match URL strings, including the host name and the MIME type, using regular expressions for matching logic.
Peer-to-peer applications	NBAR can find file-sharing applications like KaZaa, Morpheus, Grokster, and Gnutella.

Classification and Marking Tools

The final major section of this chapter covers CB Marking, with a brief mention of a few other, less popular marking tools.

Class-Based Marking (CB Marking) Configuration

As with the other QoS tools whose names begin with the phrase "Class-Based," you will use MQC commands to configure CB Marking. The following list highlights the key points regarding CB Marking configuration and logic:

KEY POINT

- CB Marking requires CEF (enabled using the **ip cef** global command).
- Packets are classified based on the logic in MQC class maps.
- An MQC policy map refers to one or more class maps using the **class** *class-map-name* command; packets classified into that class are then marked.
- CB Marking is enabled for packets either entering or exiting an interface using the MQC **service-policy in | out** *policy-map-name* interface subcommand.
- A CB Marking policy map is processed sequentially; once a packet has matched a class, it is marked based on the **set** command(s) defined for that class.
- You can configure multiple **set** commands in one class to set multiple fields; for example, to set both DSCP and CoS.
- Packets that do not explicitly match a defined class are considered to have matched a special class called *class-default*.
- For any class inside the policy map for which there is no set command, packets in that class are not marked.

Table 14-7 lists the syntax of the CB Marking **set** command, showing the familiar fields that can be set by CB Marking. Table 14-8 lists the key **show** commands available for CB Marking.

Table 14-7 **set** *Configuration Command Reference for CB Marking*

KEY POINT

Command	Function
set [**ip**] **precedence** *ip-precedence-value*	Marks the value for IP Precedence for IPv4 and IPv6 packets if the **ip** parameter is omitted; sets only IPv4 packets if the **ip** parameter is included
set [**ip**] **dscp** *ip-dscp-value*	Marks the value for IP DSCP for IPv4 and IPv6 packets if the **ip** parameter is omitted; sets only IPv4 packets if the **ip** parameter is included
set cos *cos-value*	Marks the value for CoS
set qos-group *group-id*	Marks the group identifier for the QoS group
set atm-clp	Sets the ATM CLP bit
set fr-de	Sets the Frame Relay DE bit

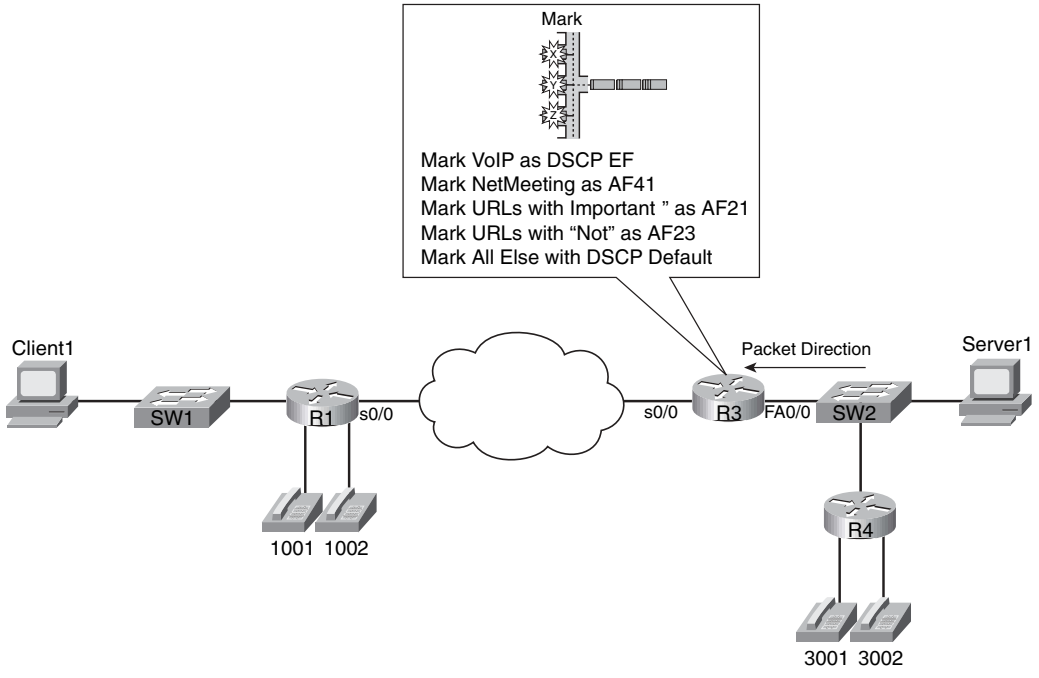
Table 14-8 *EXEC Command Reference for CB Marking*

Command	Function
show policy-map *policy-map-name*	Lists configuration information about a policy map
show policy-map *interface-spec* [*input* | *output*] [**class** *class-name*]	Lists statistical information about the behavior of a policy map when enabled on an interface

CB Marking Example

The first CB Marking example uses the network shown in Figure 14-5. Traffic was generated in the network to make the **show** commands more meaningful. Two G.711 voice calls were completed between R4 and R1 using *Foreign Exchange Station (FXS)* cards on these two routers, with *Voice Activity Detection (VAD)* disabled. Client1 performed an FTP get of a large file from Server1, and downloaded two large HTTP objects, named important.jpg and not-so.jpg. Finally, Client1 and Server1 held a Microsoft NetMeeting conference, using G.723 for the audio and H.263 for the video.

Figure 14-5 *Sample Network for CB Marking Examples*



The following criteria define the requirements for marking the various types of traffic for Example 14-3:

- VoIP payload is marked with DSCP EF.
- NetMeeting video traffic is marked with DSCP AF41.
- Any HTTP traffic whose URL contains the string "important" anywhere in the URL is marked with AF21.
- Any HTTP traffic whose URL contains the string "not-so" anywhere in the URL is marked with AF23.
- All other traffic is marked with DSCP Default (0).

Example 14-3 lists the annotated configuration, including the appropriate **show** commands.

Example 14-3 *CB Marking Example 1, with* **show** *Command Output*

```
ip cef
! Class map voip-rtp uses NBAR to match all RTP audio payload, but not the video
! or the signaling.
class-map voip-rtp
 match protocol rtp audio
```

Example 14-3 *CB Marking Example 1, with* **show** *Command Output (Continued)*

```
! Class map http-impo matches all packets related to downloading objects whose
! name contains the string "important," with any text around it. Similar logic
! is used for class-map http-not.
class-map http-impo
 match protocol http url "*important*"
!
class-map http-not
 match protocol http url "*not-so*"
! Class map NetMeet matches two RTP subtypes—one for G.723 audio (type 4) and
! one for H.263 video (type 34). Note the match-any logic so that if either is
! true, a match occurs for this class map.
class-map match-any NetMeet
 match protocol rtp payload-type 4
 match protocol rtp payload-type 34
! policy-map laundry-list calls each of the class maps. Note that the order
! listed here is the order in which the class commands were added to the policy
! map.
policy-map laundry-list
 class voip-rtp
  set ip dscp EF
class NetMeet
  set ip dscp AF41
class http-impo
  set ip dscp AF21
class http-not
  set ip dscp AF23
class class-default
  set ip DSCP default
! Above, the command class class-default is only required if some nondefault action
! needs to be taken for packets that are not explicitly matched by another class.
! In this case, packets not matched by any other class fall into the class-default
! class, and are marked with DSCP Default (decimal 0). Without these two commands,
! packets in this class would remain unchanged.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Below, the policy map is enabled for input packets on fa0/0.
 interface Fastethernet 0/0
 service-policy input laundry-list
! The command show policy-map laundry-list simply restates the configuration.
R3# show policy-map laundry-list
  Policy Map laundry-list
    Class voip-rtp
      set ip dscp 46
    Class NetMeet
      set ip dscp 34
    Class http-impo
      set ip dscp 18
    Class http-not
```

continues

Example 14-3 *CB Marking Example 1, with* **show** *Command Output (Continued)*

```
      set ip dscp 22
    Class class-default
      set ip dscp 0
! The command show policy-map interface lists statistics related to MQC features.
! Several stanzas of output were omitted for brevity.
R3# show policy-map interface fastethernet 0/0 input
 Fastethernet0/0

  Service-policy input:     laundry-list

    Class-map: voip-rtp (match-all)
      35268 packets, 2609832 bytes
      5 minute offered rate     59000 bps, drop rate 0 bps
      Match: protocol rtp audio
      QoS Set
        ip dscp 46
              Packets marked 35268

    Class-map: NetMeet (match-any)
      817 packets, 328768 bytes
      5 minute offered rate     19000 bps, drop rate 0 bps
      Match: protocol rtp payload-type 4
             protocol rtp payload-type 34
      QoS Set
        ip dscp 34
              Packets marked 817

! omitting stanza of output for class http-impo
! omitting stanza of output for class http-not

    Class-map: class-default (match-all)
      33216 packets, 43649458 bytes
      5 minute offered rate     747000 bps, drop rate 0 bps
      Match: any
      QoS Set
        ip dscp 0
Packets marked 33301
```

Example 14-3 includes several different classification options using the **match** command, including the matching of Microsoft NetMeeting traffic. NetMeeting uses RTP for the video flows, and by default uses G.723 for audio and H.323 for video. To match both the audio and video for NetMeeting, a class map that matches either of the two RTP payload subtypes for G.723 and H.263 is needed. So, class map **NetMeet** uses match-any logic, and matches on RTP payload types 4 (G.723) and 34 (H.263). (For more background information on RTP payload types, refer to http://www.cisco.com/en/US/products/ps6616/products_white_paper09186a0080110040.shtml.)

The **show policy-map interface** command provides statistical information about the number of packets and bytes that have matched each class in the policy maps. The generic syntax is as follows:

```
show policy-map interface interface-name [vc [vpi/] vci] [dlci dlci] [input | output]
[class class-name]
```

The end of Example 14-3 shows a sample of the command, which lists statistics for marking. If other MQC-based QoS features were configured, statistics for those features would also be displayed. As you see from the generic command, the **show policy-map interface** command allows you to select just one interface, either input or output, and even select a single class inside a single policy map for display.

The **load-interval** interface subcommand can also be useful when looking at any QoS tool's statistics. The **load-interval** command defines the time interval over which IOS measures packet and bit rates on an interface. With a lower load interval, the statistics change more quickly; with a larger load interval, the statistics change more slowly. The default setting is 5 minutes, and it can be lowered to 30 seconds.

KEY POINT

Example 14-3 also shows a common oversight with QoS configuration. Note that the first class in **policy-map laundry-list** is **class voip-rtp**. Because that class map matches all RTP audio, it matches the Microsoft NetMeeting audio stream as well, so the NetMeeting audio is not matched by class **NetMeet** that follows. If the first two classes (**voip-rtp** and **NetMeet**) called in the policy map had been reversed, then the NetMeeting audio would have been correctly matched in the **NetMeet** class, and all other audio would have been marked as part of the **voip-rtp** class.

CB Marking of CoS and DSCP

Example 14-4 shows how a router might be configured for CB Marking when an attached LAN switch is performing QoS based on CoS. In this case, R3 looks at frames coming in its fa0/0 interface, marking the DSCP values based on the incoming CoS settings. Additionally, R3 looks at the DSCP settings for packets exiting its fa0/0 interface toward the switch, setting the CoS values in the 802.1Q header. The actual values used on R3's fa0/0 interface for classification and marking are as follows:

- Frames entering with CoS 5 will be marked with DSCP EF.
- Frames entering with CoS 1 will be marked with DSCP AF11.
- Frames entering with any other CoS will be marked DSCP 0.
- Packets exiting with DSCP EF will be marked with CoS 5.
- Packets exiting with DSCP AF11 will be marked with CoS 1.
- Packets exiting with any other DSCP will be marked with CoS 0.

Example 14-4 *Marking DSCP Based on Incoming CoS, and Vice Versa*

```
! The class maps each simply match a single CoS or DSCP value.
class-map cos1
 match cos 1
!
class-map cos5
 match cos 5
!
class-map AF11
 match dscp af11
!
class-map EF
 match dscp EF
! This policy map will map incoming CoS to a DSCP value
policy-map map-cos-to-dscp
 class cos1
  set DSCP af11
 class cos5
  set ip DSCP EF
 class class-default
   set ip dscp default
! This policy map will map incoming DSCP to outgoing CoS. Note that the DSCP
! value is not changed.
policy-map map-dscp-to-cos
 class AF11
  set cos 1
 class EF
  set cos 5
 class class-default
   set cos 0
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! The policy maps are applied to an 802.1q subinterface.
interface FastEthernet0/0.1
 encapsulation dot1Q 102
 service-policy input map-cos-to-dscp
 service-policy output map-dscp-to-cos
!
interface FastEthernet0/0.2
 encapsulation dot1Q 2 native
```

The QoS policy requires two policy maps in this example. Policy map **map-cos-to-dscp** matches CoS values for frames entering R3's fa0/0.1 interface, and marks DSCP values, for packets flowing right to left in Figure 14-5. Therefore, the policy map is enabled on input of R3's fa0/0.1 interface. Policy map **map-dscp-to-cos** matches DSCP values for packets exiting R3's fa0/0.1 interface, and marks the corresponding CoS value. Therefore, the policy map was enabled on the output of R3's fa0/0.1 interface. Neither policy map could be applied on the WAN interface,

because only interfaces configured for 802.1Q accept **service-policy** commands that reference policy maps that either classify or mark based on CoS.

Note that you cannot enable a **policy-map** that refers to CoS on interface fa0/0.2 in this example. That subinterface is in the native VLAN, meaning that no 802.1Q header is used.

Network-Based Application Recognition

CB Marking can make use of NBAR's powerful classification capabilities via the **match protocol** subcommand. Example 14-5 shows a configuration for CB Marking and NBAR in which the following requirements are met:

- Any HTTP traffic whose URL contains the string "important" anywhere in the URL is marked with AF21.
- Any HTTP traffic whose URL contains the string "not-so" anywhere in the URL is marked with DSCP default.
- All other traffic is marked with AF11.

Example 14-5 shows the configuration, along with a few NBAR-related **show** commands.

Example 14-5 *CB Marking Based on URLs, Using NBAR for Classification*

```
ip cef
! The "*" in the url string is a wildcard meaning "0 or more characters."
class-map http-impo
     match protocol http url "*important*"
class-map http-not
     match protocol http url "*not-so*"
! The policy map lists the three classes in order, setting the DSCP values.
policy-map http
 class http-impo
  set dscp AF21
!
 class http-not
  set dscp default
!
 class class-default
  set DSCP AF11
! The ip nbar protocol discovery command may or may not be required—see the notes
! following this example.
interface fastethernet 0/0
 ip nbar protocol-discovery
 service-policy input http
! The show ip nbar command only displays statistics if the ip nbar
! protocol-discovery command is applied to an interface. These statistics are
```

continues

Example 14-5 *CB Marking Based on URLs, Using NBAR for Classification (Continued)*

```
! independent of those created by CB Marking. This example shows several of
! the large number of options on the command.
R3# show ip nbar protocol-discovery interface fastethernet 0/0 stats packet-count top-n 5
 FastEthernet0/0
                              Input                     Output
   Protocol                   Packet Count              Packet Count
   ------------------------ ------------------------ ------------------------
   http                       721                       428
   eigrp                      635                       0
   netbios                    199                       0
   icmp                       1                         1
   bgp                        0                         0
   unknown                    46058                     63
   Total                      47614                     492
```

KEY POINT

NOTE Before the 12.2T/12.3 IOS releases, the **ip nbar protocol-discovery** command was required on an interface before using a **service-policy** command that used NBAR matching. With 12.2T/12.3 train releases, this command is no longer required.

The use of the **match protocol** command implies that NBAR will be used to match the packet.

Unlike most other IOS features, you can upgrade NBAR without changing to a later IOS version. Cisco uses a feature called *Packet Description Language Modules (PDLMs)* to define new protocols that NBAR should match. When Cisco decides to add one or more new protocols to the list of protocols that NBAR should recognize, it creates and compiles a PDLM. You can then download the PDLM from Cisco, copy it into Flash memory, and add the **ip nbar pdlm** *pdlm-name* command to the configuration, where *pdlm-name* is the name of the PDLM file in Flash memory. NBAR can then classify based on the protocol information from the new PDLM.

CB Marking Design Choices

The intent of CB Marking is to simplify the work required of other QoS tools by marking packets of the same class with the same QoS marking. For other QoS tools to take advantage of those markings, packets should generally be marked as close to the ingress point of the packet as possible. However, the earliest possible point may not be a trusted device. For instance, in Figure 14-5 (the figure upon which Examples 14-3 and 14-4 are based), Server1 could set its own DSCP and even CoS if its NIC supported trunking. However, trusting the server administrator may or may not be desirable. So, the following rule summarizes how to choose the best location to perform marking:

KEY POINT

Mark as close to the ingress edge of the network as possible, but not so close to the edge that the marking is made by an untrusted device.

Cisco QoS design guide documents make recommendations not only as to where to perform marking, but also as to which CoS, IPP, and DSCP values to set for certain types of traffic. Table 14-9 summarizes those recommendations.

Table 14-9 *RFC-Recommended Values for Marking*

Type of Traffic	CoS	IPP	DSCP
Voice payload	5	5	EF
Video payload	4	4	AF41
Voice/video signaling	3	3	CS3
Mission-critical data	3	3	AF31, AF32, AF33
Transactional data	2	2	AF21, AF22, AF23
Bulk data	1	1	AF11, AF12, AF13
Best effort	0	0	BE
Scavenger (less than best effort)	0	0	2, 4,6

Also note that Cisco recommends not to use more than four or five different service classes for data traffic. By using more classes, the difference in behavior between the various classes tends to blur. For the same reason, do not give too many data service classes high-priority service.

Marking Using Policers

Traffic policers measure the traffic rate for data entering or exiting an interface, with the goal of determining if a configured *traffic contract* has been exceeded. The contract has two components: a *traffic rate*, configured in bits/second, and a *burst size*, configured as a number of bytes. If the traffic is within the contract, all packets are considered to have *conformed* to the contract. However, if the rate or burst exceeds the contract, then some packets are considered to have *exceeded* the contract. QoS actions can be taken on both categories of traffic.

The simplest form of policing enforces the traffic contract strictly by forwarding conforming packets and discarding packets that exceed the contract. However, both IOS policers allow a compromise action in which the policer *marks down* packets instead of dropping them. To mark down the packet, the policer re-marks a QoS field, typically IPP or DSCP, with a value that makes the packet more likely to be discarded downstream. For instance, a policer could re-mark AF11 packets that exceed a contract with a new DSCP value of AF13, but not discard the packet. By doing so, the packet still passes through the router, but if the packet experiences congestion later in its travels, it is more likely to be discarded than it would have otherwise been. (Remember, DiffServ suggests that AF13 is more likely to be discarded than AF11 traffic.)

When marking requirements can be performed by using CB Marking, CB Marking should be used instead of either policer. However, if a requirement exists to mark packets based on whether they conform to a traffic contract, marking with policers must be used. Chapter 16, "Shaping and Policing," covers CB policing, with an example of the syntax it uses for marking packets.

Policy Routing for Marking

Policy routing provides the capability to route a packet based on information in the packet besides the destination IP address. The policy routing configuration uses route maps to classify packets. The **route-map** clauses include **set** commands that define the route (based on setting a next-hop IP address or outgoing interface).

Policy routing can also mark the IPP field, or the entire ToS byte, using the **set** command in a route map. When using policy routing for marking purposes, the following logic sequence is used:

1. Packets are examined as they enter an interface.
2. A route map is used to match subsets of the packets.
3. Mark either the IPP or entire ToS byte using the **set** command.
4. The traditional policy routing function of using the **set** command to define the route may also be configured, but it is not required.

Policy routing should be used to mark packets only in cases where CB Marking is not available, or when a router needs to both use policy routing and mark packets entering the same interface. Refer to Chapter 7, "IP Forwarding (Routing)," for a review of policy routing configuration, and note the syntax of the **set** commands for marking, listed in Table 7-5.

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most of the Cisco Press *Exam Certification Guides*, this book does not repeat information listed in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 14-10 lists the various **match** commands that can be used for MQC tools like CB Marking.

Table 14-10 **match** *Configuration Command Reference for MQC Tools*

Command	Function
match [**ip**] **precedence** *precedence-value* [*precedence-value precedence-value precedence-value*]	Matches precedence in IPv4 packets when the **ip** parameter is included; matches IPv4 and IPv6 packets when **ip** parameter is missing.
match access-group {*access-group* | **name** *access-group-name*}	Matches an ACL by number or name.
match any	Matches all packets.
match class-map *class-map-name*	Matches based on another class map.
match cos *cos-value* [*cos-value cos-value cos-value*]	Matches a CoS value.
match destination-address mac *address*	Matches a destination MAC address.
match fr-dlci *dlci-number*	Matches a particular Frame Relay DLCI.
match input-interface *interface-name*	Matches an ingress interface.
match ip dscp *ip-dscp-value* [*ip-dscp-value ip-dscp- value ip-dscp-value ip-dscp-value ip-dscp-value ip- dscp-value ip-dscp-value*]	Matches DSCP in IPv4 packets when the **ip** parameter is included; matches IPv4 and IPv6 packets when the **ip** parameter is missing.
match ip rtp *starting-port-number port-range*	Matches the RTP's UDP port-number range, even values only.
match mpls experimental *number*	Matches an MPLS Experimental value.
match mpls experimental topmost *value*	When multiple labels are in use, matches the MPLS EXP field in the topmost label.

continues

Table 14-10 **match** *Configuration Command Reference for MQC Tools (Continued)*

Command	Function
match not *match-criteria*	Reverses the matching logic. In other words, things matched by the matching criteria do *not* match the class map.
match packet length {**max** *maximum-length-value* [**min** *minimum-length-value*] | **min** *minimum-length-value* [**max** *maximum-length-value*]}	Matches packets based on the minimum length, maximum length, or both.
match protocol citrix app *application-name-string*	Matches NBAR Citrix applications.
match protocol http [**url** *url-string* | **host** *hostname- string* | **mime** *MIME-type*]	Matches a host name, URL string, or MIME type.
match protocol *protocol-name*	Matches NBAR protocol types.
match protocol rtp [**audio** | **video** | **payload-type** *payload-string*]	Matches RTP audio or video payload, based on the payload type. Also allows explicitly specifying payload types.
match qos-group *qos-group-value*	Matches a QoS group.
match source-address mac *address-destination*	Matches a source MAC address.

Table 14-11 lists the RFCs related to DiffServ.

Table 14-11 *DiffServ RFCs*

RFC	Title	Comments
2474	Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers	Contains the details of the 6-bit DSCP field in IP header
2475	An Architecture for Differentiated Service	The core DiffServ conceptual document
2597	Assured Forwarding PHB Group	Defines a set of 12 DSCP values and a convention for their use
2598	An Expedited Forwarding PHB	Defines a single DSCP value as a convention for use as a low-latency class
3260	New Terminology and Clarifications for DiffServ	Clarifies, but does not supersede, existing DiffServ RFCs

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

IP Precedence, ToS byte, Differentiated Services, DS field, Per-Hop Behavior, Assured Forwarding, Expedited Forwarding, Class Selector, Class of Service, Differentiated Services Code Point, User Priority, Discard Eligible, Cell Loss Priority, MPLS Experimental, class map, policy map, service policy, Modular QoS CLI, Class-Based Marking, Network Based Application Recognition

Refer to the CD-based glossary to check your answers.

Further Reading

Cisco QoS Exam Certification Guide, by Wendell Odom and Michael Cavanaugh

The Enterprise QoS SRND document, posted at http://www.cisco.com/go/srnd, provides great background and details for real-life QoS deployments.

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- (Quality of Service) QoS
  - — Congestion Management
  - — Congestion Avoidance

CHAPTER 15

Congestion Management and Avoidance

Congestion management, commonly called *queuing*, refers to how a router or switch manages packets or frames while they wait to exit a device. With routers, the waiting occurs once IP forwarding has been completed, so the queuing is always considered to be output queuing. LAN switches often support both output queuing and input queuing, where input queuing is used for received frames that are waiting to be switched to the switch's output interfaces.

Congestion avoidance refers to the logic used when deciding if and when packets should be dropped as a queuing system becomes more congested. This chapter covers a wide variety of Cisco IOS queuing tools, along with the most pervasive congestion avoidance tool, namely Weighted Random Early Detection (WRED).

"Do I Know This Already?" Quiz

Table 15-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 15-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Cisco Router Queuing Concepts	1	
Queuing Tools: FIFO, PQ, CQ, WFQ, CBWFQ, and LLQ	2–5	
Weighted Random Early Detection	6–7	
LAN Switch Congestion Management and Avoidance	8–9	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. What is the main benefit of the hardware queue on a Cisco router interface?
   a. Prioritizes latency-sensitive packets so that they are always scheduled next
   b. Reserves a minimum amount of bandwidth for particular classes of traffic
   c. Provides a queue so that as soon as the interface is available to send another packet, the packet can be sent, without requiring an interrupt to the router CPU
   d. Allows configuration of a percentage of the remaining link bandwidth, after allocating bandwidth to the LLQ and the class-default queue

2. Which of the following are benefits of custom queuing being enabled on a Cisco router interface?
   a. Prioritizes latency-sensitive packets so that they are always scheduled next
   b. Reserves a minimum amount of bandwidth for particular classes of traffic
   c. Provides a place to hold packets in RAM until space becomes available in the hardware queue for the interface
   d. Allows configuration of a percentage of the remaining link bandwidth, after allocating bandwidth to the LLQ and the class-default queue

3. Which of the following are reasons why WFQ might discard a packet instead of putting it into the correct queue?
   a. The hold-queue limit for all combined WFQ queues has been exceeded.
   b. The queue length for the flow has passed the WRED minimum drop threshold.
   c. The WFQ queue length for the queue where the newly arrived packet should be placed has exceeded the CDT.
   d. ECN feedback has been signaled, requesting that the TCP sender slow down.

4. Examine the following configuration snippet. If a new class, called class3, was added to the policy map, which of the following commands could be used to reserve 25 kbps of bandwidth for the class?

```
policy-map fred
 class class1
  priority 20
 class class2
  bandwidth 30
!
interface serial 0/0
 bandwidth 100
 service-policy output fred
```

   a. **priority 25**
   b. **bandwidth 25**
   c. **bandwidth percent 25**
   d. **bandwidth remaining-percent 25**

5. Examine the following configuration snippet. How much bandwidth does Cisco IOS assign to class2?

```
policy-map fred
 class class1
  priority percent 20
 class class2
  bandwidth remaining percent 20
interface serial 0/0
 bandwidth 100
 service-policy output fred
```

   a. 10 kbps
   b. 11 kbps
   c. 20 kbps
   d. 21 kbps
   e. Not enough information to tell

6. Which of the following impacts the percentage of packet discards when using WRED, when the current average queue depth is between the minimum and maximum thresholds?

   a. The **bandwidth** command setting on the interface
   b. The mark probability denominator (MPD)
   c. The exponential weighting constant
   d. The congestive discard threshold

7. Which of the following commands, under an interface like s0/0, would enable WRED and tell it to use IP Precedence (IPP) when choosing its default traffic profiles?

   a. **random-detect**
   b. **random-detect precedence-based**
   c. **random-detect dscp-based**
   d. **random-detect precedence 1 20 30 40**

8. On a Catalyst 3550 switch, interface fa0/1 has been configured for WRR scheduling, and fa0/2 has been configured for WRR scheduling with an expedite queue. Which of the following is true regarding interface fa0/2?

   a. It must be configured with the **priority-queue out** command.
   b. The last parameter (*w4*) of the **wrr-queue bandwidth** *w1 w2 w3 w4* command must be 0.
   c. Only CoS 5 frames can be placed into queue 1, the expedite queue.
   d. Only DSCP EF frames can be placed into queue 4, the expedite queue.

9. Which of the following statements are true when comparing WRED operation in a Cisco 3550 switch with WRED operation in a Cisco router?

   a. 3550s do not include a configurable mark probability denominator.
   b. 3550s do allow for the configuration of a minimum and maximum threshold like router WRED.
   c. Both allow for a different WRED profile for each DSCP.
   d. All of these answers are correct.

Foundation Topics

Cisco Router Queuing Concepts

Cisco routers can be configured to perform *fancy queuing* for packets that are waiting to exit an interface. For instance, if a router receives 5 Mbps of traffic every second for the next several seconds, and all that traffic needs to exit a T1 serial link, the router can't forward all the traffic. So, the router places the packets into one or more *software queues*, which can then be managed—thus impacting which packets get to leave next, and which packets might be discarded.

Software Queues and Hardware Queues

Although many network engineers already understand queuing, many overlook some of the details behind the concepts of a hardware queue and a software queue associated with each physical interface. The queues created on an interface by the popularly known queuing tools are called *software queues*, as these queues are implemented in software. However, when the queuing scheduler picks the next packet to take from the software queues, the packet does not move directly out the interface. Instead, the router moves the packet from the interface software queue to a small hardware FIFO (first-in, first-out) queue on each interface. Cisco calls this separate, final queue either the *transmit queue* (TX queue) or *transmit ring* (TX ring), depending on the model of the router; generically, these queues are called *hardware queues*.

Hardware queues provide the following features:

KEY POINT

- When an interface finishes sending a packet, the next packet from the hardware queue can be encoded and sent out the interface, without requiring a software interrupt to the CPU—ensuring full use of interface bandwidth.
- Always use FIFO logic.
- Cannot be affected by IOS queuing tools.
- IOS automatically shrinks the length of the hardware queue to a smaller length than the default when a queuing tool is present.
- Short hardware queue lengths mean packets are more likely to be in the controllable software queues, giving the software queuing more control of the traffic leaving the interface.

The only function of a hardware queue that can be manipulated is the length of the queue. Example 15-1 shows how to see the current length of the queue, and how to change the length.

Example 15-1 *TX Queue Length: Finding and Changing the Length*

```
! The example begins with only FIFO queuing on the interface. For this
! router, it defaults to a TX queue length 16.
R3# show controllers serial 0/0
Interface Serial0/0
! about 30 lines omitted for brevity
tx_limited=0(16)
! lines omitted for brevity
! Next, the TX ring is set to length 1.
! (The smallest recommended value is 2.)
R3# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R3(config)# int s 0/0
R3(config-if)# tx-ring-limit 1
R3(config-if)# ^Z
```

Queuing on Interfaces Versus Subinterfaces and Virtual Circuits

IOS queuing tools can create and manage software queues associated with a physical interface, and then the packets drain into the hardware queue associated with the interface. Additionally, queuing tools can be used in conjunction with traffic shaping. Traffic-shaping tools delay packets to ensure that a class of packets does not exceed a defined traffic rate. While delaying the packets, the shaping function queues the packets—by default in a FIFO queue. Depending on the type of shaping tool in use, various queuing tools can be configured to manage the packets delayed by the shaping tool.

Chapter 16, "Shaping and Policing," covers traffic shaping, including how to use queuing tools with shapers. This chapter's queuing coverage focuses on the implementation of software queuing tools directly on the physical interface.

Comparing Queuing Tools

Cisco IOS provides a wide variety of queuing tools. The upcoming sections of this chapter describe several different IOS queuing tools, with a brief summary ending the section on queuing. Table 15-2 summarizes the main characteristics of different queuing tools that you will want to keep in mind while comparing each successive queuing tool.

Table 15-2 *Key Comparison Points for Queuing Tools*

KEY POINT

Feature	Definition
Classification	The ability to look at packet headers to choose the right queue for each packet
Drop policy	The rules used to choose which packets to drop as queues begin to fill
Scheduling	The logic used to determine which packet should be dequeued next

Table 15-2 *Key Comparison Points for Queuing Tools (Continued)*

Feature	Definition
Maximum number of queues	Defines the number of unique classes of packets for a queuing tool
Maximum queue length	The maximum number of packets in a single queue

Queuing Tools: FIFO, PQ, CQ, WFQ, CBWFQ, and LLQ

This section hits the highlights of a few of the queuing tools, and covers detailed configuration for the more popular tools—specifically weighted fair queuing (WFQ), class-based WFQ (CBWFQ), and low-latency queuing (LLQ). The queuing tools are covered in the order in which they were added as IOS features.

FIFO Queuing

The primary reason for queuing is that a router needs to hold a packet in memory while the outgoing interface is busy sending another packet. FIFO queuing simply provides a software queue to hold packets while they are waiting to exit an interface. Packets are scheduled to leave the interface based on when they arrived at the output interface.

Because IOS defaults to use WFQ on serial interfaces with bandwidths of E1 speeds (2.048 Mbps) or less, to configure FIFO queuing, you simply need to disable WFQ (or any other queuing tool) from the interface. The only other useful setting for FIFO queuing is to set the length of the FIFO queue using the **hold-queue *x* out** interface subcommand, where *x* is the maximum queue length. Example 15-2 shows how to configure and view FIFO queuing.

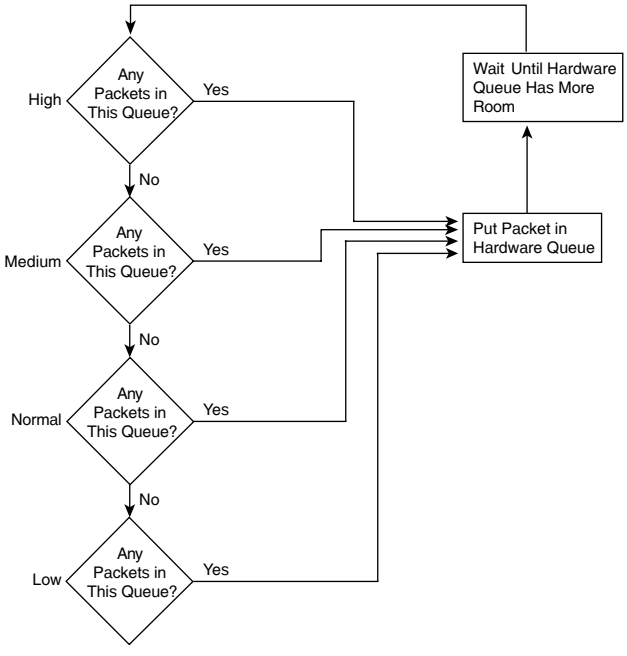
Example 15-2 *FIFO Queuing Configuration*

```
! Below, FIFO is enabled by disabling the default queuing tool (WFQ).
R3# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R3(config)# int s 0/0
R3(config-if)# no fair-queue
R3(config-if)# ^Z
! Next, the queuing type is listed as FIFO, with the current queue length of 0,
! and the maximum queue length of (default) 40.
R3# sh int s 0/0
Serial0/0 is up, line protocol is up
! lines omitted for brevity
Queueing strategy: fifo
Output queue: 0/40 (size/max)
```

Priority Queuing

Priority queuing's most distinctive feature is its scheduler. PQ schedules traffic such that the higher-priority queues always get serviced instead of lower-priority queues. PQ uses up to four queues, named high, medium, normal, and low, and they are scheduled as shown in Figure 15-1.

Figure 15-1 *PQ Scheduling Logic*



The PQ scheduler has some obvious benefits and drawbacks. Packets in the high queue get wonderful service—they can claim 100 percent of the link bandwidth, with minimal delay and minimal jitter. (Generally speaking, the idea of a queue that provides very low delay and low jitter is called a *low-latency queue* [LLQ] or *priority queue*.) However, when the interface becomes congested, the lower queues experience *queue starvation*, sometimes getting little or no bandwidth, with queuing delay times varying wildly as congestion occurs and abates.

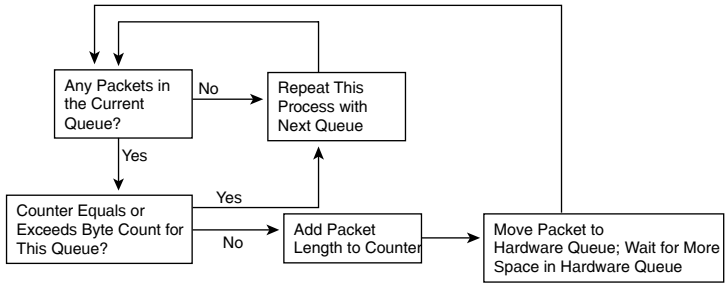
Although the scheduler is the most interesting part of PQ, it has several other features as well. PQ classifies packets into the four queues via configurable classification details, including references to IP ACLs. However, PQ supports fewer classification options than do the more recently added CBWFQ and LLQ. PQ uses *tail-drop logic*, so when a new packet arrives for a particular queue, and the queue is full, the new packet is dropped. The queue lengths can be configured to different values, with the default queue lengths being 20, 40, 60, and 80 for the high, medium, normal, and low queues, respectively.

Custom Queuing

CQ addresses the biggest drawback of PQ by providing a guaranteed minimum bandwidth to each queue, thereby avoiding queue starvation. CQ has 16 queues, implying 16 classification categories, which is plenty for most applications. (There is also one hidden system queue for important overhead traffic; the system queue cannot be configured or disabled.) The negative part of CQ, as compared to PQ, is the lack of a high-priority queue that is always serviced first. That is, CQ has no way to provide guaranteed low-latency service to any traffic.

The CQ scheduler reserves an approximate percentage of overall link bandwidth for each queue, but instead of configuring actual percentages, CQ approximates the bandwidth percentages using a simple algorithm. Figure 15-2 depicts the CQ scheduler logic.

Figure 15-2 *CQ Scheduling Logic for Current Queue*



The CQ scheduler performs round-robin service on each queue, beginning with queue 1. CQ takes packets from the queue, until the total byte count specified for the queue has been met or exceeded. After the queue has been serviced for the defined byte count, or when the queue does not have any more packets, CQ moves on to the next queue and repeats the process.

None of the queues has an implied priority—they simply have an implied percentage bandwidth. In fact, just because a queue might be configured with a byte count that implies 60 percent of the link's bandwidth, if 90 percent of the traffic ends up in that queue, you could argue that the queue is a low-priority queue. Table 15-3 shows two examples of the percentages of link bandwidth assigned to each queue. Note that if some of the possible 16 queues are not configured, as is the case with each example, the unconfigured queues are essentially ignored.

Table 15-3 *Two Examples of CQ Byte Counts and Percentage Bandwidth Reserved*

CQ Queue	Example 1 Byte Count	Example 1 Link Percent	Example 2 Byte Count	Example 2 Link Percent
1	5,000	25%	10,000	20%
2	5,000	25%	5,000	10%

continues

Table 15-3 *Two Examples of CQ Byte Counts and Percentage Bandwidth Reserved (Continued)*

CQ Queue	Example 1 Byte Count	Example 1 Link Percent	Example 2 Byte Count	Example 2 Link Percent
3	5,000	25%	10,000	20%
4	5,000	25%	15,000	30%
5	Not used	Not used	10,000	20%

The link percentage calculated in Table 15-3 used this simple formula:

$$\frac{\text{Byte count for queue } x}{\text{sum of byte counts for all queues}}$$

Like PQ, the scheduler is the most interesting part of CQ. As for the other features of CQ, it is much like PQ: CQ uses the same classification options, and can use tail drop only for managing drops. Because no one queue is better than another, CQ defaults to a queue length of 20 packets.

Weighted Fair Queuing

Weighted fair queuing differs from PQ and CQ in several significant ways. The most outwardly obvious difference is that WFQ does not allow classification options to be configured. WFQ automatically classifies packets based on flows, with each flow being placed into a separate queue. For WFQ purposes, a *flow* is defined as all packets with the same values for the following:

KEY POINT

- Source IP address
- Destination IP address
- Transport layer protocol (TCP or UDP)
- TCP or UDP source port
- TCP or UDP destination port
- IP Precedence

Because WFQ puts packets of different flows in different queues, it necessarily has a much larger number of queues than any of the non-flow-based queuing tools. The WFQ scheduler uses logic that is quite different from the logic of other queuing tools so that it can deal with the larger number of queues. However, the overall goals of the WFQ scheduler are straightforward, and are summarized as follows:

KEY POINT

- Flows with the same IPP should be given the same amount of bandwidth, regardless of how many bytes are sent in each flow.

- For flows with different IPP values, give flows with higher IPP a proportionally higher amount of bandwidth.
- The result: WFQ favors lower-volume, higher IPP flows.

For instance, if WFQ is currently managing ten flows with equal IPP values, on a 128-kbps interface, each flow effectively gets 12.8 kbps. Queues start to lengthen for flows that offer more than 12.8 kbps of traffic, meaning longer queuing delays. Queues shorten for flows that offer less than 12.8 kbps—in fact, the queues may empty, and WFQ will then allocate additional bandwidth to the other queues. The result is that the lower-volume flows prosper, and the higher-volume flows suffer.

The second goal of the WFQ scheduler is to provide more bandwidth to flows with higher IPP values. To do so, the flows are weighted based on *IPP plus 1*. In other words, precedence 7 flows get eight times more bandwidth than do precedence 0 flows, because (7 + 1) / (0 + 1) = 8. For another example, if you compare precedence 3 to precedence 0, the ratio is (3 + 1) / (0 + 1) = 4.

WFQ Scheduler: The Process

To achieve WFQ's goals for allocating link bandwidth, WFQ uses a scheduler that is actually pretty simple. The WFQ scheduler takes the packet with the lowest *sequence number (SN)* (also sometimes called *finish time*, or *FT*) when it needs to move the next packet to the hardware queue. WFQ assigns each packet an SN when the packet is added to a WFQ flow queue. The SN assignment process is actually the more interesting part of the scheduler.

The WFQ scheduler includes both the packet length and IPP when calculating the SN. The formula for calculating the SN for a packet is as follows:

Previous_SN + (*weight* * new_packet_length)

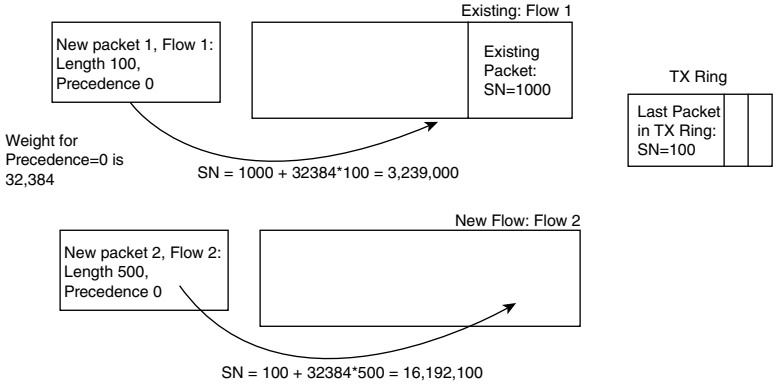
Where *weight* is calculated as follows:

Weight = 32,384 / (IP_Precedence + 1)

The formula considers the length of the new packet, the weight of the flow, and the previous SN. By considering the packet length, the SN calculation results in a higher number for larger packets, and a lower number for smaller packets. By including the SN of the previous packet enqueued into that queue, the formula assigns a larger number for packets in queues that already have a larger number of packets enqueued. And by putting the weight (IPP + 1) in the denominator, packets with higher IPP values end up with lower SNs. Figure 15-3 shows how two packets are assigned their sequence numbers.

Calculating the SN for the packet of an existing flow in Figure 15-3 is relatively straightforward. However, the first packet in a new flow does not have a previous SN to use in the formula. The figure shows that the SN of the last packet moved to the hardware queue is used as the new flow's previous SN.

Figure 15-3 *WFQ Sequence Number Assignment Example*



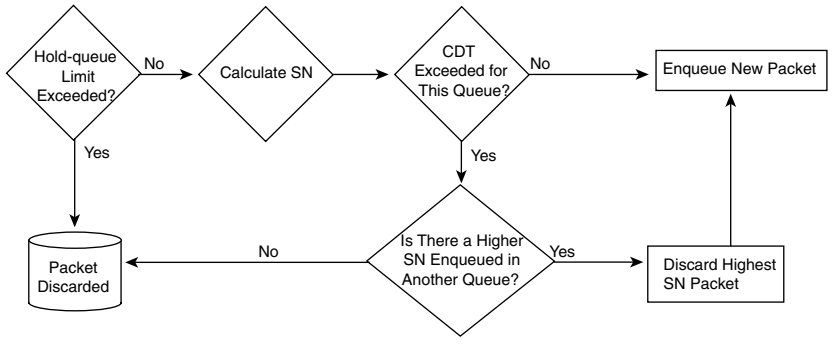
Once the sequence numbers are assigned, the scheduler's job in choosing which packet to take next is simple: it takes the packet with the lowest SN among all the queues.

WFQ Drop Policy, Number of Queues, and Queue Lengths

WFQ uses a two-step process called *modified tail drop* to choose when to drop packets. First, WFQ considers the absolute limit on the number of packets enqueued among all queues; this limit is called the *hold-queue limit*. If a new packet arrives, and the hold-queue limit has been reached, the packet is discarded. That part of the decision is based not on a single queue, but on the whole WFQ queuing system for the interface.

Second, WFQ considers the length of the queue into which the newly arrived packet will be placed. Before adding a new packet to its queue, the *congestive discard threshold (CDT)* is checked against the actual length of that queue. If that queue is longer than CDT packets long, one packet is discarded—but maybe not the newly arrived packet. Figure 15-4 depicts the WFQ drop decision process.

Figure 15-4 *WFQ Modified Tail Drop and Congestive Discard Threshold*



Note that each newly arriving packet goes through this process, but in one branch of the logic, the newly arrived packet is not discarded—another one is discarded. Essentially, if the new packet needs to go into a queue that has exceeded CDT, WFQ discards the highest-SN packet from any of the queues.

NOTE CDT must be set to a power of 2; IOS will not accept any other values.

Types of WFQ Queues

WFQ can be configured for a maximum of 4096 queues, called *dynamic queues* in WFQ **show** command output. Interestingly, the allowed configurable values are powers of 2, between 16 and 4096, inclusive. IOS restricts the values because WFQ performs a hash algorithm to classify traffic, and the hash algorithm only works when the number of queues is one of these valid values. Additionally, WFQ keeps eight hidden queues for overhead traffic generated by the router. WFQ uses a very low weight for these queues to give preference to the overhead traffic.

Also, some of the up to 4096 queues can be used by the Resource Reservation Protocol (RSVP) to reserve bandwidth. RSVP uses signaling to reserve a minimum bandwidth for particular flows in a network, relying on queuing tools to actually implement the bandwidth reservations. WFQ, CBWFQ, and LLQ support RSVP.

WFQ Configuration

WFQ requires very little configuration—in fact, it is on by default on E1-and-slower serial links. So, with default settings, WFQ literally requires no configuration commands. Table 15-4 lists the commands related to WFQ. Of particular importance, note that the **fair-queue** command allows setting of the CDT, number of queues, and number of RSVP queues, and the **hold-queue** command sets the number of packets allowed in all queues on the interface.

Table 15-4 *Command Reference for WFQ*

Command	Mode and Function
fair-queue [*congestive-discard-threshold* [*dynamic-queues* [*reservable-queues*]]]	Interface mode; enables WFQ and changes defaults
hold-queue *length* **out**	Interface mode; changes the length of the hold queue
show queue *interface-name interface-number* [**vc** [*vpi*/] *vci*]]	Lists information about the packets waiting in a queue
show queueing [**custom** | **fair** | **priority** | **random-detect** [**interface** *atm-subinterface* [**vc** [[*vpi*/] *vci*]]]]	Lists configuration and statistical information about the queuing tool

Example 15-3 shows a basic configuration that defines non-default settings, along with **show** commands. To configure WFQ as simply as possible, the command **fair-queue** would be used, with no parameters, and the **hold-queue** command would not be needed at all.

Example 15-3 *WFQ Configuration and* **show** *Commands*

```
! Below, CDT is set to 100, number of queues to 64, number of RSVP queues to 10,
! and the overall limit (hold queue) of 500.
R3# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R3(config)# int s 0/0
R3(config-if)# fair-queue 100 64 10
R3(config-if)# hold-queue 500 out
R3(config-if)# ^Z
! The output below does not match typical WFQ terminology in some cases. The
! "max total" is the hold queue size, with "threshold" being CDT, and "size"
! being the actual number of packets enqueued at that time.
R3# show interface serial 0/0
Serial0/0 is up, line protocol is up
!lines omitted for brevity
  Queueing strategy: weighted fair
  Output queue: 95/500/100/12474 (size/max total/threshold/drops)
     Conversations  5/6/64 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 1158 kilobits/sec
! lines omitted for brevity
! Below, details for a VoIP flow queue are shown, with statistics for queue
! depth and weight. The weight is listed as 5397, which was calculated as
! (32,384 / ((IPP of 5) + 1)).
R3# sh queue s 0/0
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 13567
  Queueing strategy: weighted fair
  Output queue: 125/500/100/13567 (size/max total/threshold/drops)
     Conversations  5/7/64 (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) 61/5397/654/0/0
  Conversation 61, linktype: ip, length: 64
  source: 192.168.3.254, destination: 192.168.2.251, id: 0x0134, ttl: 253,
  TOS: 184 prot: 17, source port 16638, destination port 19476
! lines omitted for brevity
```

The most interesting **show** command for WFQ is the **show queue** command. Note that a summary section is listed first, followed by a stanza of output for each active flow queue. The details of the flow are listed, as well as statistics for the flow queue.

Class-Based WFQ and Low-Latency Queuing

Cisco created CBWFQ and LLQ using some of the best concepts from PQ, CQ, and WFQ, while adding several additional features. CBWFQ reserves bandwidth for each queue, and provides the ability to use WFQ concepts for packets in the default (class-default) queue. LLQ adds to CBWFQ the concept of a priority queue, but unlike PQ, LLQ prevents the high-priority queue from starving other queues. Additionally, both CBWFQ and LLQ use MQC for configuration, which means that they have robust classification options, including NBAR.

CBWFQ and LLQ use almost identical configuration; the one major difference is whether the **bandwidth** command (CBWFQ) or the **priority** command (LLQ) is used to configure the tool. Because both tools use MQC, both use class maps for classification and policy maps to create a set of classes to be used on an interface. The classes defined in the policy map each define a single queue; as a result, the terms *queue* and *class* are often used interchangeably when working with LLQ and CBWFQ.

CBWFQ and LLQ support 64 queues/classes. The maximum queue length can be changed, with the maximum possible value and the default length varying based on the model of router and the amount of memory installed. They both also have one special queue called the *class-default queue*. This queue exists even if it is not configured. If a packet does not match any of the explicitly configured classes in a policy map, IOS places the packet into the class-default class/queue. CBWFQ settings can be configured for the class-default queue.

The sections that follow cover the details of CBWFQ, and then LLQ.

CBWFQ Basic Features and Configuration

The CBWFQ scheduler guarantees a minimum percentage of a link's bandwidth to each class/queue. If all queues have a large number packets, each queue gets the percentage bandwidth implied by the configuration. However, if some queues are empty and do not need their bandwidth for a short period, the bandwidth is proportionally allocated across the other classes. (Cisco does not publish the details of how CBWFQ achieves these functions.)

Table 15-5 summarizes some of the key features of CBWFQ.

Table 15-5 *CBWFQ Functions and Features*

KEY POINT

CBWFQ Feature	Description
Classification	Classifies based on anything that MQC commands can match
Drop policy	Tail drop or WRED, configurable per queue
Number of queues	64

continues

Table 15-5 *CBWFQ Functions and Features (Continued)*

KEY POINT

CBWFQ Feature	Description
Maximum queue length	Varies based on router model and memory
Scheduling inside a single queue	FIFO on 63 queues; FIFO or WFQ on class-default queue*
Scheduling among all queues	Result of the scheduler provides a percentage of guaranteed bandwidth to each queue

*Cisco 7500 series routers support FIFO or WFQ in all the CBWFQ queues.

Table 15-6 lists the key CBWFQ commands that were not covered in Chapter 14.

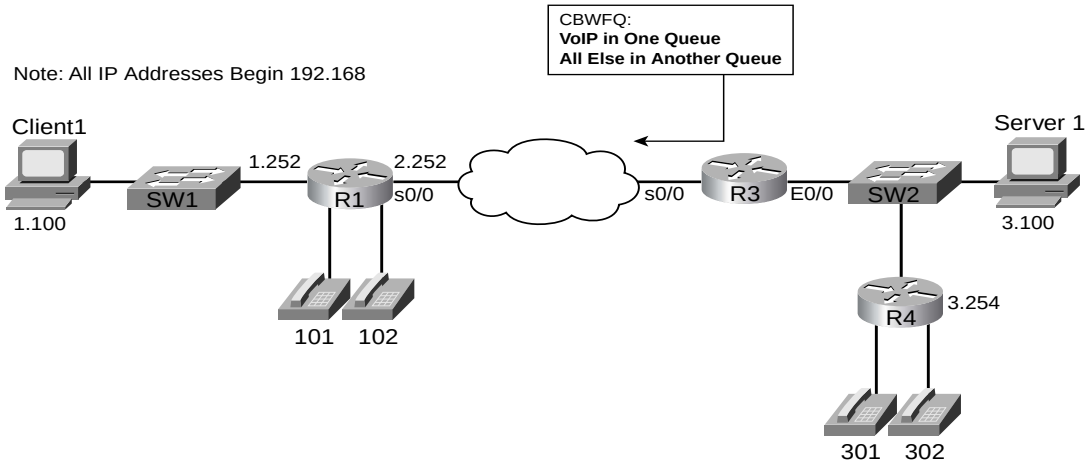
Table 15-6 *Command Reference for CBWFQ*

Command	Mode and Function
bandwidth {*bandwidth-kbps* | **percent** *percent*}	Class subcommand; sets literal or percentage bandwidth for the class
bandwidth {**remaining percent** *percent*}	Class subcommand; sets percentage of remaining bandwidth for the class
queue-limit *queue-limit*	Class subcommand; sets the maximum length of a CBWFQ queue
fair-queue [**queue-limit** *queue-value*]	Class subcommand; enables WFQ in the class (class-default only)
max-reserved-bandwidth *percent*	Interface subcommand; defines the percentage of link bandwidth that can be reserved for CBWFQ queues besides class-default (default: 75 percent)

Example 15-4 shows a simple CBWFQ configuration that uses the class-default queue. The configuration was created on R3 in Figure 15-5, using the following requirements:

- All VoIP payload traffic is placed in a queue.
- All other traffic is placed in another queue.
- Give the VoIP traffic 50 percent of the bandwidth.
- WFQ should be used on the non-VoIP traffic.

Figure 15-5 *Network Used with CBWFQ and LLQ Configuration Examples*



Example 15-4 *CBWFQ with VoIP in One Queue, Everything Else in Class-Default*

```
! The class map matches on UDP/RTP header and RTP port numbers.
class-map match-all voip-rtp
  match ip rtp 16384 16383
! Next, the policy map uses the bandwidth command to reserve 64 kbps for the class
! voip-rtp. Class-default gets some of the leftover bandwidth by default.
policy-map queue-voip
  class voip-rtp
   bandwidth 64
  class class-default
   fair-queue
! The interface's bandwidth 128 command is used as the basis for the limit on the
! amount of bandwidth that can be allocated in the policy map queue-voip.
! The load-interval command sets how often counters are updated. Also, note
! that the policy-map is enabled for output; input is not allowed on routers for
! policy maps that perform queuing.
interface Serial0/0
 encapsulation frame-relay
 load-interval 30
 bandwidth 128
 service-policy output queue-voip
! This command lists counters, reserved bandwidth, maximum queue length (listed
! as max threshold), and a reminder that WFQ is used in the class-default queue.
R3# show policy-map int s 0/0
 Serial0/0
  Service-policy output:     queue-voip
```

continues

Example 15-4 *CBWFQ with VoIP in One Queue, Everything Else in Class-Default (Continued)*

```
    Class-map: voip-rtp (match-all)
      136435 packets, 8731840 bytes
      30 second offered rate 51000 bps, drop rate 0 bps
      Match:     ip rtp 16384 16383
      Weighted Fair Queueing
        Output Queue: Conversation 265
        Bandwidth 64 (kbps) Max Threshold 64 (packets)
        (pkts matched/bytes matched) 48550/3107200
        (depth/total drops/no-buffer drops) 14/0/0

    Class-map: class-default (match-any)
      1958 packets, 1122560 bytes
      30 second offered rate 59000 bps, drop rate 0 bps
      Match: any
      Weighted Fair Queueing
        Flow Based Fair Queueing
        Maximum Number of Hashed Queues 256
        (total queued/total drops/no-buffer drops) 15/0/0
! This command just lists the configuration in a concise manner.
R3# show policy-map
  Policy Map queue-voip
    Class voip-rtp
      Weighted Fair Queueing
                Bandwidth 64 (kbps) Max Threshold 64 (packets)
    Class class-default
      Weighted Fair Queueing
Flow based Fair Queueing Max Threshold 64 (packets)
```

Defining and Limiting CBWFQ Bandwidth

Cisco IOS checks a CBWFQ policy map to ensure that it does not allocate too much bandwidth. IOS performs the check when the **service-policy output** command is added; if the policy map defines too much bandwidth for that interface, the **service-policy** command is rejected. IOS defines the allowed bandwidth based on two interface subcommands: the **bandwidth** command, and the reserved bandwidth implied by the **max-reserved-bandwidth** command (abbreviated hereafter as **int-bw** and **max-res**, respectively). The non-reservable bandwidth is meant for overhead traffic, much like CQ's system queue.

IOS allows a policy map to allocate bandwidth based on the product of **int-bw** and **max-res**. In other words, with a default **max-res** setting of 75 (75 percent), on an interface with **int-bw** of 256 (256 kbps), the policy map could allocate at most 192 kbps of bandwidth with its various **bandwidth** commands. Example 15-5 shows a simple example with a policy map that contains one class that has 64 kbps configured. The **service-policy** command is rejected on an interface whose bandwidth is set to 64 kbps.

Example 15-4 *CBWFQ Rejected Due to Request for Too Much Bandwidth*

```
! max-res was defaulted to 75, so only 75% of 64 kbps, or 48 kbps,
! is available. Note that the 48 kbps is mentioned in the error message.
R3(config-cmap)# policy-map explicit-bw
R3(config-pmap)# class class1
R3(config-pmap-c)# bandwidth 64
R3(config-pmap-c)# int s 0/1
R3(config-if)# bandwidth 64
R3(config-if)# service-policy output explicit-bw
I/f Serial0/1 class class1 requested bandwidth 64 (kbps), available only 48 (kbps)
```

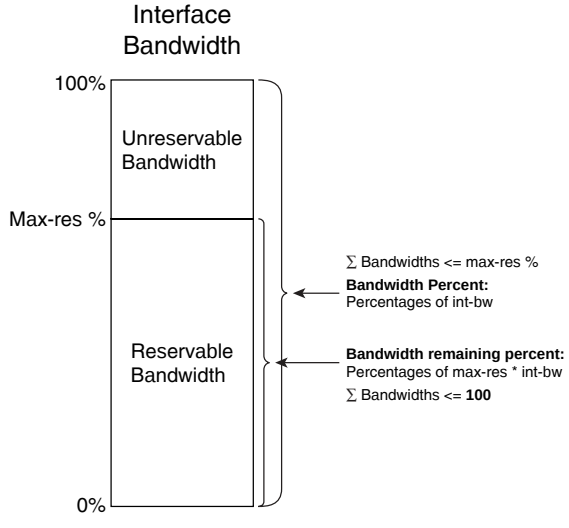
To overcome such problems, the engineer can simply pay attention to details and ensure that the policy map's configured **bandwidth** commands do not total more than **max-res** * **int-bw**. Alternatively, **max-res** can be defined to a higher number, up to a value of 100; however, Cisco does not recommend changing **max-res**.

The bandwidths can also be defined as percentages using either the **bandwidth percent** or **bandwidth remaining percent** command. By using percentages, it is easier to ensure that a policy map does not attempt to allocate too much bandwidth.

The two percentage-based **bandwidth** command options work in slightly different ways. Figure 15-6 shows the concept for each.

Figure 15-6 *Bandwidth Percent and Bandwidth Remaining Percent Concepts*

KEY POINT



The **bandwidth percent** *bw-percent* command sets a class's reserved bandwidth as a percentage of **int-bw**. For instance, in Example 15-4, if the **bandwidth percent 50** command had been used

instead of **bandwidth 64**, the voip-rtp class would have used 50% * 128 kbps, or 64 kbps. IOS checks all the **bandwidth percent** commands in a single policy map to ensure that the total does not exceed the **max-res** setting for the interface—in other words, with a default setting for **max-res**, all the **bandwidth percent** commands in a single policy map cannot total more than 75.

The **bandwidth remaining percent** *bw-percent* command sets a class's reserved bandwidth as a percentage of remaining bandwidth. *Remaining bandwidth* is the reservable bandwidth, calculated as **int-bw** * **max-res**. This method allows a policy map to allocate percentages that total 100 (100 percent). Using Example 15-4 again, the remaining bandwidth would be 75% * 128 kbps, or 96 kbps, and the command **bandwidth remaining percent 50** would allocate 48 kbps for a class.

NOTE Using the **bandwidth remaining percent** command is particularly useful with LLQ, and will be explained in that context later in the chapter. The reason is that the remaining-bandwidth calculation is changed by the addition of LLQ.

Note that in a single policy map, only one of the three variations of the **bandwidth** command may be used. Table 15-7 summarizes the three methods for reserving bandwidth with CBWFQ.

Table 15-7 *Reference for CBWFQ Bandwidth Reservation*

KEY POINT

Method	Amount of Bandwidth Reserved by the bandwidth Command	The Sum of Values in a Single Policy Map Must Be <= ...
Explicit bandwidth	As listed in commands	**max-res** * **int-bw**
Percent	A percentage of the **int-bw**	**max-res** setting
Remaining percent	A percentage of the reservable bandwidth (**int-bw** * **max-res**)	100

Low-Latency Queuing

Low-latency queuing sounds like the best queuing tool possible, just based on the name. What packet wouldn't want to experience low latency? As it turns out, for delay (latency) sensitive traffic, LLQ is indeed the queuing tool of choice. LLQ looks and acts just like CBWFQ in most regards, except it adds the capability for some queues to be configured as low-latency queues. LLQ schedules these specific queues as strict-priority queues, just like PQ schedules the high-priority queue. In other words, LLQ always services packets in these priority queues first.

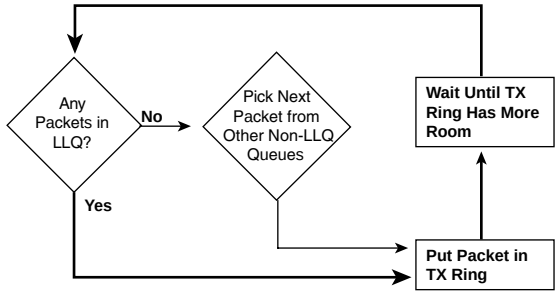
LLQ lingo can sometimes be used in a couple of different ways. With a single policy map that has at least one low-latency queue, the policy map might be considered to be implementing LLQ, while at the

same time, that one low-latency queue is often called "the LLQ." Sometimes, a single low-latency queue is even called "the PQ" as a reference to the PQ-like behavior, or even a "priority queue."

While LLQ adds a low-latency queue to CBWFQ, it also prevents the queue starvation that occurs with PQ. LLQ actually *polices* the PQ based on the configured bandwidth. In effect, the bandwidth given to an LLQ priority queue is both the guaranteed minimum and policed maximum. (You may recall from Chapter 14, "Classification and Marking," that the DiffServ Expedited Forwarding PHB formally defines the priority queuing and policing PHBs.) As a result, the packets that make it out of the queue experience low latency, but some may be discarded to prevent starving the other queues.

Figure 15-7 depicts the scheduler logic for LLQ. Note that the PQ logic is shown, but with the policer check as well.

Figure 15-7 *LLQ Scheduler Logic*



LLQ configuration requires one more command in addition to the commands used for CBWFQ configuration. Instead of using the **bandwidth** command on a class, use the **priority** command:

```
priority {bandwidth-kbps | percent percentage} [burst]
```

This class subcommand enables LLQ in the class, reserves bandwidth, and enables the policing function. You can also configure the burst size for the policer with this command, but the default setting of 20 percent of the configured bandwidth is typically a reasonable choice.

Example 15-5 shows a sample LLQ configuration, using the following criteria. Like Example 15-4, the LLQ policy is applied to R3's s0/0 interface from Figure 15-5.

- R3's s0/0 bandwidth is 128 kbps.
- Packets will already have been marked with good DSCP values.
- VoIP payload is already marked DSCP EF, and should be LLQed with 58 kbps of bandwidth.
- AF41, AF21, and AF23 traffic should get 22, 20, and 8 kbps, respectively.
- All other traffic should be placed into class class-default, which should use WRED and WFQ.

Example 15-5 *LLQ for EF, CBWFQ for AF41, AF21, AF23, and All Else*

```
! The class maps used by the queue-on-dscp are not shown, but the names imply what
! each class map has been configured to match. Note the priority 58 command makes
! class dscp-ef an LLQ.
policy-map queue-on-dscp
  class dscp-ef
    priority 58
  class dscp-af41
   bandwidth 22
  class dscp-af21
   bandwidth 20
   random-detect dscp-based
  class dscp-af23
   bandwidth 8
   random-detect dscp-based
  class class-default
   fair-queue
   random-detect dscp-based
! max-res has to be raised or the policy map would be rejected.
interface Serial0/0
 bandwidth 128
 encapsulation frame-relay
 load-interval 30
 max-reserved-bandwidth 85
 service-policy output queue-on-dscp
! Below, for class dscp-ef, note the phrase "strict priority," as well as the
! computed policing burst of 1450 bytes (20% of 58 kbps and divided by 8 to convert
! the value to a number of bytes.)
R3# show policy-map queue-on-dscp
    Policy Map queue-on-dscp
    Class dscp-ef
      Weighted Fair Queueing
            Strict Priority
            Bandwidth 58 (kbps) Burst 1450 (Bytes)
! lines omitted for brevity
! Note the statistics below. Any packets dropped due to the policer would show
! up in the last line below.
R3# show policy-map interface s 0/0 output class dscp-ef
 Serial0/0
  Service-policy output: queue-on-dscp
    Class-map: dscp-ef (match-all)
      227428 packets, 14555392 bytes
      30 second offered rate 52000 bps, drop rate 0 bps
      Match: ip dscp ef
      Weighted Fair Queueing
        Strict Priority
        Output Queue: Conversation 40
        Bandwidth 58 (kbps) Burst 1450 (Bytes)
        (pkts matched/bytes matched) 12194/780416
 (total drops/bytes drops) 0/0
```

Defining and Limiting LLQ Bandwidth

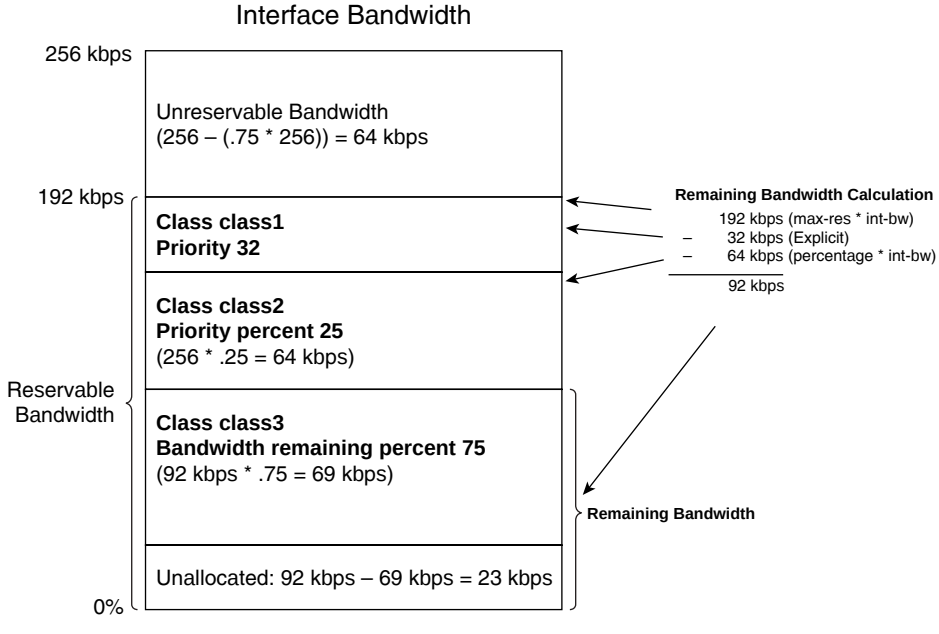
The LLQ **priority** command provides two syntax options for defining the bandwidth of an LLQ—a simple explicit amount, or bandwidth as a percentage of interface bandwidth. (There is no *remaining bandwidth* equivalent for the **priority** command.) However, unlike the **bandwidth** command, both the explicit and percentage versions of the **priority** command can be used inside the same policy map.

IOS still limits the amount of bandwidth in an LLQ policy map, with the actual bandwidth from both LLQ classes (with **priority** commands) and non-LLQ classes (with **bandwidth** commands) not being allowed to exceed **max-res** * **int-bw**. Although the math is easy, the details can get confusing, especially because a single policy map could have one queue configured with priority *bw*, another with **priority percent** *bw*, and others with one of the three versions of the **bandwidth** command. Figure 15-8 shows an example with three versions of the commands.

The figure shows both versions of the **priority** command. Class3 has an explicit **priority 32** command, which reserves 32 kbps. Class2 has a **priority percent 25** command, which, when applied to the interface bandwidth (256 kbps), gives class2 64 kbps.

Figure 15-8 *Priority, Priority Percent, and Bandwidth Remaining Percent*

KEY POINT



The most interesting part of Figure 15-8 is how IOS views the remaining-bandwidth concept when priority queues are configured. IOS subtracts the bandwidth reserved by the **priority** commands as well. As a result, a policy map can essentially allocate non-priority classes based on percentages of the leftover (remaining) bandwidth, with those values totaling 100 (100 percent).

LLQ with More Than One Priority Queue

LLQ allows multiple queues/classes to be configured as priority queues. This begs the question, "Which queue gets scheduled first?" As it turns out, LLQ actually places the packets from multiple LLQs into a single internal LLQ. So, packets in the different configured priority queues still get scheduled ahead of non-priority queues, but they are serviced based on their arrival time for all packets in any of the priority queues.

KEY POINT So why use multiple priority queues? The answer is *policing*. By policing traffic in one class at one speed, and traffic in another class at another speed, you get more granularity for the policing function of LLQ. For instance, if planning for video and voice, you can place each into a separate LLQ and get low latency performance for both types of traffic, but at the same time prevent video traffic from consuming the bandwidth engineered for voice, and vice versa.

Miscellaneous CBWFQ/LLQ Topics

CBWFQ and LLQ allow a policy map to either allocate bandwidth to the class-default class, or not. When a **bandwidth** command is configured under **class class-default**, the class is indeed reserved that minimum bandwidth. (IOS will not allow the **priority** command in **class-default**.) When class **class-default** does not have a **bandwidth** command, IOS internally allocates any unassigned bandwidth among all classes. As a result, **class class-default** may not get much bandwidth unless the class is configured a minimum amount of bandwidth using the **bandwidth** command.

This chapter's coverage of guaranteed bandwidth allocation is based on the configuration commands. In practice, a policy map may not have packets in all queues at the same time. In that case, the queues get more than their reserved bandwidth. IOS allocates the extra bandwidth proportionally to each active class's bandwidth reservation.

Finally, IOS uses queuing only when congestion occurs. IOS considers congestion to be occurring when the hardware queue is full; that generally happens when the offered load of traffic is far less than the clock rate of the link. So, a router could have a **service-policy out** command on an interface, with LLQ configured, but the LLQ logic would only be used when the hardware queue is full.

Queuing Summary

Table 15-8 summarizes some of the key points regarding the IOS queuing tools covered in this chapter.

Table 15-8 *Queuing Protocol Comparison*

KEY POINT

Feature	PQ	CQ	WFQ	CBWFQ	LLQ
Includes a strict-priority queue	Yes				Yes
Polices priority queues to prevent starvation					Yes

Table 15-8 *Queuing Protocol Comparison (Continued)*

Feature	PQ	CQ	WFQ	CBWFQ	LLQ
Reserves bandwidth per queue		Yes		Yes	Yes
Includes robust set of classification fields				Yes	Yes
Classifies based on flows			Yes	Yes[1]	Yes[1]
Supports RSVP			Yes	Yes	Yes
Maximum number of queues	4	16[2]	4096	64	64

[1]WFQ can be used in the class-default queue, or in all CBWFQ queues in 7500 series routers.
[2]Also includes a system queue that is unavailable for customer use.

Weighted Random Early Detection

When a queue is full, IOS has no place to put newly arriving packets, so it discards them. This phenomenon is called *tail drop*. Oftentimes, when a queue fills, several packets are tail dropped at a time, given the bursty nature of data packets.

Tail drop can have an overall negative effect on network traffic, particularly TCP traffic. When packets are lost, for whatever reason, TCP senders slow down their rate of sending data. When tail drops occur and multiple packets are lost, the TCP connections slow down even more. Also, most networks send a much higher percentage of TCP traffic than UDP, meaning that the overall network load tends to drop after multiple packets are tail dropped.

Interestingly, overall throughput can be improved by discarding a few packets as a queue begins to fill, rather than waiting for the larger impact of tail drops. Cisco created *Weighted Random Early Detection (WRED)* specifically for the purpose of monitoring queue length and discarding a percentage of the packets in the queue to improve overall network performance. As a queue gets longer and longer, WRED begins to discard more packets, hoping that a small reduction in offered load that follows may be just enough to prevent the queue from filling.

WRED uses several numeric settings when making its decisions. First, WRED uses the measured *average queue depth* when deciding if a queue has filled enough to begin discarding packets. WRED then compares the average depth to a minimum and maximum queue threshold, performing different discard actions depending on the outcome. Table 15-9 lists the actions.

When the average queue depth is very low or very high, the actions are somewhat obvious, although the term full drop in Table 15-9 may be a bit of a surprise. When the average depth rises above the maximum threshold, WRED discards all new packets. Although this action might seem like tail drop, technically it is not, because the actual queue might not be full. So, to make this fine distinction, WRED calls this action category *full drop*.

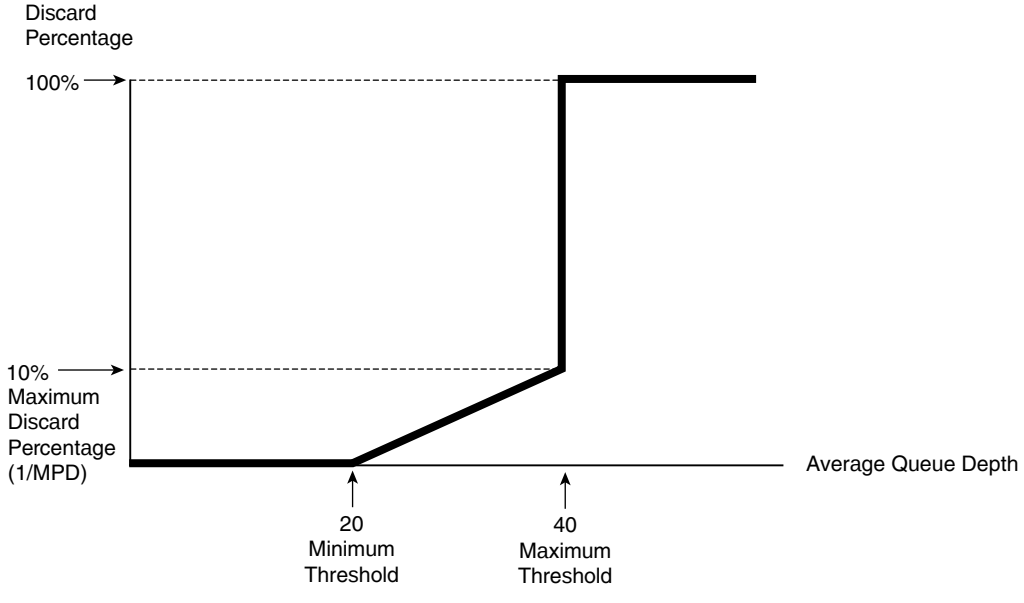
Table 15-9 *WRED Discard Categories*

KEY POINT

Average Queue Depth Versus Thresholds	Action	WRED Name for Action
Average < minimum threshold	No packets dropped.	No drop
Minimum threshold < average depth < maximum threshold	A percentage of packets dropped. Drop percentage increases from 0 to a maximum percent as the average depth moves from the minimum threshold to the maximum.	Random drop
Average depth > maximum threshold	All new packets discarded; similar to tail drop.	Full drop

When the average queue depth is between the two thresholds, WRED discards a percentage of packets. The percentage grows linearly as the average queue depth grows from the minimum threshold to the maximum, as depicted in Figure 15-9 (which shows WRED's default settings for IPP 0 traffic).

Figure 15-9 *WRED Discard Logic with Defaults for IPP 0*



The last of the WRED numeric settings that affect its logic is the *mark probability denominator (MPD)*, from which the maximum percentage of 10 percent is derived in Figure 15-9. IOS calculates the discard percentage used at the maximum threshold based on the simple formula 1/MPD. In the figure, an MPD of 10 yields a calculated value of $\frac{1}{10}$, meaning the discard rate grows from 0 percent to 10 percent as the average queue depth grows from the minimum threshold to the maximum. Also, when WRED discards packets, it randomly chooses the packets to discard.

How WRED Weights Packets

WRED gives preference to packets with certain IPP or DSCP values. To do so, WRED uses different traffic profiles for packets with different IPP and DSCP values. A WRED *traffic profile* consists of a setting for three key WRED variables: the minimum threshold, the maximum threshold, and the MPD. Figure 15-10 shows just such a case, with two WRED traffic profiles (for IPP 0 and IPP 3).

As Figure 15-10 illustrates, IPP 3's minimum threshold was higher than for IPP 0. As a result, IPP 0 traffic will be discarded earlier than IPP 3 packets. Also, the MPD is higher for IPP 3, resulting in a lower discard percentage (based on the formula discard percentage = 1/MPD).

Figure 15-10 *Example WRED Profiles for Precedences 0 and 3*

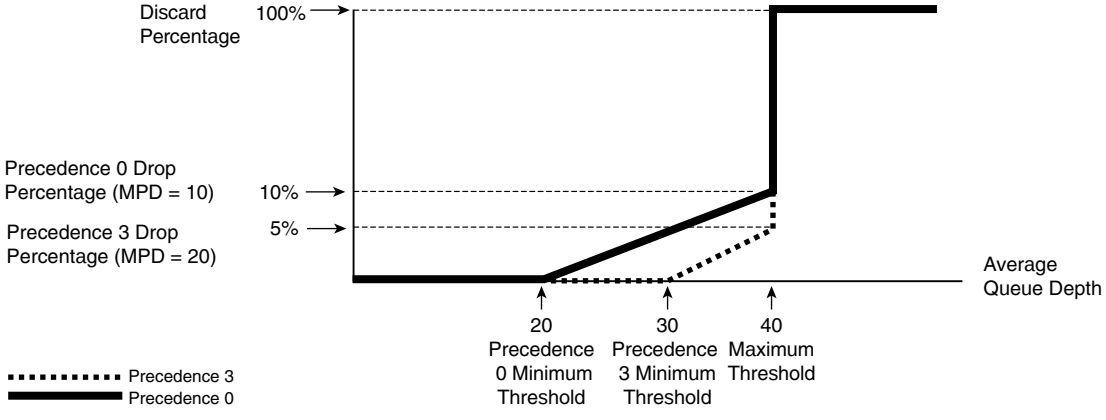


Table 15-10 lists the IOS default WRED profile settings for various DSCP values. You may recall from Chapter 14 that Assured Forwarding DSCPs whose names end in 1 (for example, AF21) should get better WRED treatment than those settings that end in 2 (for example, AF32). The IOS defaults listed in Table 15-10 achieve that goal by setting lower minimum thresholds for the appropriate AF DSCPs.

Table 15-10 *Cisco IOS Software Default WRED Profiles for DSCP-Based WRED*

DSCP	Minimum Threshold	Maximum Threshold	MPD	1/MPD
AFx1	33	40	10	10%
AFx2	28	40	10	10%
AFx3	24	40	10	10%
EF	37	40	10	10%

WRED Configuration

Because WRED manages drops based on queue depth, WRED must be configured alongside a particular queue. However, most queuing mechanisms do not support WRED; as a result, WRED can be configured only in the following locations:

- On a physical interface (with FIFO queuing)
- For a non-LLQ class inside a CBWFQ policy map
- For an ATM VC

To use WRED directly on a physical interface, IOS actually disables all other queuing mechanisms and creates a single FIFO queue. WRED then manages the queue with regard to drops. For CBWFQ, WRED is configured in a class inside a policy map, in the same location as the **bandwidth** and **priority** commands discussed earlier in this chapter.

The **random-detect** command enables WRED, either under a physical interface or under a **class** in a policy map. This command enables WRED to use IPP, and not DSCP. The **random-detect dscp-based** command both enables WRED and tells it to use DSCP for determining the traffic profile for a packet.

To change WRED configuration from the default WRED profile for a particular IPP or DSCP, use the following commands, in the same location as the other **random-detect** command:

```
random-detect precedence precedence min-threshold max-threshold [mark-prob-
  denominator]
random-detect dscp dscpvalue min-threshold max-threshold [mark-probability-
  denominator]
```

Finally, calculation of the rolling average queue depth can be affected through configuring a parameter called the *exponential weighting constant*. A low exponential weighting constant means that the old average is a small part of the calculation, resulting in a more quickly changing average. The setting can be changed with the following command, although changing it is not recommended:

```
random-detect exponential-weighting-constant exponent
```

Note that earlier, Example 15-6 showed basic WRED configuration inside some classes of a CBWFQ configuration.

LAN Switch Congestion Management and Avoidance

The final section of this chapter looks at queuing and WRED on Cisco 3550 switches, with some comparisons made with Cisco 2950 switches.

Cisco 3550 Switch Egress Queuing

Cisco 3550 switches perform both ingress and egress queuing. However, Cisco 3550 switches use a single FIFO ingress queue as a place to hold frames waiting to be forwarded to the egress interface, so the details are not terribly interesting. For egress, the Cisco 3550 supports four queues per interface, with classification into the queues based on CoS, and scheduling based on weighted round-robin (WRR) logic and an optional expedited (priority) queue.

The Cisco 3550 uses a relatively simple classification scheme, assuming you consider only what happens once the forwarding decision has been made. Cisco 3550 switches make most internal QoS decisions based on an *internal DSCP* setting. The internal DSCP has been determined once the frame is forwarded. So, once a frame has been assigned an internal DSCP and an egress interface, the following logic determines into which of the four interface output queues the frame is placed:

KEY POINT

1. The frame's internal DSCP is compared to a global DSCP-to-CoS map to determine a CoS value.
2. The per-interface CoS-to-queue map determines the queue for a frame based on the assigned CoS.

WRR scheduling works much like the router CQ scheduler, but instead of taking a number of bytes per cycle, WRR specifies a number of frames taken from each queue. For example, the **wrr-queue bandwidth 10 20 30 40** interface subcommand indicates that the switch would send 10 frames from the first queue, then 20 from the next, and so on. The switch is not really concerned about allocating bandwidth; in fact, because frames vary in length, switch WRR logic does not even indirectly define a reserved minimum bandwidth percentage. However, it does empty the queue slots as defined, so that there is then more space to temporarily store subsequent frames.

The Cisco 3550 may treat queue 4 on an interface as a PQ. To do so, the interface subcommand **priority-queue out** is configured. On Cisco 3550 switches, only queue 4 can become the PQ. The switch still applies WRR scheduling to queues 1 through 3.

Example 15-6 shows Cisco 3550 egress queuing configuration. By default, the global DSCP-to-CoS map maps the first eight DSCP values to CoS 0, the next eight to CoS 1, and so on. Also, the default per-interface CoS-to-queue mapping maps the first two CoS values to queue 1, the next two to queue 2, and so on. For this example, the following criteria are used:

- The global DSCP-to-CoS map is changed so that DSCPs 60–63 are mapped to CoS 1.
- Interface gi0/1's CoS-to-queue map is changed so that CoS 5 is mapped to queue 4, and CoS 6 and 7 are mapped to queue 3.
- The gi0/1 bandwidth ratios are set to 10, 15, 25, and 150.

- Later, the expedite queue is enabled, giving queues 1, 2, and 3 around 20 percent, 30 percent, and 50 percent, respectively, of the bandwidth not consumed by priority queue 4.

Example 15-6 *Cisco 3550 Egress Queuing Example*

```
! For the global DSCP-to-CoS map, up to eight DSCPs can be mapped in a single command,
! as seen below. The show mls qos map dscp-cos command shows a grid, with the
! DSCP's decimal first digit on the left-side column, and the 2nd digit across the
! top.
S1(config)# mls qos map dscp-cos 60 61 62 63 to 1
S1# sh mls qos map dscp-cos
   Dscp-cos map:
     d1 :  d2 0  1  2  3  4  5  6  7  8  9
     ---------------------------------------
      0 :    00 00 00 00 00 00 00 00 01 01
      1 :    01 01 01 01 01 01 02 02 02 02
      2 :    02 02 02 02 03 03 03 03 03 03
      3 :    03 03 04 04 04 04 04 04 04 04
      4 :    05 05 05 05 05 05 05 05 06 06
      5 :    06 06 06 06 06 06 07 07 07 07
      6 :    01 01 01 01
! Next, queue 4 is assigned CoS 5, queue 3 is assigned CoSs 6 and 7. Note that
! the wrr-queue cos-map 3 6 7 command does not remove other CoS values from queue
! 3, but just assigns CoSs 6 and 7 to queue 3. The non-highlighted CoS values in
! the show mls qos int gi 0/1 queue command reflect default settings.
S1(config)# int gi 0/1
S1(config-if)# wrr-queue cos-map 4 5
S1(config-if)# wrr-queue cos-map 3 6 7
S1(config-if)# do show mls qos int gi 0/1 queue | begin Cos-queue
Cos-queue map:
cos-qid
 0—1
 1—1
 2—2
 3—2
 4—3
 5—4
 6—3
 7—3
! Next, the ratios used by the WRR scheduler are defined. Note that the show
! command lists the exact weights, and that it lists "dis" beside the phrase
! "Egress Expedite Queue," meaning that the PQ is not yet enabled. Note also that
! the four weights must be values between 1 and 65,536, inclusive.
S1(config)# int gi 0/1
S1(config-if)# wrr-queue bandwidth 10 15 25 150
S1(config-if)# do sh mls qos int gi 0/1 queue
GigabitEthernet0/1
Egress expedite queue: dis
wrr bandwidth weights:
```

Example 15-6 *Cisco 3550 Egress Queuing Example (Continued)*

```
qid-weights
 1—10
 2—15
 3—25
 4—150
! Finally, the PQ is enabled.
S1(config-if)# priority-queue out
S1(config-if)# do sh mls qos int gi 0/1 queue
GigabitEthernet0/1
Egress expedite queue: ena
wrr bandwidth weights:
qid-weights
 1—10
 2—15
 3—25
 4—150    when expedite queue is disabled
! Lines omitted for brevity
```

Cisco 3550 Congestion Avoidance

Catalyst 3550 Gigabit interfaces support a mutually exclusive choice of either WRED or tail-drop logic for managing drops in egress queues. 3550 Fast Ethernet interfaces do not use WRED or tail drop, but rather use a switch-specific method of managing internal buffers (which is not covered in this book).

Cisco 3550 WRED has the same overall strategy as WRED as implemented in Cisco routers, but with many differences in implementation details. The key features of Cisco 3550 WRED are as follows, with Figure 15-11 depicting the main concepts:

KEY POINT

- Each egress queue has two WRED thresholds.
- Thresholds are defined as percentages of the queue length.
- The thresholds can be set differently for each egress queue on each interface.
- When the actual queue depth is below the threshold, WRED does not discard packets (no drop).
- When the actual queue depth exceeds a threshold, WRED discards a percentage of packets; the percentage ranges linearly from 0 to 100 percent, as the queue depth grows from the threshold to 100 percent full.

Figure 15-11 *Cisco 3550 WRED Logic and Commands*

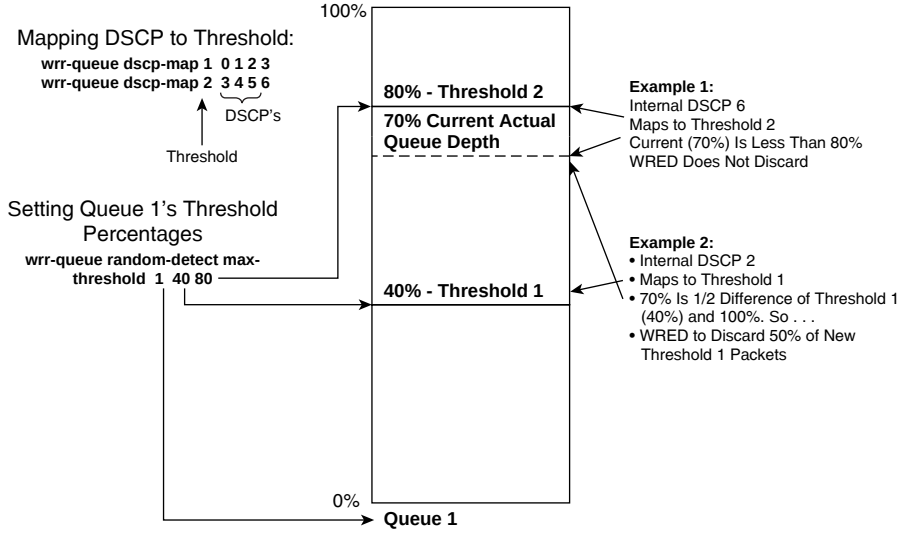


Figure 15-11 shows WRED configuration and logic once a particular egress queue has been chosen. In the figure, a frame has been assigned to egress queue 1. Before enqueuing the frame, WRED finds the frame's internal DSCP in the DSCP-to-threshold map, as configured with the **wrr-queue dscp-map** interface subcommand. That mapping identifies the WRED threshold to use (either threshold 1 or 2). The default DSCP map maps all 64 DSCP values to WRED threshold 1, so to use threshold 2, the map must be configured.

The existence of a single **wrr-queue random-detect** command under an interface both enables WRED on the interface for all queues and disables the default tail-drop logic for each queue. If only one interface egress queue has a **wrr-queue random-detect** command, the other three egress queues do use WRED. However, WRED defaults to use thresholds of 100 percent, meaning that queues that have no specifically configured thresholds behave as if tail drop were used. To enable WRED-like behavior, each queue needs to have non-default thresholds configured.

The Cisco 3550 uses tail drop by default, but with some interesting optional features. You can configure tail drop at levels less than a 100-percent-full queue, with two thresholds as configured with the **wrr-queue threshold** interface subcommand. Along with the DSCP map (configured with the **wrr-queue dscp-map** command), frames with one set of DSCP values will all be discarded once the queue reaches the defined depth. Frames of the other DSCPs are discarded at a second threshold. In effect, this configuration provides for differentiated tail drop, discarding all packets at one lower threshold.

Comparisons Between Cisco 3550 and 2950 Switches

Cisco includes the 2950 series switches in most examples used for the CCIP QoS exam, but the CCIE Routing and Switching lab includes the 3550 switch. Cisco is not specific about any particular switch models to expect on the CCIE Routing and Switching written exam. As a result, it is useful to compare the QoS features of the two switches. Table 15-11 summarizes the key differences, with some additional explanations following the table. (The comparisons listed here assume the Enhanced software image is used on both models of switches.)

Table 15-11 *Comparison of Cisco 2950 and 3550 Queuing Options*

KEY POINT

Feature Description	2950	3550
Number of egress queues	4	4
Queue number of expedite queue	4	4
Granularity for setting queue weights	Global	Interface
Frames classified into queues based on…	CoS	CoS
Granularity for CoS-to-queue mapping	Global	Interface
Expedite queue enabled via **priority-queue out** interface subcommand	No	Yes
Expedite queue enabled via weight of queue 4 being set to "0"	Yes	No
Supports WRED, with two thresholds/queue	No	Yes
Default scheduler	Strict priority	WRR

The most important key conceptual differences relate to scheduling. Cisco 2950 switches default to use *strict-priority scheduling*, treating queue 4 as the highest priority and queue 1 as the lowest. Additionally, both support WRR scheduling (the default on 3550s), and both allow queue 4 to be used as a PQ.

The biggest configuration differences relate to the Cisco 2950 global configuration settings. Because the 2950 queuing details are configured with global commands, each interface uses the same queuing configuration. Queuing details can be configured per interface on 3550s. Also, the Cisco 2950 enables the PQ portion of WRR with PQ differently as compared to the Cisco 3550. Example 15-7 shows the Cisco 2950 queuing configuration that is similar to the Cisco 3550 queuing configuration seen in Example 15-6. Note that the **priority-queue out** command returns an error message on the Cisco 2950.

Example 15-7 *Close Equivalent Cisco 2950 Queuing Configuration as Compared with Example 15-6*

```
! The CoS-to-queue mapping is done with identical commands as with the 3550,
! but the commands are configured globally.
S4(config)# wrr-queue cos-map 4 5
S4(config)# wrr-queue cos-map 3 6 7
! The next command does two important functions: sets the queue weights for each
! interface's set of 4 queues, and enables WRR scheduling on every interface—
! thereby disabling the default strict-priority scheduling.
S4(config)# wrr-queue bandwidth 10 15 25 150
! Alternatively, the weight value 0 for queue 4 makes every interface's queue 4
! act as the expedite queue.
S4(config)# wrr-queue bandwidth 10 15 25 0
S4(config)# int fa 0/1
S4(config-if)# priority-queue out
                     ^
% Invalid input detected at '^' marker.
```

Foundation Summary

Please review the items in the "Foundation Topics" section noted with a Key Point icon.

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

priority queuing, custom queuing, weighted fair queuing, class-based weighted fair queuing, low-latency queuing, weighted round-robin, tail drop, full drop, no drop, priority queue, sequence number, finish time, modified tail drop, scheduler, queue starvation, strict priority, software queue, hardware queue, remaining bandwidth, maximum reserved bandwidth, actual queue depth, average queue depth, minimum threshold, maximum threshold, mark probability denominator, exponential weighting constant, expedite queue, DSCP-to-CoS map, DSCP-to-threshold map, internal DSCP, differentiated tail drop

Refer to the CD-based glossary to check your answers.

Further Reading

Cisco QoS Exam Certification Guide, by Wendell Odom and Michael Cavanaugh

Cisco Catalyst QoS: Quality of Service in Campus Networks, by Mike Flanagan, Richard Froom, and Kevin Turek

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- Quality of Service (QoS)

CHAPTER 16

Shaping and Policing

Traffic-shaping tools delay packets exiting a router so that the overall bit rate does not exceed a defined shaping rate. This chapter covers the concepts behind traffic shaping, as well as two Cisco IOS shapers, namely Frame Relay Traffic Shaping (FRTS) and Class-Based Shaping (CB Shaping).

Traffic policers measure bit rates for packets either entering or exiting an interface. If the defined rate is exceeded, the policer either discards enough packets so that the rate is not exceeded, or marks some packets such that the packets are more likely to be discarded later. This chapter covers the concepts and configuration behind Class-Based Policing (CB Policing), with a brief mention of committed access rate (CAR).

"Do I Know This Already?" Quiz

Table 16-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 16-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Traffic-Shaping Concepts	1–2	
Class-Based Shaping Configuration	3–5	
Frame Relay Traffic Shaping Configuration	6–7	
Policing Concepts and Configuration	8–10	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. When does Class-Based Shaping add tokens to its token bucket, and how many tokens does it add when Bc and Be are both set to something larger than 0?
   a. Upon the arrival of each packet, a pro-rated portion of Bc is added to the token bucket.
   b. Upon the arrival of each packet, a pro-rated portion of Bc + Be is added to the token bucket.
   c. At the beginning of each time interval, Bc worth of tokens are added to the token bucket.
   d. At the beginning of each time interval, Bc + Be worth of tokens are added to the token bucket.
   e. None of the answers is correct.

2. If shaping was configured with a rate of 128 kbps and a Bc of 3200 bits, what value would be calculated for Tc?
   a. 125 ms
   b. 125 sec
   c. 25 ms
   d. 25 sec
   e. Shaping doesn't use a Tc.
   f. Not enough information is provided to tell.

3. Which of the following commands, when typed in the correct configuration mode, enables CB Shaping at 128 kbps, with no excess burst?
   a. **shape average 128000 8000 0**
   b. **shape average 128 8000 0**
   c. **shape average 128000**
   d. **shape peak 128000 8000 0**
   e. **shape peak 128 8000 0**
   f. **shape peak 128000**

4. Examine the following configuration, noting the locations of the comment lines labeled point 1, point 2, and so on. Assume that a correctly configured policy map that implements CBWFQ, called **queue-it**, is also configured but not shown. To enable CBWFQ for the packets queued by CB Shaping, what command is required, and at what point in the configuration is the command required?

```
policy-map shape-question
! point 1
 class class-default
```

```
! point 2
  shape average 256000 5120
! point 3
interface serial 0/0
! point 4
  service-policy output shape-question
! point 5
interface s0/0.1 point-to-point
! point 6
  ip address 1.1.1.1
! point 7
  frame-relay interface-dlci 101
! point 8
```

a. **service-policy queue-it**, at point 1

b. **service-policy queue-it**, at point 3

c. **service-policy queue-it**, at point 5

d. **shape queue service-policy queue-it**, at point 1

e. **shape queue service-policy queue-it**, at point 3

f. **shape queue service-policy queue-it**, at point 6

5. Using the same configuration snippet as in the previous question, what command would list the calculated Tc value, and what would that value be?

   a. **show policy-map**, Tc = 125 ms

   b. **show policy-map**, Tc = 20 ms

   c. **show policy-map**, Tc = 10 ms

   d. **show policy-map interface s0/0**, Tc = 125 ms

   e. **show policy-map interface s0/0**, Tc = 20 ms

   f. **show policy-map interface s0/0**, Tc = 10 ms

6. Assume that several **map-class frame-relay** commands exist in addition to the following configuration. The map classes are named C1, C2, and C3. Which VC use the settings in map class C2?

```
interface s0/0
 encapsulation frame-relay
 frame-relay traffic-shaping
 frame-relay class C2
!
interface s0/0.1 point-to-point
 frame-relay class C1
 frame-relay interface-dlci 101
interface s0/0.3 multipoint
  frame-relay interface-dlci 103
  frame-relay interface-dlci 203
    class C3
```

a. The VC with DLCI 101.
b. The VC with DLCI 103.
c. The VC with DLCI 203.
d. None of the answers is correct.

7. Which of the following FRTS commands, in the same FRTS map class, set the shaping rate to 128 kbps, with a shaping time interval of 62.5 ms?

a. **frame-relay traffic-rate 128**
b. **frame-relay traffic-rate 128000**
c. **frame-relay cir 128**, **frame-relay Bc 8000**
d. **frame-relay cir 128000**, **frame-relay Bc 8000**

8. Which of the following are true about policers in general, but not true about shapers?

a. Monitor traffic rates using the concept of a token bucket
b. Can discard traffic that exceeds a defined traffic rate
c. Can delay packets by queuing to avoid exceeding a traffic rate
d. Can re-mark a packet

9. Which of the following commands, when typed in the correct configuration mode, enables CB Policing at 128 kbps, with no excess burst?

a. **police 128000 conform-action transmit exceed-action transmit violate-action drop**
b. **police 128 conform-action transmit exceed-action transmit violate-action drop**
c. **police 128000 conform-action transmit exceed-action drop**
d. **police 128 conform-action transmit exceed-action drop**
e. **police 128k conform-action transmit exceed-action drop**

10. Which of the following features of CB Policing are not supported by CAR?

a. The capability to categorize packets as conforming, exceeding, and violating a traffic contract
b. The capability to police all traffic at one rate, and subsets of that same traffic at other rates
c. The capability to configure policing using MQC commands
d. The capability to police input or output packets on an interface

Foundation Topics

Traffic-Shaping Concepts

Traffic shaping prevents the bit rate of the packets exiting an interface from exceeding a configured shaping rate. To do so, the shaper monitors the bit rate at which data is being sent. If the configured rate is exceeded, the shaper delays packets, holding the packets in a *shaping queue*. The shaper then releases packets from the queue such that, over time, the overall bit rate does not exceed the shaping rate.

Traffic shaping solves two general types of problems that can occur in multi-access networks. First, if a service provider purposefully discards any traffic on a VC when the traffic rate exceeds the committed information rate (CIR), then it makes sense for the router to not send traffic faster than the CIR.

Egress blocking is the second type of problem for which shaping provides some relief. *Egress blocking* occurs when a router sends data into a Frame Relay or ATM service, and the egress Frame Relay or ATM switch has to queue the data before it can be sent out to the router on the other end of the VC. For example, when a T1-connected router sends data, it must be sent at T1 speed. If the router on the other end of the VC has a link clocked at 256 kbps, the frames/cells will start to back up in the output queue of the egress switch. Likewise, if that same T1 site has VCs to 20 remote sites, and each remote site uses a 256-kbps link, then when all 20 remote sites send at about the same time, frames/cells will be queued, waiting to exit the WAN egress switch to the T1 router. In this case, shaping can be used to essentially prevent egress queuing, moving the packets back into a queue in the router, where they can then be manipulated with fancy queuing tools.

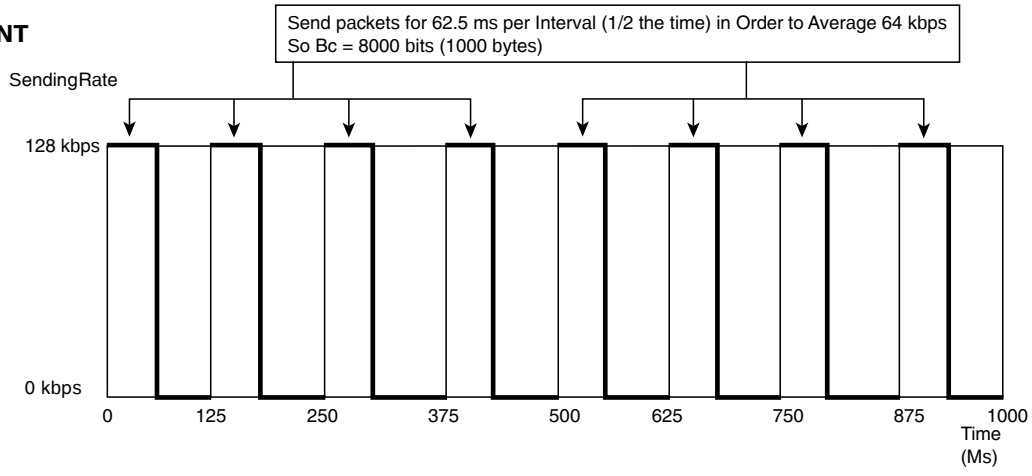
Shaping Terminology

Routers can send bits out an interface only at the physical clock rate. To average sending at a lower rate, the router has to alternate between sending packets and being silent. For instance, to average sending at a packet rate of half the physical link speed, the router should send packets half of the time, and not send packets the other half of the time. Over time, it looks like a staccato series of sending and silence. Figure 16-1 shows a graph of what happens when a router has a link with a clock rate of 128 kbps and a shaper configured to shape traffic to 64 kbps.

Figure 16-1 shows the sending rate and implies quite a bit about how Cisco IOS implements shaping. A shaper sets a static time interval, called *Tc*. Then, it calculates the number of bits that can be sent in the Tc interval such that, over time, the number of bits/second sent matches the shaping rate.

Figure 16-1 *Mechanics of Traffic Shaping—128-kbps Access Rate, 64-kbps Shaped Rate*

KEY POINT



The number of bits that can be sent in each Tc is called the *committed burst (Bc)*. In Figure 16-1, an 8000-bit Bc can be sent in every 125-ms Tc to achieve a 64-kbps average rate. In other words, with a Tc of 125 ms, there will be eight Tc intervals per second. If Bc bits (8000) are sent each Tc, then eight sets of 8000 bits will be sent each second, resulting in a rate of 64,000 bps.

Because the bits must be encoded on the link at the clock rate, the 8000 bits in each interval require only 62.5 ms (8000/128,000) to exit the interface onto the link. The graph shows the results: the interface sends at the line rate (access rate) for 62.5 ms, and then waits for 62.5 ms, while packets sit in the shaping queue.

Table 16-2 lists the terminology related to this shaping model. Note in particular that the term *CIR* refers to the traffic rate for a VC based on a business contract, and *shaping rate* refers to the rate configured for a shaper on a router.

Table 16-2 *Shaping Terminology*

KEY POINT

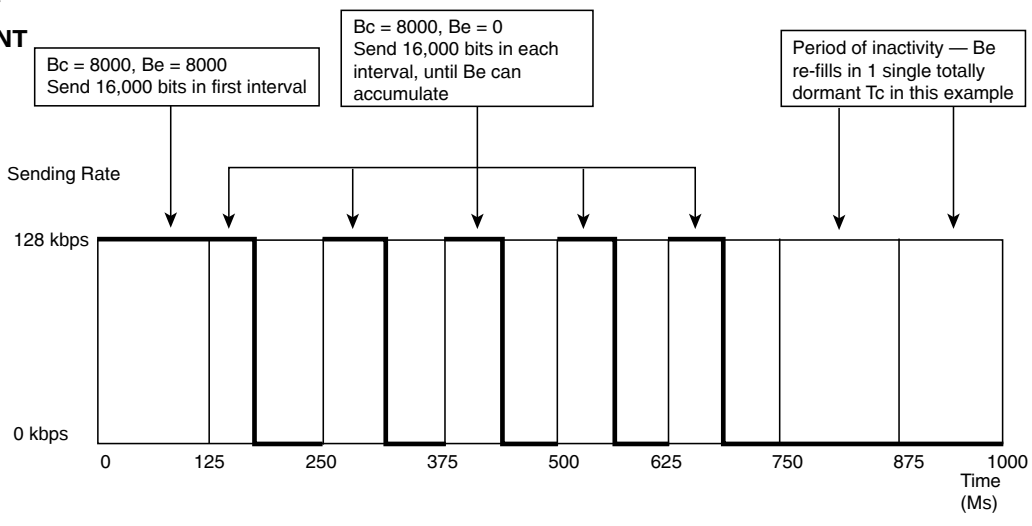
Term	Definition
Tc	Time interval, measured in milliseconds, over which the committed burst (Bc) can be sent. With many shaping tools, Tc = Bc/CIR.
Bc	Committed burst size, measured in bits. This is the amount of traffic that can be sent during the Tc interval. Typically defined in the traffic contract.
CIR	Committed information rate, in bits per second, which defines the rate of a VC according to the business contract.
Shaped rate	The rate, in bits per second, to which a particular configuration wants to shape the traffic. It may or may not be set to the CIR.
Be	Excess burst size, in bits. This is the number of bits beyond Bc that can be sent after a period of inactivity.

Shaping with an Excess Burst

To accommodate bursty data traffic, shapers implement a concept by which, after a period in which an interface sends relatively little data compared to its CIR, more than Bc bits can be sent in one or more time intervals. This concept is called *excess burst (Be)*. When using a Be, the shaper can allow, in addition to the Bc bits per Tc, Be extra bits to be sent. Depending on the settings, it may take one time interval to send the extra bits, or it may require multiple time intervals. Figure 16-2 shows a graph of the same example in Figure 16-1, but with a Be also equal to 8000 bits. In this case, the Be extra bits are all sent in the first time interval after the relative inactivity.

Figure 16-2 *Bc and Be, After a Period of Inactivity*

KEY POINT



In the first interval, traffic shaping can send a total of 16,000 bits (Bc + Be bits). On a 128-kbps link, assuming a 125-ms Tc, all 125 ms is required to send 16,000 bits. In this particular case, after a period of inactivity, R1 sends continuously for the entire first interval. In the second interval, the shaper allows the usual Bc bits to be sent. In effect, with these settings, the shaper allows 192.5 ms of consecutive sending after a period of low activity.

Underlying Mechanics of Shaping

Shapers apply a simple formula to the Tc, Bc, and shaping rate parameters:

KEY POINT

Tc = Bc/shaping rate

For example, in Figures 16-1 and 16-2, if the shaping rate (64 kbps) and the Bc (8000 bits) were both configured, the shaper would then calculate the Tc as 8000/64,000 = 0.125 seconds. Alternatively, if the rate and Tc had been configured, the shaper would have calculated Bc as Bc = rate * Tc (a simple derivation of the formula listed earlier), or 64 kbps * 0.125 ms = 8000 bits. (Both CB Shaping and FRTS use default values in some cases, as described in the configuration sections of this chapter.)

Traffic shaping uses a *token bucket* model to manage the shaping process. First, consider the case in which the shaper is not using Be. Imagine a bucket of size Bc, with the bucket filled with tokens at the beginning of each Tc. Each token lets the shaper buy the right to send 1 bit. So, at the beginning of each Tc, the shaper has the ability to release Bc worth of bits.

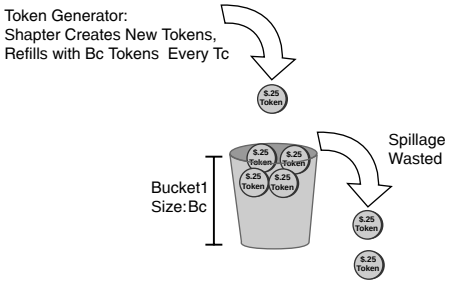
Shapers perform two main actions related to the bucket:

1. Refill the bucket with new tokens at the beginning of each Tc.
2. Spend tokens to gain the right to forward packets.

Step 1 describes how the bucket is filled with Bc tokens to start each interval. Figure 16-3 shows a visual representation of the process. Note that if some of the tokens from the previous time interval are still in the bucket, some of the new tokens spill over the side of the bucket and are wasted.

Figure 16-3 *Mechanics of Filling the Shaping Token Bucket*

KEY POINT



Step 2 describes how the shaper spends the tokens. The shaper has to take tokens from the bucket equal to the number of bits in a packet in order to release that packet for transmission. For example, if the packet is 1000 bits long, the shaper must remove 1000 tokens from the bucket to send that packet. When traffic shaping tries to send a packet, and the bucket does not have enough tokens in it to buy the right to send the packet, traffic shaping must wait until the next interval, when the token bucket is refilled.

KEY POINT

Traffic shaping implements Be by making the single token bucket bigger, with no other changes to the token-bucket model. In other words, only Bc tokens are added each Tc, and tokens must still be consumed in order to send packets. The key difference using Be (versus not using Be) is that when some of the tokens are left in the bucket at the end of the time interval, and Bc tokens are added at the beginning of the next interval, more than Bc tokens are in the bucket—therefore allowing a larger burst of bits in this new interval.

Traffic-Shaping Adaptation on Frame Relay Networks

A shaper used with Frame Relay can be configured to vary the shaping rate over time based on the presence or absence of congestion. When there is no congestion, the shaper uses the shaping rate, but when congestion occurs, it lowers the shaping rate, eventually reaching a *minimum shaping rate*. The minimum rate can be configured, or default to 50 percent of the shaping rate. This lower rate is typically called either the *minimum information rate (MIR)* or the *mincir.*

To lower the rate, shapers must notice congestion via one of two methods:

- Receipt of a frame with the *Backward Explicit Congestion Notification (BECN)* bit set
- Receipt of a Cisco-proprietary *ForeSight* congestion message

Each time a BECN or ForeSight message is received, the shaper slows down by 25 percent of the maximum rate. To slow down, CB Shaping simply decreases Bc and Be by 25 percent, keeping the Tc value the same. If more BECNs or ForeSight messages are received, the Bc and Be settings are ratcheted down another 25 percent, until they bottom out at values that match the mincir. The rate grows again after 16 consecutive Tc values without a BECN or ForeSight congestion message. At that point, the shaping rate grows by $\frac{1}{16}$ of the shaping rate during each Tc, in this case by increasing the actual Bc and Be values used, until the maximum rate is reached again.

Class-Based Shaping Configuration

Class-Based Shaping (CB Shaping) implements all the core concepts described so far in this chapter, plus several other important features. First, it allows for several Cisco IOS queuing tools to be applied to the packets delayed by the shaping process. At the same time, it allows for fancy queuing tools to be used on the interface software queues. It also allows for classification of packets, so that some types of packets can be shaped at one rate, a second type of packet can be shaped at another rate, while allowing a third class of packets to not be shaped at all.

The only new MQC command required to configure CB Shaping is the **shape** command. The "Foundation Summary" section provides a CB Shaping command reference, in Table 16-9:

```
shape [average | peak] mean-rate [[burst-size] [excess-burst-size]]
```

CB Shaping can be implemented for output packets only, and it can be associated with either a physical interface or a subinterface.

To enable CB Shaping, the **service-policy output** command is configured under either the interface or the subinterface, with the referenced policy map including the **shape** command.

Example 16-1 shows a simple CB Shaping configuration that uses the following criteria:

- Interface clock rate is 128 kbps.
- Shape all traffic at a 64-kbps rate.
- Use the default setting for Tc.
- Shape traffic exiting subinterface s0/0.1.
- The software queuing on s0/0 will use WFQ (the default).
- The shaping queue will use FIFO (the default).

Example 16-1 *CB Shaping of All Traffic Exiting S0/0.1 at 64 kbps*

```
! Policy map shape-all places all traffic into the class-default class, matching
! all packets. All packets will be shaped to an average of 64 kbps. Note the
! units are in bits/second, so 64000 means 64 kbps.
policy-map shape-all
  class class-default
   shape average 64000
! The physical interface will not show the fair-queue command, but it is
! configured by default, implementing WFQ for interface s0/0 software queuing.
interface serial0/0
 bandwidth 128
! Below, CB Shaping has been enabled for all packets forwarded out s0/0.1.
interface serial0/0.1
 service-policy output shape-all
! Refer to the text after this example for more explanations of this next command.
R3# show policy-map interface s0/0.1
 Serial0/0.1
  Service-policy output: shape-all

    Class-map: class-default (match-any)
      7718 packets, 837830 bytes
      30 second offered rate 69000 bps, drop rate 5000 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
            64000/64000     2000   8000      8000      125       1000

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        —      56        6393      692696    6335      684964    yes
```

The configuration itself is relatively straightforward. The **shape-all** policy map matches all packets in a single class (class-default) and is enabled on s0/0.1. So, all packets exiting s0/0.1 will be shaped to the defined rate of 64 kbps.

The output of the **show policy-map interface s0/0.1** command shows the settings for all the familiar shaping concepts, but it uses slightly different terminology. CB Shaping defaults to a Bc and Be of 8000 bits each, listed under the columns **Sustain bits/int** (with "int" meaning "interval," or Tc) and **Excess bits/int**, respectively. The heading **Byte Limit** represents the size of the token bucket—the sum of Bc and Be, but listed as a number of bytes (2000 bytes in this case) instead of bits. The last column in that same part of the command output, **Increment (bytes)**, indicates how many bytes' worth of tokens are replenished each Tc. This value is equal to Bc (8000 bits), but the output is listed as a number of bytes (1000 bytes).

The CB Shaping **shape** command requires the shaping rate to be set. However, Bc and Be can be omitted, and Tc cannot be set directly. As a result, CB Shaping calculates some or all of these settings. CB Shaping calculates the values differently based on whether the shaping rate exceeds 320 kbps. Table 16-3 summarizes the rules.

Table 16-3 *CB Shaping Calculation of Default Variable Settings*

KEY POINT

Variable	Rate <= 320 kbps	Rate > 320 kbps
Bc	8000 bits	Bc = shaping rate * Tc
Be	Be = Bc = 8000	Be = Bc
Tc	Tc = Bc/shaping rate	25 ms

Tuning Shaping for Voice Using LLQ and a Small Tc

Example 16-1 in the previous section shows default settings for queuing for the interface software queues (WFQ) and for the shaping queue (FIFO). Example 16-2 shows an alternative configuration that works better for voice traffic by using LLQ for the shaped traffic. Also, the configuration forces the Tc down to 10 ms, which means that each packet will experience only a short delay waiting for the beginning of the next Tc. By keeping Tc to a small value, the LLQ logic applied to the shaped packets does not have to wait nearly as long to release packets from the PQ, as compared with the default Tc settings.

The revised requirements, as compared with Example 16-1, are as follows:

- Enable LLQ to support a single G.729 voice call.
- Shape to 96 kbps—less than the clock rate (128 kbps), but more than the CIR of the VC.
- Tune Tc to 10 ms.

Example 16-2 *CB Shaping on R3, 96-kbps Shape Rate, with LLQ for Shaping Queues*

```
class-map match-all voip-rtp
  match ip rtp 16384 16383
! queue-voip implements a PQ for VoIP traffic, and uses WFQ in the default class.
policy-map queue-voip
  class voip-rtp
   priority 32
  class class-default
   fair-queue
! shape-all shapes all traffic to 96 kbps, with Bc of 960. Tc is calculated as
! 960/96000 or 10 ms. Also note the service-policy queue-voip command. This applies
! policy map queue-voip to all packets shaped by the shape command.
policy-map shape-all
  class class-default
   shape average 96000 960
   service-policy queue-voip
!
interface serial0/0.1
 service-policy output shape-all
! Note the Interval is now listed as 10 ms. Also, note the detailed stats for LLQ
! are also listed at the end of the command.
R3# show policy-map interface serial 0/0.1
 Serial0/0.1
  Service-policy output: shape-all

    Class-map: class-default (match-any)
      5189 packets, 927835 bytes
      30 second offered rate 91000 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           96000/96000      1200      960       960       10        120

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        —      17        5172      910975    4002      831630    yes

      Service-policy : queue-voip
        Class-map: voip-rtp (match-all)
          4623 packets, 295872 bytes
          30 second offered rate 25000 bps, drop rate 0 bps
          Match: ip rtp 16384 16383
          Weighted Fair Queueing
            Strict Priority
            Output Queue: Conversation 24
            Bandwidth 32 (kbps) Burst 800 (Bytes)
```

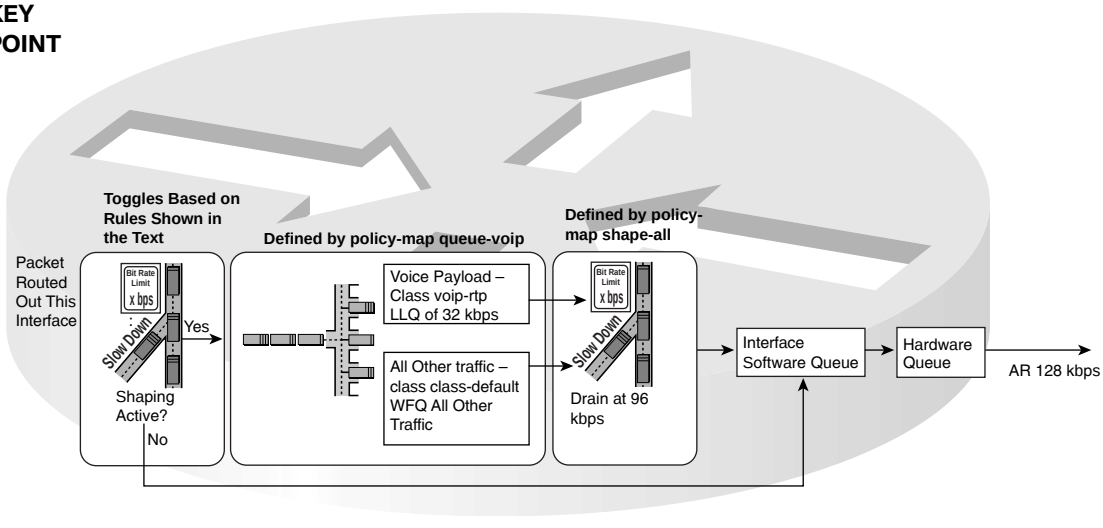
Example 16-2 *CB Shaping on R3, 96-kbps Shape Rate, with LLQ for Shaping Queues (Continued)*

```
            (pkts matched/bytes matched) 3528/225792
            (total drops/bytes drops) 0/0

        Class-map: class-default (match-any)
          566 packets, 631963 bytes
          30 second offered rate 65000 bps, drop rate 0 bps
          Match: any
          Weighted Fair Queueing
            Flow Based Fair Queueing
            Maximum Number of Hashed Queues 16
(total queued/total drops/no-buffer drops) 17/0/0
```

Example 16-2 shows how to use LLQ against the packets shaped by CB Shaping by calling an LLQ policy map with the **service-policy** command. Note the command syntax (**service-policy queue-voip**) does not include the **output** keyword; the output direction is implied. Figure 16-4 shows the general idea behind what is happening in the configuration.

Figure 16-4 *Interaction Between Shaping Policy Map* **shape-all** *and Queuing Policy Map* **queue-voip**

KEY POINT



KEY POINT Scanning Figure 16-4 from left to right, CB Shaping must make the first decision after a packet has been routed out the subinterface. CB Shaping first needs to decide if shaping is active; if it is, CB Shaping should put the packet into a shaping queue. If it is not active, the packet can move right on to the appropriate interface software queue. Shaping becomes active when a single packet exceeds the traffic contract; shaping only becomes inactive again when all the shaping queues are drained.

Assuming that a packet needs to be delayed by CB Shaping, the LLQ logic of **policy-map queue-voip** determines into which of the two shaping queues the packet should be placed. Later, when CB Shaping decides to release the next packet (typically when the next Tc begins), LLQ

determines which packets are taken next. This example has only two queues, one of which is an LLQ, so packets are always taken from the LLQ if any are present in that queue.

When a packet leaves one of the two shaping queues, it drains into the interface software queues. For routers with many VCs on the same physical interface, the VCs compete for the available interface bandwidth. Examples 16-1 and 16-2 both defaulted to use WFQ on the interface. However, LLQ or CBWFQ could have been used on the interface in addition to its use on the shaping function, simply by adding a **service-policy output policy-map-name** command under s0/0.

NOTE When one policy map refers to another, as in Example 16-2, the configurations are sometimes called "hierarchical" policy maps. Other times, they are called "nested" policy maps. Or, you can just think of it as how CBWFQ and LLQ can be configured for the shaping queues.

Configuring Shaping by Bandwidth Percent

The **shape** command allows the shaping rate to be stated as a percentage of the setting of the interface or subinterface **bandwidth** setting. Configuring based on a simple percentage of the bandwidth command setting seems obvious at first. However, you should keep in mind the following facts when configuring the **shape** command based on percentage of interface bandwidth:

KEY POINT

- The **shape percent** command uses the bandwidth of the interface or subinterface under which it is enabled.
- Subinterfaces do not inherit the bandwidth setting of the physical interface, so if it not set via the **bandwidth** command, it defaults to 1544.
- The Bc and Be values are configured as a number of milliseconds; the values are calculated as the number of bits that can be sent at the configured shaping rate, in the configured time period.
- Tc is set to the configured Bc value, which is in milliseconds.

Example 16-3 shows a brief example of CB Shaping configuration using percentages, including explanations of the points from the preceding list.

Example 16-3 *Shaping Based on Percent*

```
! With s0/0.1 bandwidth of 128, the rate is 50% * 128, or 64 kbps. At 64 kbps, 8000
! bits can be sent in the configured 125-ms time interval (64000 * 0.125 = 8000).
! Note that the ms parameter in the shape command is required after the Bc
! (shown) or Be (not shown), otherwise the command is rejected. Not shown: The
! Tc was set to 125 ms, the exact value configured for Bc.
policy-map percent-test
  class class-default
   shape average percent 50 125 ms
interface Serial0/1
 bandwidth 128
 service-policy output percent-test
```

CB Shaping to a Peak Rate

The **shape average** command has been used in all the examples so far. However, the command **shape peak** *mean-rate* is also allowed, which implements slightly different behavior as compared with **shape average** for the same configured rate. The key actions of the **shape peak** *mean-rate* command are summarized as follows:

KEY POINT

- It calculates (or defaults) Bc, Be, and Tc the same way as the **shape average** command.
- It refills Bc + Be tokens (instead of just Bc tokens) into the token bucket for each time interval.

This logic means that CB Shaping gets the right to send the committed burst, and the excess burst, every time period. As a result, the actual shaping rate is as follows:

KEY POINT

Shaping_rate = configured_rate (1 + Be/Bc)

For instance, the **shape peak 64000** command, with Bc and Be defaulted to 8000 bits each, results in an actual shaping rate of 128 kbps, based on the following formula:

64 (1 + 8000/8000) = 128

Adaptive Shaping

Adaptive shaping configuration requires only a minor amount of effort compared to the topics covered so far. To configure it, just add the **shape adaptive** *min-rate* command under the **shape** command. Example 16-4 shows a short example.

Example 16-4 *Adaptive CB Shaping Configuration*

```
policy-map shape-all
  class class-default
   shape average 96000 9600 ms
   shape adaptive 32000
```

Frame Relay Traffic Shaping Configuration

Frame Relay Traffic Shaping (FRTS) differs from CB Shaping in several significant ways, although the underlying token-bucket mechanics are identical. The following list highlights some of the key similarities and differences:

KEY POINT

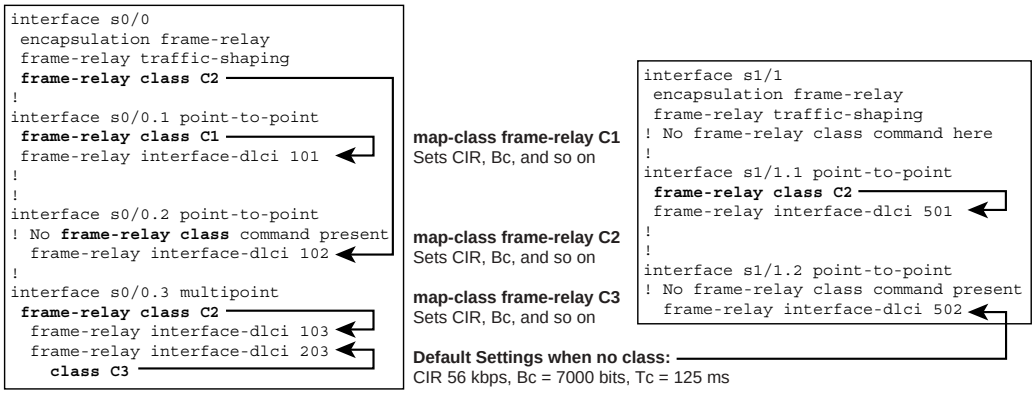
- FRTS can be used only on Frame Relay interfaces, whereas CB Shaping can be used with any underlying data link protocol.
- Like CB Shaping, FRTS allows a large number of IOS queuing tools to be used instead of a single FIFO shaping queue.

- Unlike CB Shaping, FRTS does not allow any fancy queuing tools to be enabled on the physical interface concurrent with FRTS.
- FRTS *always* shapes the traffic on each VC separately.
- FRTS cannot classify traffic in order to shape a subset of traffic on a particular VC.
- Unlike CB Shaping, FRTS can dynamically learn the CIR, Bc, and Be values configured on the Frame Relay switch by using the *Enhanced Local Management Interface (ELMI)* feature.

Additionally, FRTS does not use MQC commands, making the configuration significantly different from that of CB Shaping. FRTS organizes a set of shaping parameters (rate, Bc, and so on) into a named Frame Relay map class, using the **map-class frame-relay** command. The **frame-relay class** command and the **class** command then refer to those map classes, defining the shaping parameters to use for each Frame Relay VC. Figure 16-5 shows several examples of how these commands work together.

Figure 16-5 *Assignment of Map Classes to DLCIs with FRTS*

KEY POINT



As Figure 16-5 illustrates, FRTS uses the map class referenced by the **class** command under the **frame-relay interface-dlci** command, if it exists (example: DLCI 203). If not, FRTS assigns the map class based on the subinterface's **frame-relay class** command (example: DLCI 103). Otherwise, FRTS looks for the setting on the physical interface (example: DLCI 102). If FRTS still has not found a reference to a map class, it uses default settings for that VC (example: DLCI 502). (Beware of enabling FRTS and not setting a VC's shaping parameters, especially if you want to get more than 56 kbps out of that VC!) These rules can be summarized as follows:

KEY POINT

- If the **class** *map-class-name* command is configured under the **interface-dlci** command, that map class defines the FRTS parameters for that VC.
- If not, if the **frame-relay class** *map-class-name* command is configured under the subinterface, that map class defines the FRTS parameters for the remaining underlying VCs.

- If not, if the **frame-relay class** *map-class-name* command is configured under the physical interface, that map class defines the FRTS parameters for the remaining underlying VCs.
- If not, FRTS uses the default settings of shaping at 56 kbps, Bc = 7000 bits, and Tc = 125 ms.

FRTS Configuration Using the traffic-rate Command

FRTS uses two main styles of configuration for the shaping parameters. The **frame-relay traffic-rate** *average* [*peak*] command configures the average and peak rate, with Cisco IOS calculating Bc and Be with an assumed Tc of 125 ms. This method is simpler to configure, but offers no ability to tune Tc or set Bc and Be.

Example 16-5 uses FRTS to implement the same requirements as the first CB Shaping example shown in Example 16-1, except that it uses FIFO queuing for the interface software queues.

Example 16-5 *FRTS Configuration, 64 kbps, with the* **frame-relay traffic-rate** *Command*

```
! The frame-relay traffic-shaping command enables FRTS for all VCs on s0/0. The
! frame-relay class shape-all-64 command refers to a map class.
interface Serial0/0
 encapsulation frame-relay
 frame-relay traffic-shaping
!
interface Serial0/0.1 point-to-point
 frame-relay class shape-all-64
 frame-relay interface-dlci 101
! lines omitted for brevity
! Above, note that the frame-relay class shape-all-64 command could have been
! listed under S0/0 instead, with the same results, as only one VC exists on the
! interface. Alternately, the class shape-all-64 command could have been used
! under the frame-relay interface-dlci 101 command.
! Next, The traffic-rate command sets the peak equal to the average, which results
! in a Be of 0.
map-class frame-relay shape-all-64
 frame-relay traffic-rate 64000 64000
! The show frame pvc command, with no DLCI listed, does not list FRTS info, but
! it does show FRTS info when the specific DLCI is given. The word "fifo" refers
! to the shaping queue.
R3# show frame-relay pvc 101
PVC Statistics for interface Serial0/0 (Frame Relay DTE)
DLCI = 101, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0.1
! lines omitted for brevity
  shaping active
  traffic shaping drops 2774
  Queueing strategy: fifo
  Output queue 3/40, 678 drop, 3777 dequeued
! The next command shows the default 125-ms Tc, the calculated Bc = Tc * CIR,
```

continues

Example 16-5 *FRTS Configuration, 64 kbps, with the* **frame-relay traffic-rate** *Command (Continued)*

```
! and it uses the same text in the headings as in the CB Shaping examples.
R3# show traffic-shape
Interface   Se0/0.1
       Access Target    Byte   Sustain   Excess    Interval  Increment Adapt
VC     List   Rate      Limit  bits/int  bits/int  (ms)      (bytes)   Active
101           64000     1000   64000     0         125       1000      —
! This command lists basic stats for FRTS. The "fcfs" refers to the shaping queue
! as well, meaning "first come first served," which means the same thing as "fifo."
R3# show traffic-shape queue
Traffic queued in shaping queue on Serial0/0.1 dlci 101
  Queueing strategy: fcfs
  Queueing Stats: 23/40/959 (size/max total/drops)
! lines omitted for brevity
```

To use the **frame-relay traffic-rate** command to use a Be, the peak rate must be configured, and it must be more than the average rate. This command causes FRTS to calculate Be based on this formula:

KEY POINT

$$Be = Tc * (PIR - CIR)$$

In Example 16-5, Be = 0.125 * (64,000 – 64,000) = 0, as shown in the output of the **show traffic-shape** command in the example. However, if the **frame-relay traffic-rate 64000 96000** command had been used, the Be would be .125 (96,000 – 64,000) = 4000.

Setting FRTS Parameters Explicitly

The **frame-relay cir**, **frame-relay Bc**, and **frame-relay Be** commands can be used to directly set FRTS parameters in an FRTS map class, instead of setting the Bc, Be, and Tc values indirectly using the **frame-relay traffic-rate** command. Example 16-6 shows two new map classes on the same router configured in Example 16-5. These new map classes use these additional commands to set FRTS parameters explicitly, which is particularly useful for tuning FRTS to use a small Tc.

Example 16-6 *FRTS Configuration by Setting CIR and BC to Manipulate Tc*

```
! map-class shape-all-64-long sets CIR and Bc directly, defaulting Be to 0, with
! Tc calculated via Tc = Bc/CIR
map-class frame-relay shape-all-64-long
 frame-relay cir 64000
 frame-relay bc 8000
! All VCs on s0/0.1 that do not have class commands will use shape-all-64-long.
R3(config)# interface serial 0/0.1
R3(config-subif)# frame class shape-all-64-long
R3(config-subif)# ^Z
! This command confirms the configured rate, with the Tc calculated as Bc/rate, or
```

Example 16-6 *FRTS Configuration by Setting CIR and BC to Manipulate Tc (Continued)*

```
! in this case, 8000/64000. Note the default Be of 0 is also listed.
R3# show traffic-shape
Interface   Se0/0.1
       Access Target    Byte   Sustain   Excess    Interval  Increment Adapt
VC     List   Rate      Limit  bits/int  bits/int  (ms)      (bytes)   Active
101           64000     1000    8000     0         125       1000      -
! The next commands create another map class, with the Bc set to 1/100th
! of the shaping rate (10 ms).
R3(config)# map-class frame-relay shape-all-64-shortTC
R3(config-map-class)# frame-relay cir 64000
R3(config-map-class)# frame-relay bc 640
R3(config-map-class)# int s 0/0.1
R3(config-subif)# frame class shape-all-64-shortTC
R3# show traffic-shape
Interface   Se0/0.1
       Access Target    Byte   Sustain   Excess    Interval  Increment Adapt
VC     List   Rate      Limit  bits/int  bits/int  (ms)      (bytes)   Active
101           64000     80     640       0         10        80        -
```

FRTS Configuration Using LLQ

FRTS supports a variety of queuing tools for managing packets it queues. The queuing tool is enabled via a command in the map class. Example 16-7 shows just such an example, with a new map class. The requirements implemented in this example are as follows:

- Shape traffic on the two VCs (101 and 102) on s0/0 with the same settings for shaping.
- Use LLQ only on the VC with DLCI 101.
- Set Be to 0, and tune Tc to 10 ms.

Note that the example does not show the configuration for policy map **queue-voip**. Its full configuration can be seen back in Example 16-2.

Example 16-7 *FRTS to Two Sites, with LLQ Used to Shape the Queue to Site 1*

```
R3# show running-config
! FRTS is first enabled, and class shape-all-96 is set up to filter down to the
! remaining VCs, assuming no other frame-relay class or class subcommands are applied
! to them.
interface Serial0/0
 encapsulation frame-relay
 frame-relay class shape-all-96
 frame-relay traffic-shaping
! DLCI 101 will use class shape-with-LLQ based on the next few commands.
interface Serial0/0.1 point-to-point
```

continues

Example 16-7 *FRTS to Two Sites, with LLQ Used to Shape the Queue to Site 1 (Continued)*

```
 frame-relay class shape-with-LLQ
 frame-relay interface-dlci 101
! DLCI 102 will use class shape-all-96 because it is configured under s0/0.
interface Serial0/0.2 point-to-point
 frame-relay interface-dlci 102
! The only difference between the two map classes is the service-policy output
! voip-and-allelse command, which enables LLQ in the shape-with-LLQ class.
map-class frame-relay shape-all-96
 frame-relay cir 96000
 frame-relay bc 960
 frame-relay be 0
!
map-class frame-relay shape-with-LLQ
 frame-relay cir 96000
 frame-relay bc 960
 frame-relay be 0
 service-policy output queue-voip
! The show policy-map interface command does not show any LLQ stats with FRTS.
! Instead, the show frame-relay pvc DLCI command is required, with output similar
! to the show policy-map interface command.
R3# show frame-relay pvc 101
PVC Statistics for interface Serial0/0 (Frame Relay DTE)
DLCI = 101, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0.1
! lines omitted for brevity
  shaping active
  traffic shaping drops 0
  service policy queue-voip
 Serial0/0.1: DLCI 101 -

  Service-policy output: queue-voip
    Class-map: voip-rtp (match-all)
      5101 packets, 326464 bytes
      30 second offered rate 25000 bps, drop rate 0 bps
      Match: ip rtp 16384 16383
      Weighted Fair Queueing
        Strict Priority
! lines omitted for brevity
```

FRTS Adaptive Shaping

Adding FRTS adaptive shaping configuration to an existing FRTS configuration is relatively simple. To enable it, do the following:

KEY POINT

1. Add either a **frame-relay adaptive-shaping becn** or **frame-relay adaptive-shaping foresight** command into the appropriate map class.

2. To set the minimum to something other than the default of 50 percent of the shaping rate, add the **frame-relay mincir** *rate* command in the map class.

Policing Concepts and Configuration

Class-Based Policing (CB Policing) performs different internal processing than the older, alternative policer in Cisco router IOS, namely committed access rate (CAR). This section focuses on CB Policing, starting with concepts and then covering configuration details.

CB Policing Concepts

CB Policing is enabled for packets either entering or exiting an interface, or those entering or exiting a subinterface. It monitors, or *meters*, the bit rate of the combined packets; when a packet pushes the metered rate past the configured policing rate, the policer takes action against that packet. The most aggressive action is to discard the packet. Alternately, the policer can simply re-mark a field in the packet. This second option allows the packets through, but if congestion occurs at later places during a marked-down packet's journey, it is more likely to be discarded.

Table 16-4 lists the keywords used to imply the policer's actions.

Table 16-4 *Policing Actions Used CB Policing*

KEY POINT

Command Option	Mode and Function
drop	Drops the packet
set-dscp-transmit	Sets the DSCP and transmits the packet
set-prec-transmit	Sets the IP Precedence (0 to 7) and sends the packet
set-qos-transmit	Sets the QoS Group ID (1 to 99) and sends the packet
set-clp-transmit	Sets the ATM CLP bit (ATM interfaces only) and sends the packet
set-fr-de	Sets the Frame Relay DE bit (Frame Relay interfaces only) and sends the packet
transmit	Sends the packet

CB Policing categorizes packets into two or three categories, depending on the style of policing, and then applies one of these actions to each category of packet. The categories are *conforming* packets, *exceeding* packets, and *violating* packets. The CB Policing logic that dictates when packets are placed into a particular category varies based on the type of policing. The next three sections outline the types of CB Policing logic.

Single-Rate, Two-Color Policing (One Bucket)

Single-rate, two-color policing is the simplest option for CB Policing. This method uses a single policing rate with no excess burst. The policer will then use only two categories (*conform* and *exceed*), defining a different action on packets of each type. (Typically, the conform action is to transmit the packet, with the exceed action either being to drop the packet or mark it down.)

While this type of policing logic is often called *single-rate, two-color* policing, it is sometimes called *single-bucket two-color* policing because it uses a single token bucket for internal processing. Like shaping's use of token buckets, the policer's main logic relates to filling the bucket with tokens, and then spending the tokens. Over time, the policer refills the bucket according to the policing rate. For instance, policing at 96 kbps, over the course of 1 second, adds 12,000 tokens to the bucket. (A token represents a byte with policers, so 12,000 tokens is 96,000 bits' worth of tokens.)

CB Policing does not refill the bucket based on a time interval. Instead, CB Policing reacts to the arrival of a packet by replenishing a prorated number of tokens into the bucket. The number of tokens is defined by the following formula:

$$\frac{(\text{Current_packet_arrival_time} - \text{Previous_packet_arrival_time}) * \text{Police_rate}}{8}$$

NOTE Note that a token represents the right to send 1 byte, so the formula includes the division by 8 to convert the units to bytes instead of bits.

The idea behind the formula is simple—essentially, a small number of tokens are replenished before each packet is policed; the end result is that tokens are replenished at the policing rate. For example, for a police rate of 128 kbps, the policer should replenish 16,000 tokens per second. If 1 second has elapsed since the previous packet arrived, CB Policing would replenish the bucket with 16,000 tokens. If 0.1 second has passed since the previous packet had arrived, CB Policing would replenish the bucket with 0.1 second's worth of tokens, or 1600 tokens. If 0.01 second had passed, CB Policing would replenish 160 tokens at that time.

The policer then considers whether it should categorize the newly arrived packet as either conforming or exceeding the traffic contract. The policer compares the number of bytes in the packet (represented here as *X*p, with "p" meaning "packet") to the number of tokens the token bucket (represented here as *X*b, with "b" meaning "bucket"). Table 16-5 shows the decision logic, along with whether the policer spends/removes tokens from the bucket.

Table 16-5 *Single-Rate, Two-Color Policing Logic for Categorizing Packets*

Category	Requirements	Tokens Drained from Bucket
Conform	If *X*p <= *X*b	*X*p tokens
Exceed	If *X*p > *X*b	None

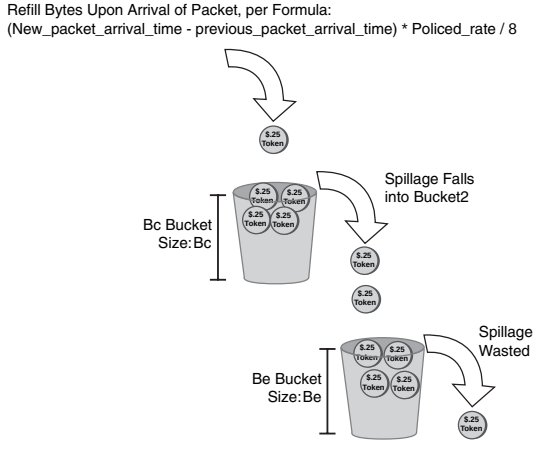
As long as the overall bit rate does not exceed the policing rate, the packets will all conform. However, if the rate is exceeded, then as tokens are removed for each conforming packet, the bucket will eventually empty—causing some packets to exceed the contract. Over time, tokens are added back to the bucket, so some packets will conform. Once the bit rate lowers below the policing rate, all packets will again conform to the contract.

Single-Rate, Three-Color Policer (Two Buckets)

When you want the policer to police at a particular rate, but to also support a Be, the policer uses two token buckets. It also uses all three categories for packets—conform, exceed, and violate. Combining those concepts together, such policing is typically called *single-rate, three-color policing*.

As before, CB Policing fills the buckets in reaction to packet arrival. (For lack of a better set of terms, this discussions calls the first bucket the Bc bucket, because it is Bc in size, and the other one the Be bucket, because it is Be in size.) CB Policing fills the Bc bucket just like a single-bucket model. However, if the Bc bucket has any tokens left in it, some will spill; these tokens then fill the Be bucket. Figure 16-6 shows the basic process.

Figure 16-6 *Refilling Dual Token Buckets with CB Policing*



After filling the buckets, the policer then determines the category for the newly arrived packet, as shown in Table 16-6. In this case, *X*bc is the number of tokens in the Bc bucket, and *X*be is the number in the Be bucket.

Table 16-6 *Single-Rate Three-Color Policing Logic for Categorizing Packets*

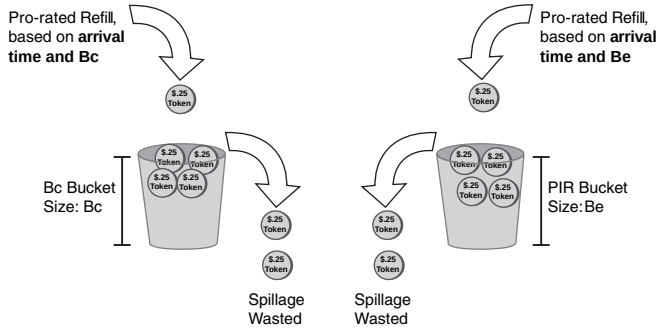
Category	Requirements	Tokens Drained from Bucket
Conform	*X*p <= *X*bc	*X*p tokens from the Bc bucket
Exceed	*X*p > *X*bc and *X*p <= *X*be	*X*p tokens from the Be bucket
Violate	*X*p > *X*bc and *X*p > *X*be	None

Two-Rate, Three-Color Policer (Two Buckets)

The third main option for CB Policing uses two separate policing rates. The lower rate is the previously discussed committed information rate (CIR), and the higher, second rate is called the *peak information rate (PIR)*. Packets that fall under the CIR conform to the traffic contract. Packets that exceed the CIR, but fall below PIR, are considered to exceed the contract. Finally, packets beyond the PIR are considered to violate the contract.

The key difference between the single-rate and dual-rate three-color policers is that the dual-rate method essentially allows sustained excess bursting. With a single-rate, three-color policer, an excess burst exists, but the burst is sustained only until the Be bucket empties. A period of relatively low activity has to occur to refill the Be bucket. With the dual-rate method, the Be bucket does not rely on spillage when filling the Bc bucket, as depicted in Figure 16-7. (Note that these buckets are sometimes called the CIR and PIR buckets with dual-rate policing.)

Figure 16-7 *Refilling CIR and PIR Dual Token Buckets*



The refilling of the two buckets based on two different rates is very important. For example, imagine you set a CIR of 128 kbps (16 kilobytes/second), and a PIR of 256 kbps (32 kBps). If 0.1 second passed before the next packet arrived, then the CIR bucket would be replenished with 1600 tokens (1/10 of 1 second's worth of tokens, in bytes), while the PIR bucket would be replenished with 3200 tokens. So, there are more tokens to use in the PIR bucket, as compared to the CIR bucket.

Next, the policer categorizes the packet. The only difference in logic as compared with the single-rate, three-color policer is highlighted in Table 16-7, specifically related to how tokens are consumed for conforming packets.

Table 16-7 *Two-Rate, Three-Color Policing Logic for Categorizing Packets*

Category	Requirements	Tokens Drained from Bucket
Conform	*X*p <= *X*bc	*X*p tokens from the Bc bucket AND *X*p tokens from the Be bucket
Exceed	*X*p > *X*bc and *X*p <= *X*be	*X*p tokens from the Be bucket
Violate	*X*p > Xbc and *X*p > Xbe	None

While Table 16-7 does outline each detail, the underlying logic might not be obvious from the table. In effect, by filling the Be bucket based on the higher PIR, but also draining tokens from the Be bucket for packets that conform to the lower CIR, the Be bucket has tokens that represent the difference between the two rates.

Class-Based Policing Configuration

CB Policing uses the familiar MQC commands for configuration. As a result, a policy map can police all packets using the convenient class-default class, or it can separate traffic into classes, apply different policing parameters to different classes of traffic, or even simply not police some classes.

The **police** command configures CB Policing inside a policy map. On the **police** command, you define the policing rate in bps, the Bc in bytes, and the Be in bytes, along with the actions for each category:

```
police bps burst-normal burst-max conform-action action exceed-action action
  [violate-action action]
```

Single-Rate, Three-Color Policing of All Traffic

Example 16-8 shows how to police all traffic, with criteria as follows:

- Create a single-rate, three-color policing configuration.
- All traffic policed at 96 kbps at ingress.
- Bc of 1 second's worth of traffic is allowed.
- Be of 0.5 second's worth of traffic is allowed.
- The conform, exceed, and violate actions should be to forward, mark down to DSCP 0, and discard, respectively.

Example 16-8 *Single-Rate, Three-Color CB Policing at 96 kbps*

```
! The police command sets the rate (in bps), Bc and Be (in bytes), and the three
! actions.
policy-map police-all
  class class-default
! note: the police command wraps around to a second line.
  police cir 96000 bc 12000 be 6000 conform-action transmit exceed-action set-dscp-
  transmit 0 violate-action drop
!
interface Serial1/0
 encapsulation frame-relay
 service-policy input police-all
```

continues

Example 16-8 *Single-Rate, Three-Color CB Policing at 96 kbps (Continued)*

```
! The show command below lists statistics for each of the three categories.
ISP-edge# show  policy-map  interface s 1/0
 Serial1/0
  Service-policy input: police-all

    Class-map: class-default (match-any)
      8375 packets, 1446373 bytes
      30 second offered rate 113000 bps, drop rate 15000 bps
      Match: any
      police:
        cir 96000 bps, conform-burst 12000, excess-burst 6000
        conformed 8077 packets, 1224913 bytes; action: transmit
        exceeded 29 packets, 17948 bytes; action: set-dscp-transmit 0
        violated 269 packets, 203512 bytes; action: drop
        conformed 95000 bps, exceed 0 bps violate 20000 bps
```

The **police** command defines a single rate, but the fact that it is a three-color policing configuration, and not a two-color configuration, is not obvious at first glance. To configure a single-rate, three-color policer, you need to configure a violate action or explicitly set Be to something larger than 0.

Policing a Subset of the Traffic

One of the advantages of CB Policing is the ability to perform policing per class. Example 16-9 shows CB Policing with HTTP traffic classified and policed differently than the rest of the traffic, with the following criteria:

- Police web traffic at 80 kbps at ingress to the ISP-edge router. Transmit conforming and exceeding traffic, but discard violating traffic.
- Police all other traffic at 16 kbps at ingress to the ISP-edge router. Mark down exceeding and violating traffic to DSCP 0.
- For both classes, set Bc and Be to 1 second's worth and .5 second's worth of traffic, respectively.

Example 16-9 *CB Policing 80 kbps for Web Traffic, 16 kbps for the Rest with Markdown to Be, at ISP-Edge Router*

```
class-map match-all match-web
  match protocol http
! The new policy map uses the new class to match http, and class-default to
! match all other traffic.
policy-map police-web
  class match-web
```

Example 16-9 *CB Policing 80 kbps for Web Traffic, 16 kbps for the Rest with Markdown to Be, at ISP-Edge Router (Continued)*

```
    police cir 80000 bc 10000 be 5000 conform-action transmit exceed-action transmit
violate-action drop
  class class-default
    police cir 16000 bc 2000 be 1000 conform-action transmit exceed-action
transmit violate-action set-dscp-transmit 0
!
interface Serial1/0
 encapsulation frame-relay
 service-policy input police-web
```

CB Policing Defaults for Bc and Be

If you do not configure a Bc value on the **police** command, then CB Policing configures a default value equivalent to the bytes that could be sent in $^1/_4$ second at the defined policing rate. The formula is as follows:

$$Bc = \frac{(CIR * 0.25 \text{ second})}{8 \text{ bits/byte}} = \frac{CIR}{32}$$

The only part that may not be obvious is the division by 8 on the left—that is simply for the conversion from bits to bytes. The math reduces to CIR/32. Also, if the formula yields a number less than 1500, CB Policing uses a Bc of 1500.

If the **police** command does not include a Be value, the default Be setting depends on the type of policing. Table 16-8 summarizes the details.

Table 16-8 *Setting CB Policing Bc and Be Defaults*

KEY POINT

Type of Policing Configuration	Telltale Signs in the police Command	Defaults
Single rate, two color	No **violate-action** configured	Bc = CIR/32; Be = 0
Single rate, three color	**violate-action** is configured	Bc = CIR/32; Be = Bc
Dual rate, three color	PIR is configured	Bc = CIR/32; Be = PIR/32

Configuring Dual-Rate Policing

Dual-rate CB Policing requires the same MQC commands, but with slightly different syntax on the **police** command, as shown here:

```
police {cir cir} [bc conform-burst] {pir pir} [be peak-burst]
  [conform-action action [exceed-action action [violate-action action]]]
```

Note that the syntax of this command requires configuration of both the CIR and a PIR because the curly brackets mean that the parameter is required. The command includes a place to set the Bc value and the Be value as well, plus the same set of options for conform, exceed, and violate actions. For example, if you wanted to perform dual-rate policing, with a CIR of 96 kbps and a PIR of 128 kbps, you would simply use a command like **police cir 96000 pir 128000**, with optional setting of Bc and Be, plus the settings for the actions for each of the three categories.

Multi-Action Policing

When CB Policing re-marks packets instead of discarding them, the design might call for marking more than one field in a packet. For instance, when transmitting into a Frame Relay cloud, it might be useful to mark both DSCP and FR DE when a packet violates the contract. Marking multiple fields in the same packet with CB Policing is called *multi-action policing*.

The **police** command uses a slightly different syntax to implement multi-action policing. By omitting the actions from the command, the **police** command places the user into a policing subconfiguration mode in which the actions can be added via separate commands (the **conform-action**, **exceed-action**, and **violate-action** commands). To configure multiple actions, one of these three **action** commands would be used more than once, as shown in Example 16-10, which marks DSCP 0 and sets FR DE for packets that violate the traffic contract.

Example 16-10 *Multi-Action Policing*

```
R3# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R3(config)# policy-map testpol1
R3(config-pmap)# class class-default
! This command implements dual-rate policing as well, but it is not required
R3(config-pmap-c)# police 128000 256000
R3(config-pmap-c-police)# conform-action transmit
R3(config-pmap-c-police)# exceed-action transmit
R3(config-pmap-c-police)# violate-action set-dscp-transmit 0
R3(config-pmap-c-police)# violate-action set-frde-transmit
```

Policing by Percentage

As it does with the **shape** command, Cisco IOS supports configuring policing rates as a percentage of link bandwidth. The Bc and Be values are configured as a number of milliseconds, from which IOS calculates the actual Bc and Be values based on how many bits can be sent in that many milliseconds. Example 16-11 shows an example of a dual-rate policing configuration using the **percentage** option.

Example 16-11 *Configuring Percentage-Based Policing*

```
R3# show running-config
! Portions omitted for Brevity
 policy-map test-pol6
  class class-default
   police cir percent 25 bc 500 ms pir percent 50 be 500 ms conform transmit exceed transmit
      violate drop
!
interface serial0/0
 bandwidth 256
 service-policy output test-pol6
! The output below shows the configured percentage for the rate and the time for
! Bc and Be, with the calculated values immediately below.
R3# show policy-map interface s0/0
! lines omitted for brevity
      police:
          cir 25 % bc 500 ms
          cir 64000 bps, bc 4000 bytes
          pir 50 % be 500 ms
          pir 128000 bps, be 8000 bytes
! lines omitted
```

Committed Access Rate

CAR implements single-rate, two-color policing. As compared with that same option in CB Policing, CAR and CB Policing have many similarities. They both can police traffic either entering or exiting an interface or subinterface; they can both police subsets of that traffic based on classification logic; and they both set the rate in bps, with Bc and Be configured as a number of bytes.

CAR differs from CB Policing regarding four main features, as follows:

KEY POINT

- CAR uses the **rate-limit** command, which is not part of the MQC set of commands.
- CAR has a feature called *cascaded* or *nested* **rate-limit** commands, which allows multiple **rate-limit** commands on an interface to process the same packet.
- CAR does support Be; however, even in this case, it still supports only conform and exceed categories, and never supports a third (violate) category.
- When CAR has a Be configured, the internal logic used to determine which packets conform and exceed differs as compared with CB Policing.

CAR puts most parameters on the **rate-limit** command, which is added under an interface or subinterface:

```
rate-limit {input | output} [access-group [rate-limit] acl-index] bps burst-normal
  burst-max conform-action conform-action exceed-action exceed-action
```

Example 16-12 shows an example CAR configuration for perspective. The criteria for the CAR configuration in Example 16-12 are as follows:

- All traffic policed at 96 kbps at ingress to the ISP-edge router.
- Bc of 1 second's worth of traffic is allowed.
- Be of 0.5 second's worth of traffic is allowed.
- Traffic that exceeds the contract is discarded.
- Traffic that conforms to the contract is forwarded with Precedence reset to 0.

Example 16-12 *CAR at 96 kbps at ISP-Edge Router*

```
! The rate-limit command omits the access-group option, meaning that it has no matching
! parameters, so all packets are considered to match the command. The rest of the
! options simply match the requirements.
interface Serial1/0.1 point-to-point
ip address 192.168.2.251 255.255.255.0
! note: the rate-limit command wraps around to a second line.
 rate-limit input 96000 12000 18000 conform-action set-prec-transmit 0
   exceed-action drop
 frame-relay interface-dlci 103
! The output below confirms the parameters, including matching all traffic.
ISP-edge# show interfaces s 1/0.1 rate-limit
  Input
    matches: all traffic
      params:  96000 bps, 12000 limit, 18000 extended limit
      conformed 2290 packets, 430018 bytes; action: set-prec-transmit 0
      exceeded 230 packets, 67681 bytes; action: drop
      last packet: 0ms ago, current burst: 13428 bytes
last cleared 00:02:16 ago, conformed 25000 bps, exceeded 3000 bps
```

To classify traffic, CAR requires the use of either a normal ACL or a *rate-limit ACL*. A rate-limit ACL can match MPLS Experimental bits, IP Precedence, or MAC Address. For other fields, an IP ACL must be used. Example 16-13 shows an example in which CAR polices three different subsets of traffic using ACLs for matching the traffic, as well as limiting the overall traffic rate. The criteria for this example are as follows (Note that CAR allows only policing rates that are multiples of 8 kbps):

- Police all traffic on the interface at 496 kbps; but before sending this traffic on its way…
- Police all web traffic at 400 kbps.
- Police all FTP traffic at 160 kbps.
- Police all VoIP traffic at 200 kbps.
- Choose Bc and Be so that Bc has 1 second's worth of traffic, and Be provides no additional burst capability over Bc.

Example 16-13 *Cascaded CAR* **rate-limit** *Commands, with Subclassifications*

```
! ACL 101 matches all HTTP traffic
! ACL 102 matches all FTP traffic
! ACL 103 matches all VoIP traffic
interface s 0/0
rate-limit input 496000 62000 62000 conform-action continue exceed-action drop
rate-limit input access-group 101 400000 50000 50000 conform-action transmit exceed-action
drop
rate-limit input access-group 102 160000 20000 20000 conform-action transmit exceed-action
drop
rate-limit input access-group 103 200000 25000 25000 conform-action transmit exceed-action
drop
```

The CAR configuration refers to IP ACLs in order to classify the traffic, using three different IP ACLs in this case. ACL 101 matches all web traffic; ACL 102 matches all FTP traffic; and ACL 103 matches all VoIP traffic.

Under subinterface s1/0.1, four **rate-limit** commands are used. The first sets the rate for all traffic, dropping traffic that exceeds 496 kbps. However, the conform action is "continue." This means that packets conforming to this statement will be compared to the next **rate-limit** statements, and when matching a statement, some other action will be taken. For instance, web traffic matches the second **rate-limit** command, with a resulting action of either transmit or drop. VoIP traffic would be compared with the next three **rate-limit** commands before matching the last one. As a result, all traffic is limited to 496 kbps, and three particular subsets of traffic are prevented from taking all the bandwidth.

CB Policing can achieve the same effect of policing subsets of traffic by using nested policy maps.

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most Cisco Press *Exam Certification Guides*, this book does not repeat information listed in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 16-9 lists commands related to CB Shaping.

Table 16-9 *Class-Based Shaping Command Reference*

Command	Mode and Function
shape [**average** | **peak**] *mean-rate* [[*burst-size*] [*excess-burst-size*]]	Class configuration mode; enables shaping for the class
shape [**average** | **peak**] **percent** *percent* [[*burst- size*] [*excess-burst-size*]]	Enables shaping based on percentage of bandwidth
Shape adaptive *min-rate*	Enables the minimum rate for adaptive shaping
Shape fecn-adapt	Causes reflection of BECN bits after receipt of an FECN
service-policy {**input** | **output**} *policy- map-name*	Interface or subinterface configuration mode; enables CB Shaping on the interface
shape max-buffers *number-of-buffers*	Sets the maximum queue length for the default FIFO shaping queue
show policy-map *policy-map-name*	Lists configuration information about all MQC-based QoS tools
show policy-map *interface-spec* [**input** | **output**] [**class** *class-name*]	Lists statistical information about the behavior of all MQC-based QoS tools

Table 16-10 lists commands related to FRTS.

Table 16-10 *FRTS Command Reference*

Command	Mode and Function
frame-relay traffic-shaping	Interface subcommand; enables FRTS on the interface
class *name*	Used under the **interface-dlci** to point to a map class
frame-relay class *name*	Used under an interface or subinterface to point to a map class
map-class frame-relay *map-class-name*	Global command to name map class, with subcommands detailing a set of shaping parameters
service-policy output *policy-map-name*	Used in a map class to enable LLQ or CBWFQ
frame-relay traffic-rate ***average*** [*peak*]	Used in a map class to define shaping rates
frame-relay bc out *bits*	Used in a map class to explicitly set Bc
frame-relay be out *bits*	Used in a map class to explicitly set Be
frame-relay cir out *bps*	Used in a map class to explicitly set CIR
frame-relay adaptive-shaping {**becn** | **foresight**}	Used in a map class to both enable adaptive shaping and define what causes FRTS to slow down
frame-relay mincir out *bps*	Used in a map class to define how far adaptive shaping will lower the rate
frame-relay tc *milliseconds*	Used in a map class to explicitly set Tc
frame-relay qos-autosense	Interface command telling the router to use ELMI to discover the CIR, Bc, and Be from the switch
show frame-relay pvc [**interface** *interface*] [*dlci*]	Shows PVC statistics, including shaping statistics
show traffic-shape [*interface-type interface-number*]	Shows information about FRTS configuration per VC
show traffic-shape queue [*interface-number* [**dlci** *dlci-number*]]	Shows information about the queuing tool used with the shaping queue
show traffic-shape statistics [*interface-type interface-number*]	Shows traffic-shaping statistics

Table 16-11 provides a command reference for CB Policing.

Table 16-11 *Class-Based Policing Command Reference*

Command	Mode and Function
police *bps burst-normal burst-max* **conform-action** *action* **exceed-action** *action* [**violate-action** *action*]	**policy-map** class subcommand; enables policing for the class
police cir percent *percent* [**bc** *conform-burst-in-msec*] [**pir percent** *percent*] [**be** *peak-burst-in-msec*] [**conform-action** *action* [**exceed-action** *action* [**violate-action** *action*]]]	**policy-map** class subcommand; enables policing using percentages of bandwidth
police {**cir** *cir*} [**bc** *conform-burst*] {**pir** *pir*} [**be** *peak-burst*] [**conform-action** *action* [**exceed-action** *action* [**violate-action** *action*]]]	**policy-map** class subcommand; enables dual-rate policing
service-policy {**input** | **output**} *policy-map-name*	Enables CB Policing on an interface or subinterface

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

Tc, Bc, Be, CIR, shaping rate, policing rate, token bucket, Bc bucket, Be bucket, adaptive shaping, BECN, ForeSight, ELMI, mincir, map class, marking down, single-rate two-color policer, single-rate three-color policer, dual-rate three-color

policer, conform, exceed, violate, traffic contract, dual token bucket, PIR, nested policy maps, multi-action policing

Refer to the CD-based glossary to check your answers.

Further Reading

Cisco QoS Exam Certification Guide, by Wendell Odom and Michael Cavanaugh

Part V: WAN

Chapter 17 Synchronous Serial Links and Protocols

Chapter 18 Frame Relay

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- WAN
  - — Physical Layer
  - — Leased Line Protocols

CHAPTER 17

Synchronous Serial Links and Protocols

This chapter covers the Layer 1 and 2 standards and protocols related to point-to-point serial links. The chapter begins with coverage of T1/E1 technology, followed by detailed coverage of the link layer Point-to-Point Protocol (PPP).

"Do I Know This Already?" Quiz

Table 17-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 17-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Synchronous Serial Links	1–3	
Point-to-Point Protocol	4–7	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. Which of the following is true regarding T1 links in North America?

a. They typically use either Superframe or B8ZS framing.

b. They support 23 DS0 channels plus 72 kbps of framing and OAM overhead.

c. The bits used for OAM are gathered from bits stolen from inside each DS0 channel.

d. Both popular encoding standards use Bipolar Violations to ensure enough signal transitions to maintain synchronization.

e. None of the answers is correct.

2. Which of the follow pin leads are raised or lowered by an external CSU/DSU that is connected to a router?
   a. DCD
   b. CTS
   c. RTS
   d. DSR
   e. DTR

3. Which of the following are true about Bipolar Violations?
   a. BPVs occur when AMI encoding sends more than 60 percent binary 1s over a time period.
   b. With AMI, BPVs always indicate at least one miscoded bit, implying that the frame will be received in error.
   c. B8ZS uses BPVs to correctly transmit some of the 0s in a string of eight consecutive 0s.
   d. BPVs cannot occur when the encoded binary is 1111 . . ., but cannot occur when the binary is 101010. . . .

4. Imagine that a PPP link failed and has just recovered. Which of the following features is negotiated last?
   a. CHAP authentication
   b. RTP header compression
   c. Looped link detection
   d. Link Quality Monitoring

5. Interfaces s0/0, s0/1, and s1/0 are up and working as part of a multilink PPP bundle that connects to another router. The multilink interface has a bandwidth setting of 1536. When a 1500-byte packet is routed out the multilink interface, which of the following determines out which link the packet will flow?
   a. The current CEF FIB and CEF load-balancing method.
   b. The current fast-switching cache.
   c. The packet is sent out one interface based on round-robin scheduling.
   d. One fragment is sent over each of the three links.

6. R1 and R2 connect over a leased line, with each interface using its s0/1 interface. When configuring CHAP to use a locally defined name and password, which of the following statements is false about the commands and configuration mode in which they are configured?

   a. The **encapsulation ppp** interface subcommand

   b. The **ppp authentication chap** interface subcommand

   c. The **username** *R1* **password** *samepassword* global command on R1

   d. The **username** *R2* **password** *samepassword* interface subcommand on R2

7. Which of the following types of payload compression is supported on HDLC links?

   a. Lempel-Ziv Stacker

   b. MPPC

   c. Predictor

   d. FRF.9

Foundation Topics

Synchronous Serial Links

The original digital circuits created by telcos send a synchronized serial digital signal over copper media, using a twisted pair for each direction of transmission. These original circuits used a 64-kbps basic unit of transmission called *Digital Signal Level 0 (DS0)*. The telcos chose the DS0 speed to support the *pulse code modulation (PCM)* voice codec developed by AT&T, which used 8000 voice samples per second, with 1 byte encoded per sample, to digitize a single voice call.

The original digital WAN switches also used *time-division multiplexing (TDM)* to combine multiple DS0s into faster links. Although they all used the DS0 as the basic unit, telcos in different parts of the world chose different methods of combining DS0s into faster links using TDM. The most popular standards were the T1 carrier (originated in North America), J1 carrier (originated in Japan), and E1 (originated in Europe).

T1 Framing and Encoding

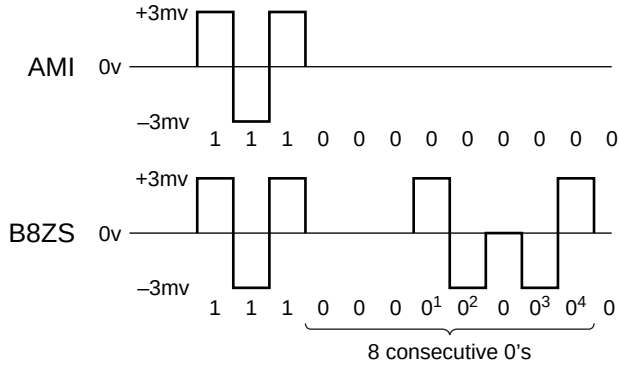
The CSU/DSUs on each end of a T1 link apply *framing* logic to the serial data stream in each direction on the link. The framing process allows a device to identify the individual DS0 channels, as well as overhead bits. The overhead bits are used for synchronization and framing, management, and a CRC error check. T1s use either the older Superframe (SF, also known as D4) framing standard or the Extended Superframe (ESF) frame format. E1 lines use either the ITU G.704 or G.706 standard.

Using a T1 line, the telco could install a T1 circuit to the customer facility, and then via configuration decide how much bandwidth to provide to the customer as a multiple of the 64-kbps DS0 channel. With a physical T1 installed, the CSU/DSU on each end could find the individual DS0 channels. If configured to use only specific DS0 channels, the link would be a fractional or subrate T1. Alternatively, if the full T1 bandwidth is needed, the CSU/DSU can ignore the DS0 channel boundaries, treating all non-overhead bits as a single serial data stream.

The devices attached to the T1 line must conform to the same framing standard and the same line-coding standard. The *line coding* defines the electrical characteristics that imply a 0 or 1 on the link. Figure 17-1 shows a comparison of the two popular encoding standards used with T1s, namely *Bipolar 8 Zero Substitution (B8ZS)* and *Alternate Mark Inversion (AMI)*. AMI and B8ZS use similar rules for encoding 1s and 0s. Both standards use a 3-millivolt (3mV) signal to encode a binary 1, with each binary 1 alternating the *polarity* (current) to avoid inducing a DC component. For binary 0, no voltage is used.

Figure 17-1 *Comparing AMI and B8ZS*

KEY POINT



[1]Intentional Bipolar Violation (4th bit time of the 8 binary 0s)
[2]Normal valid signal for binary 1 (5th bit)
[3]Intentional Bipolar Violation (7th bit time of the 8 binary 0s)
[4]Normal valid signal for binary 1 (8th bit)

B8ZS differs from AMI when eight consecutive 0s occur in the data stream. Long sequences of 0s, meaning 0 V on the line for the duration of the consecutive 0s, result in no signal transitions. Signal transitions are needed to maintain proper clock synchronization. To create signal transitions, B8ZS sends an alternative sequence as shown in Figure 17-1, using *Bipolar Violations (BPVs)*, to represent eight consecutive 0s. (BPVs exist when a 3-mV signal has the same polarity as the previous 3-mV signal; Figure 17-1 points out two BPVs.) Instead of 8-bit times with no signal, B8ZS sends what looks like three 0s, one BPV, a valid 1, a 0, another BPV, and then another valid 1. However, the receiver then converts this sequence as a string of eight 0s, because it knows that this sequence has occurred because of the B8ZS line code rules.

Because B8ZS uses BPVs to correctly encode bits, not all BPVs result in an actual bit error. However, AMI does not use the BPVs to encode eight consecutive 0s, so BPVs would always indicate some incorrect line encoding—that is, a bit error.

Table 17-2 summarizes the key features of DS1 standards.

Table 17-2 *T1 and E1 Compared*

KEY POINT

Feature	T1	E1
DS1 line rate	1.544 Mbps	2.048 Mbps
Subscriber DS0 channels	24	30
Overhead	8 kbps	128 kbps (1 DS0 overhead, 1 DS0 reserved for signaling)
Framing options	SF or ESF	ITU G.704 and G.706

continues

Table 17-2 *T1 and E1 Compared (Continued)*

Feature	T1	E1
Encoding options	B8ZS or AMI	High Density Binary 3 (HDB3)
DS3 comments	T3; 44.736 Mbps; muxes 28 T1s	E3; 34.368 Mbps; muxes 16 E1s

T1 Alarms

Like the other DS levels in the TDM hierarchy, T1 and E1 define some overhead bits for the purpose of *Operation, Administration, and Maintenance (OAM)*. For example, ESF framing devotes 4 kbps of the 8 kbps of T1 overhead to OAM. The telco gear and the CSU/DSU use the bits in the OAM data link to communicate status about the leased circuit—for instance, when an error condition occurs, the CSU/DSU or T1 mux sets bits inside the 4-kbps OAM data link to define the type of error. Table 17-3 lists some of the more common errors signaled for T1 circuits. (Refer to RFC 3895 for a more complete reference.)

Table 17-3 *Popular T1 Alarms*

KEY POINT

Alarm	Meaning
Out of Frame (OOF)/Loss of Frame (LOF)	The receiver can no longer consistently identify the frame boundaries.
Loss of Signal (LOS)	Absence of any received pulses of either polarity for a defined time period.
Alarm Indication Signal (AIS)	Practice of sending all binary 1s on the line in reaction to framing problems, to provide signal transitions and allow recovery of synchronization and framing.
Red Alarm	An alarm state that occurs on a device on one end of the link when that device has detected a LOF/LOS/AIS condition. The device in Red alarm state then sends a Yellow alarm signal to the other end of the link.
Yellow Alarm	An alarm state that occurs when the device on the other end of the link has experienced a red alarm. The device in a Yellow alarm state has received a Yellow alarm signal from the far end of the link.

Carrier Detect and Interface Resets

A router has visibility into T1 framing, along with the alarm conditions, but only if the serial card has a built-in CSU/DSU. However, with an external CSU/DSU, the router has no insight into the framing or encoding on the circuit. Instead, the router can sense the voltages on the various pins on the serial cable. Table 17-4 lists and describes the purpose of the five most important signaling pins in serial cables. (Note that in this model, the term *data communications equipment [DCE]* refers to the local CSU/DSU, and *data terminal equipment [DTE]* refers to the router's serial interface.)

Table 17-4 *Common Serial Cable Control Pin Leads*

Pin	Purpose
Data Carrier Detect (DCD)	Set by the DCE to imply a working link
Data Set Ready (DSR)	Set by the DCE to imply that the DCE is ready to signal using pin leads
Data Terminal Ready (DTR)	Set by the DTE to imply that the DTE is ready to signal using pin leads
Ready To Send (RTS)	Set by the DTE to tell the DCE that the DTE wants to send data
Clear To Send (CTS)	Set by the DCE to tell the DTE that the DTE is allowed to send data

The **show interfaces** command lists the pin lead settings, along with some related statistics. Of particular note, this command lists counters labeled *interface resets* and *carrier transitions*. The router resets an interface for many reasons, including such varied reasons as loss of the DCD signal, severe interface congestion, and simply to re-drive data-link establishment.

The carrier transitions counter increments each time the DCD lead changes from one state to the other. The CSU/DSU changes DCD based on the health of the circuit—for instance, lowering DCD when LOS/LOF occurs, and raising DCD when the CSU/DSU recovers the signal and framing. As a result, the carrier transitions counter increments by two each time the link completes a cycle of failing and then recovering. Example 17-1 shows **show interfaces** output on a router (R3) when the router on the other end of the link was looped back, causing frequent state changes for the DCD pin, which in turn creates multiple interface resets.

Example 17-1 **show interfaces** *Command Output*

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
R3# sh int s 0/1/0
! lines omitted for brevity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     1219 packets output, 17960 bytes, 0 underruns
     0 output errors, 0 collisions, 161 interface resets
     0 output buffer failures, 0 output buffers swapped out
     324 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

Point-to-Point Protocol

The two most popular Layer 2 protocols used on point-to-point links are *High-Level Data Link Control (HDLC)* and *Point-to-Point Protocol (PPP)*. The ISO standard for the much older HDLC does not include a Type field, so the Cisco HDLC implementation adds a Cisco-proprietary 2-byte Type field to support multiple protocols over an HDLC link. PPP includes an architected Protocol field, plus a long list of rich features. Table 17-5 points out some of the key comparison points of these two protocols.

Table 17-5 *HDLC and PPP Comparisons*

KEY POINT

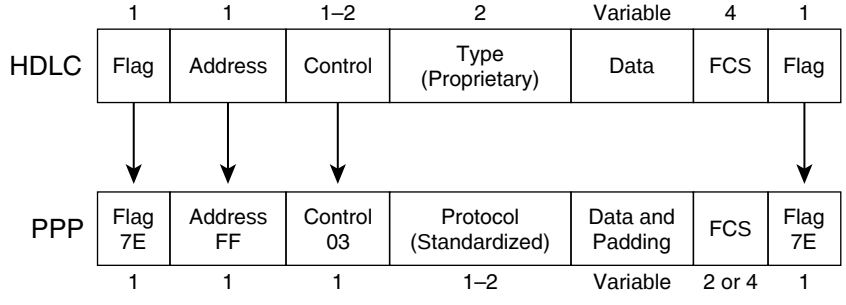
Feature	HDLC	PPP
Error detection?	Yes	Yes
Error recovery?	No	Yes[1]
Standard Protocol Type field?	No	Yes
Default on IOS serial links?	Yes	No
Supports synchronous and asynchronous links?	No	Yes

[1]Cisco IOS defaults to not use the reliable PPP feature, which allows PPP to perform error recovery.

PPP framing (RFC 1662) defines the use of a simple HDLC header and trailer for most parts of the PPP framing, as shown in Figure 17-2. PPP simply adds the Protocol field and optional Padding field to the original HDLC framing. (The Padding field allows PPP to ensure that the frame has an even number of bytes.)

Figure 17-2 *HDLC and PPP Framing Compared*

KEY POINT



PPP Link Control Protocol

PPP standards can be separated into two broad categories—those features unrelated to any specific Layer 3 protocol, and those specific to a Layer 3 protocol. The PPP *Link Control Protocol (LCP)* controls the features independent of any specific Layer 3 protocol. For each Layer 3 protocol supported by PPP, PPP defines a *Network Control Protocol (NCP)*. For instance, the PPP IPCP protocol defines PPP features for IP, such as dynamic address assignment.

When a PPP serial link first comes up—for example, when a router senses the CTS, DSR, and DCD leads come up at the physical layer—LCP begins parameter negotiation with the other end of the link. For example, LCP controls the negotiation of which authentication methods to attempt, and in what order, and then allows the authentication protocol (for example, CHAP) to complete its work. Once all LCP negotiation has completed successfully, LCP is considered to be "up." At that point, PPP begins each Layer 3 Control Protocol.

Table 17-6 lists and briefly describes some of the key features of LCP. Following that, several of the key LCP features are covered in more detail.

Table 17-6 *PPP LCP Features*

KEY POINT

Function	Description
Link Quality Monitoring (LQM)	LCP exchanges statistics about the percentage of frames received without any errors; if the percentage falls below a configured value, the link is dropped.
Looped link detection	Each router generates and sends a randomly chosen magic number. If a router receives its own magic number, the link is looped, and may be taken down.
Layer 2 load balancing	Multilink PPP (MLP) balances traffic by fragmenting each frame into one fragment per link, and sending one fragment over each link.
Authentication	Supports CHAP and PAP.

Basic LCP/PPP Configuration

PPP can be configured with a minimal number of commands, requiring only an **encapsulation ppp** command on each router on opposite ends of the link. Example 17-2 shows a simple configuration with basic PPP encapsulation, plus the optional LQM and CHAP authentication features. For this configuration, routers R3 and R4 connect to each other's s0/1/0 interfaces.

Example 17-2 *PPP Configuration with LQM and CHAP*

```
! R3 configuration is first. The username/password could be held in a AAA
! server, but is shown here as a local username/password. The other router (R4)
! sends its name "R4" with R3 being configured with that username and password setting.
username R4 password 0 rom838
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! The LQM percentage is set with the ppp quality command. CHAP simply needs to be
! enabled, with this router reacting to the other router's host name as stated in
! the CHAP messages.
interface Serial0/1/0
 ip address 10.1.34.3 255.255.255.0
 encapsulation ppp
 ppp quality 80
 ppp authentication chap
```

```
! R4 configuration is next. The configuration is mostly a mirror image of R3.
username R3 password 0 rom838
!
interface Serial0/1/0
 ip address 10.1.34.4 255.255.255.0
 encapsulation ppp
 ppp quality 70
 ppp authentication chap
```

continues

Example 17-2 *PPP Configuration with LQM and CHAP (Continued)*

```
! Next, on R3, the show command lists the phrase "LCP Open," implying that LCP has
! completed negotiations. On the next line, two NCPs (CDPCP and IPCP) are listed.
R3# show int s 0/1/0
Serial0/1/0 is up, line protocol is up
  Hardware is GT96K Serial
  Internet address is 10.1.34.3/24
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation PPP, LCP Open
  Open: CDPCP, IPCP, loopback not set
  Keepalive set (10 sec)
! (The following debug output has been shortened in several places.) The link was
! shut/no shut after issuing the debug ppp negotiation command. The first messages
! state a configuration request, listing CHAP for authentication, that LQM should
! be used, and with the (default) setting of using magic numbers to detect loops.
*Apr 11 14:48:14.795: Se0/1/0 PPP: Phase is ESTABLISHING, Active Open
*Apr 11 14:48:14.795: Se0/1/0 LCP: O CONFREQ [Closed] id 186 len 23
*Apr 11 14:48:14.795: Se0/1/0 LCP:    AuthProto CHAP (0x0305C22305)
*Apr 11 14:48:14.795: Se0/1/0 LCP:    QualityType 0xC025 period 1000 (0x0408C025000003E8)
*Apr 11 14:48:14.795: Se0/1/0 LCP:    MagicNumber 0x13403093 (0x050613403093)
*Apr 11 14:48:14.807: Se0/1/0 LCP: State is Open
! LCP completes, with authentication occurring next. In succession below, the
! challenge is issued in both directions ("O" means "output," "I" means "Input").
! Following that, the response is made, with the hashed value. Finally, the
! confirmation is sent ("success"). Note that by default the process occurs in
! both directions.
*Apr 11 14:48:14.807: Se0/1/0 PPP: Phase is AUTHENTICATING, by both
*Apr 11 14:48:14.807: Se0/1/0 CHAP: O CHALLENGE id 85 len 23 from "R3"
*Apr 11 14:48:14.811: Se0/1/0 CHAP: I CHALLENGE id 41 len 23 from "R4"
*Apr 11 14:48:14.811: Se0/1/0 CHAP: Using hostname from unknown source
*Apr 11 14:48:14.811: Se0/1/0 CHAP: Using password from AAA
*Apr 11 14:48:14.811: Se0/1/0 CHAP: O RESPONSE id 41 len 23 from "R3"
*Apr 11 14:48:14.815: Se0/1/0 CHAP: I RESPONSE id 85 len 23 from "R4"
*Apr 11 14:48:14.819: Se0/1/0 CHAP: O SUCCESS id 85 len 4
*Apr 11 14:48:14.823: Se0/1/0 CHAP: I SUCCESS id 41 len 4
*Apr 11 14:48:14.823: Se0/1/0 PPP: Phase is UP
```

Multilink PPP

Multilink PPP, abbreviated as MLP, MP, or MLPPP, defines a method to combine multiple parallel serial links at Layer 2. The original motivation for MLP was to combine multiple ISDN B-channels without requiring any Layer 3 load balancing; however, MLP can be used to load balance traffic across any type of point-to-point serial link.

MLP balances traffic by fragmenting each packet based on the number of parallel links and then sending one fragment over each link. For each packet, with three parallel links, MLP fragments

each packet into three fragments. To allow reassembly on the receiving end, MLP adds a header (either 4 or 2 bytes) to each fragment. The header includes a Sequence Number field as well as Flag bits designating the beginning and ending fragments.

MLP can be configured using either multilink interfaces or virtual templates. Example 17-3 shows an MLP multilink interface with two underlying serial interfaces. Following the configuration, the example shows some interface statistics that result from a **ping** from one router (R4) to the other router (R3) across the MLP connection.

Example 17-3 *MLP Configuration and Statistics with Multilink Interfaces-R3*

KEY POINT

```
! All Layer 3 parameters are configured on the multilink interface. The
! serial links are associated with the multilink interface using the ppp
! multilink group commands.
interface Multilink1
 ip address 10.1.34.3 255.255.255.0
 encapsulation ppp
 ppp multilink
 ppp multilink group 1
!
interface Serial0/1/0
 no ip address
 encapsulation ppp
 ppp multilink group 1
!
interface Serial0/1/1
 no ip address
 encapsulation ppp
 ppp multilink group 1
! Below, the interface statistics reflect that each of the two serial links sends
! the same number of packets, one fragment of each original packet. Note that the
! multilink interface shows roughly the same number of packets, but the bit rate
! matches the sum of the bit rates on the two serial interfaces. These stats
! reflect the fact that the multilink interface shows prefragmentation
! counters, and the serial links show post-fragmentation counters.
R3# sh int s 0/1/0
Serial0/1/0 is up, line protocol is up
! lines omitted for brevity
  5 minute input rate 182000 bits/sec, 38 packets/sec
  5 minute output rate 182000 bits/sec, 38 packets/sec
     8979 packets input, 6804152 bytes, 0 no buffer
     8977 packets output, 6803230 bytes, 0 underruns
R3# sh int s 0/1/1
Serial0/1/1 is up, line protocol is up
! lines omitted for brevity
  5 minute input rate 183000 bits/sec, 38 packets/sec
  5 minute output rate 183000 bits/sec, 38 packets/sec
```

continues

Example 17-3 *MLP Configuration and Statistics with Multilink Interfaces-R3 (Continued)*

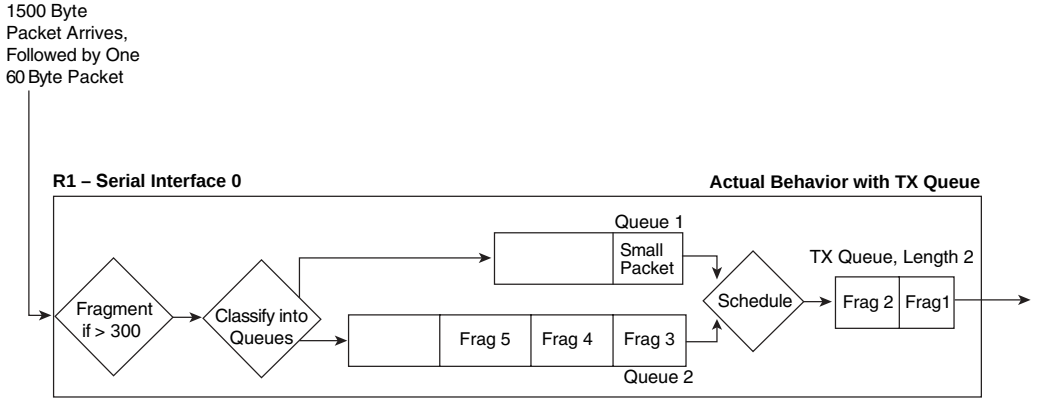
```
     9214 packets input, 7000706 bytes, 0 no buffer
     9213 packets output, 7000541 bytes, 0 underruns
R3# sh int multilink1
Multilink1 is up, line protocol is up
! lines omitted for brevity
  Hardware is multilink group interface
  Internet address is 10.1.34.3/24
  MTU 1500 bytes, BW 3088 Kbit, DLY 100000 usec,
     reliability 255/255, txload 31/255, rxload 30/255
  Encapsulation PPP, LCP Open, multilink Open
  Open: CDPCP, IPCP, loopback not set
  5 minute input rate 374000 bits/sec, 40 packets/sec
  5 minute output rate 377000 bits/sec, 40 packets/sec
     9385 packets input, 14112662 bytes, 0 no buffer
     9384 packets output, 14243723 bytes, 0 underruns
```

MLP Link Fragmentation and Interleaving

The term *Link Fragmentation and Interleaving (LFI)* refers to a type of Cisco IOS QoS tool that prevents small, delay-sensitive packets from having to wait on longer, delay-insensitive packets to be completely serialized out an interface. To do so, LFI tools fragment larger packets, and then send the delay-sensitive packet after just a portion of the original, longer packet. The key elements include fragmentation, the ability to interleave parts of one packet between fragments of another packet, and a queuing scheduler that interleaves the packets. Figure 17-3 depicts the complete process. A 1500-byte packet is fragmented and a 60-byte packet is interleaved between the fragments by the queuing scheduler after the first two fragments.

Figure 17-3 *MLP LFI Concept*

KEY POINT



MLP supports LFI, the key elements of which are detailed in the following list:

- The **ppp multilink interleave** interface subcommand tells the router to allow interleaving.

- The **ppp multilink fragment-delay *x*** command defines the fragment size, based on the following formula:

  size = *x* * bandwidth.

- MLP LFI can be used with only one link or with multiple links.
- The queuing scheduler on the multilink interface determines the next packet to send; as a result, many implementations use LLQ to always interleave delay-sensitive traffic between fragments.

Example 17-4 shows an updated version of the configuration in Example 17-3, with LFI enabled.

Example 17-4 *MLP LFI with LLQ to Interleave Voice*

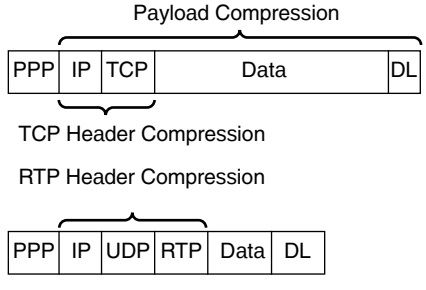
```
! The fragment delay is set to 10 ms, so the fragments will be of size (256,000 *
! .01 second) = 2560 bits = 320 bytes. The ppp multilink interleave command allows
! the queuing tool to interleave packets between fragments of other packets, and
! the referenced policy map happens to use LLQ to interleave voice packets.
interface Multilink1
 bandwidth 256
 ip address 10.1.34.3 255.255.255.0
 encapsulation ppp
 ppp multilink
 ppp multilink group 1
 ppp multilink fragment-delay 10
 ppp multilink interleave
 service-policy output queue-on-dscp
```

PPP Compression

PPP can negotiate to use Layer 2 payload compression, TCP header compression, and/or RTP header compression. Each type of compression has pros and cons, with the most obvious relating to what is compressed, as shown in Figure 17-4.

Figure 17-4 *Fields Compressed with Compression Features*

KEY POINT



Comparing payload compression and header compression, payload compression works best with longer packet lengths, and header compression with shorter packet lengths. Header compression takes advantage of the predictability of headers, achieving a compression ratio for the header fields around 10:1 to 20:1. However, when the data inside the packet is much larger than the header, saving some bytes with header compression may be only a small reduction in the overall bandwidth required, making payload compression more appealing.

PPP Layer 2 Payload Compression

Cisco IOS software supplies three different payload compression options for PPP, namely Lempel-Ziv Stacker (LZS), Microsoft Point-to-Point Compression (MPPC), and Predictor. Stacker and MPPC both use the same underlying Lempel-Ziv (LZ) compression algorithm, with Predictor using an algorithm called Predictor. LZ uses more CPU and less memory in comparison to Predictor, and it typically results in a better compression ratio.

Table 17-7 summarizes some of the key points regarding payload compression. Note that of the three options, only LZS is supported on Frame Relay and HDLC links. Also note that for payload compression when using ATM-to-Frame Relay Service Interworking, MLP must be used; as a result, all payload compression types supported by PPP are also supported for Interworking.

Table 17-7 *Point-to-Point Payload Compression Tools: Feature Comparison*

KEY POINT

Feature	Stacker	MPPC	Predictor
Uses LZ algorithm?	Yes	Yes	No
Uses Predictor algorithm?	No	No	Yes
Supported on HDLC?	Yes	No	No
Supported on PPP?	Yes	Yes	Yes
Supported on Frame Relay?	Yes	No	No
Supports ATM and ATM-to-Frame Relay Service Interworking (using MLP)?	Yes	Yes	Yes

Configuring payload compression simply requires a matching **compress** command under each interface on each end of the link(s), with matching parameters for the type of compression. Once compression is configured, PPP starts the Compression Control Protocol (CCP), which is another NCP, to perform the compression negotiations and manage the compression process.

Header Compression

PPP supports two styles of IP header compression: TCP header compression and RTP header compression. (Figure 17-4 shows the headers compressed by each.)

Voice and video flows use the RTP encapsulation shown in Figure 17-4. Voice flows, particularly for low-bitrate codecs, have very small data fields—for instance, with G.729, the packet is typically 60 bytes, with 40 bytes of the 60 bytes being the IP/UDP/RTP headers. RTP header compression compresses the IP/UDP/RTP headers (40 bytes) into 2 or 4 bytes. With G.729 in use, RTP header compression reduces the required bandwidth by more than 50 percent.

TCP header compression compresses the combined IP and TCP headers, a combined 40 bytes, into 3 or 5 bytes. For TCP packets with small payloads, the saving can be significant; the math is similar to the RTP compression example in the previous paragraph. However, TCP header compression might not be worth the CPU and memory expense for larger packets—for instance, for a 1500-byte packet, compressing the 40 bytes of header into 3 bytes reduces the packet size by only about 2 percent.

Header compression can be configured using a pair of legacy commands, or it can be configured using MQC commands. The legacy commands are **ip tcp header-compression** [**passive**] and **ip rtp header-compression** [**passive**], used under the serial (PPP) or multilink (MLP) interfaces on each end of the link. PPP reacts to this command by using IPCP to negotiate to enable each type of compression. (If you use the **passive** keyword, that router waits for the other router to initiate the IPCP negotiation.) With this style of configuration, all TCP flows and/or all RTP flows using the link are compressed.

Example 17-5 shows the alternative method using an MQC policy map to create class-based header compression. In the example, TCP header compression is applied only to the class that holds Telnet traffic. As a result, TCP header compression is applied to the packets that are most likely to benefit from TCP compression, without wasting CPU and memory to compress larger packets. (Recall that Telnet sends one keystroke per TCP segment, unless **service nagle** is configured, making Telnet highly inefficient by default.)

Example 17-5 *MQC Class-Based Header Compression*

```
! RTP compression is enabled in the voice class, TCP header compression in the
! critical data class, and no compression in the class-default class.
policy-map cb-compression
 class voice
  bandwidth 82
  compress header ip rtp
 class critical
  bandwidth 110
  compress header ip tcp
!
interface Multilink1
 bandwidth 256
 service-policy output cb-compression
```

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most Cisco Press *Exam Certification Guides*, this book does not repeat information listed in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 17-8 lists the key protocols covered in this chapter.

Table 17-8 *Protocols and Standards for Chapter 17*

Topic	Standard
Point-to-Point Protocol (PPP)	RFC 1661
PPP in HDLC-like Framing	RFC 1662
PPP Internet Protocol Control Protocol IPCP	RFC 1332
IP Header Compression over PPP	RFC 3544
PPP Multilink Protocol (MLP)	RFC 1990
Managed Objects for TDM Circuits	RFC 3895

Table 17-9 lists the Cisco IOS commands covered in this chapter.

Table 17-9 *Command Reference for Chapter 17*

Command	Mode and Function
linecode {**ami** | **b8zs** | **hdb3**}	T1/E1 controller mode; defines line coding for the line
framing {**sf** | **esf**} **framing** {**crc4** | **no-crc4**} [**australia**]	T1/E1 controller mode; defines framing
ppp authentication {*protocol1* [*protocol2...*]} [**if-needed**] [*list-name* | **default**] [**callin**] [**one-time**] [**optional**]	Interface mode; defines the authentication protocol (PAP, CHAP, EAP) and other parameters
ppp multilink [**bap**]	Interface mode; enables MLP on an interface
ppp multilink fragment-delay *delay-max*	Interface mode; defines the fragment size based on this delay and interface bandwidth
ppp multilink group *group-number*	Interface mode; associates a physical interface to a multilink interface

Table 17-9 *Command Reference for Chapter 17 (Continued)*

Command	Mode and Function
ppp multilink interleave	Interface mode; allows the queuing scheduled to interleave packets between fragments of another packet
compress [**predictor** | **stac** | **mppc** [**ignore-pfc**]]	Interface mode; configures payload compression
ip rtp header-compression [**passive**]	Interface mode; enables RTP header compression
ip tcp header-compression [**passive**]	Interface mode; enables TCP header compression
compression header ip [**rtp** | **tcp**]	Class configuration mode; enables RTP or TCP header compression inside an MQC class
ppp quality *percentage*	Interface mode; enables LQM monitoring at the stated percentage
debug ppp negotiation	Enables debugging that shows the various stages of PPP negotiation

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

encoding, line coding, framing, B8ZS, AMI, HDB3, SF, D4 framing, ESF, DS0, DS1, DS3, TDM hierarchy, T1, E1, T3, E3, PCM, TDM, BPV, Red Alarm, Yellow Alarm, OOF, LOS, LOF, AIS, OAM, DCD, DSR, DTR, CTS, RTS, PPP, MLP, LCP, NCP, IPCP, CDPCP, MLP LFI, CHAP, PAP, LFI, Layer 2 payload compression, TCP header compression, RTP header compression

Further Reading

Troubleshooting Remote Access Networks, by Dr. Plamen Nedeltchev

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- WAN
  - — Frame Relay
  - — Physical Layer

CHAPTER 18

Frame Relay

This chapter covers the details of Frame Relay.

"Do I Know This Already?" Quiz

Table 18-1 outlines the major sections in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 18-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Frame Relay Concepts	1–4	
Frame Relay Configuration	5–7	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. Which of the following is true about both the ANSI and ITU options for Frame Relay LMI settings in a Cisco router, but not for the LMI option called **cisco**?

- **a.** They use DLCI 1023 for LMI functions.
- **b.** They use DLCI 0 for LMI functions.
- **c.** They include support for a maximum of 1022 DLCIs on a single access link.
- **d.** They include support for a maximum of 992 DLCIs on a single access link.
- **e.** They can be autosensed by a Cisco router.

2. R1 sends a Frame Relay frame over a PVC to R2. When R2 receives the frame, the frame has the DE and FECN bits set, but not the BECN bit. Which of the following statements accurately describes how R2 could have reacted to this frame, or how R1 might have impacted the contents of the frame?

   a. R2 would lower its shaping rate on the PVC assuming R2 has configured adaptive shaping.

   b. R2 could discard the received frame because of the DE setting.

   c. R2 could set BECN in the next frame it sends to R1, assuming FECN reflection is configured.

   d. R1 could have set the FECN bit before sending the frame if R1 had configured outbound policing with the policer marking FECN for out-of-contract frames.

3. Which of the following statements are true regarding Frame Relay encapsulation?

   a. Encapsulation type *cisco* can be configured using the **encapsulation cisco** interface subcommand.

   b. Encapsulation type *ietf* can be configured using the **encapsulation frame-relay** interface subcommand.

   c. Encapsulation type *cisco* can be configured using the **encapsulation frame-relay cisco** subinterface subcommand.

   d. Encapsulation types must be the same on all VCs on the same physical access link.

   e. Different encapsulations can be configured for each VC on the same physical interface using the **frame-relay interface-dlci** *dlci encapsulation-type* command.

   f. Different encapsulations can be configured for each VC using the **encapsulation frame-relay encapsulation-type** subcommand under the **frame-relay interface-dlci** command.

4. Which of the following commands disables Frame Relay LMI?

   a. The **no frame-relay lmi** command under the physical interface.

   b. The **no keepalive** command under the physical interface.

   c. The **frame-relay lmi-interval 0** command under the physical interface.

   d. The **keepalive 0** command under the physical interface.

   e. It cannot be disabled, as it is required for a working Frame Relay access link.

5. Which of the following answers are true regarding the three options for Frame Relay payload compression?

   a. FRF.9 and packet-by-packet use a per-packet compression dictionary.

   b. Data-stream and FRF.9 compression use the LZS compression algorithm.

   c. The only method of enabling data steam compression is through the **frame-relay payload-compress** subinterface subcommand.

   d. The data-stream compression type is Cisco proprietary.

6. R1 has a Frame Relay access link on s0/0. The attached Frame Relay switch has ten PVCs configured on the link, with DLCIs 80–89. Which of the following is true regarding definition of DLCIs and encapsulation on the link?

   a. The **frame-relay interface-dlci** command associates a DLCI with the subinterface under which it is configured.

   b. The LMI Status message from the switch can be used by the router to associate the DLCIs with the correct subinterface.

   c. PVCs using IETF encapsulation require a **frame-relay map** command on the related subinterface.

   d. The LMI Status message from the switch tells the router which encapsulation to use for each PVC.

   e. Different encapsulation types can be mixed over this same access link.

7. R1 has a Frame Relay access link on s0/0. The attached Frame Relay switch has ten PVCs configured on the link, with DLCIs 80–89. R1's configuration includes ten point-to-point subinterfaces, also numbered 80 through 89, but only four of those subinterfaces list a DLCI using the **frame-relay interface-dlci** command. All ten subinterfaces have IP addresses configured, but no other **frame-relay** commands are configured on the subinterfaces. Which of the following could be true regarding R1's use of Frame Relay?

   a. Six subinterfaces will not be able to send traffic.

   b. Six subinterfaces will learn their associated DLCIs as a result of received Inverse ARP messages.

   c. Six subinterfaces will learn their associated DLCIs as a result of sent Inverse ARP messages.

   d. Four subinterfaces need a **frame-relay map** command before they can successfully pass traffic.

   e. LMI will learn the missing DLCIs and assign them to the subinterface bearing the same value as the DLCI.

Foundation Topics

Frame Relay Concepts

Frame Relay remains the most commonly deployed WAN technology used by routers. A slow migration away from Frame Relay has already begun with the advent and rapid growth of IP-based VPNs and MPLS. However, Frame Relay will likely be a mainstay of enterprise networks for the fore-seeable future.

Frame Relay standards have been developed by many groups. Early on, Cisco and some other companies (called the *gang of four*) developed vendor standards to aid Frame Relay adoption and product development. Later, a vendor consortium called the *Frame Relay Forum (FRF)* formed for the purpose of furthering Frame Relay standards; the IETF concurrently defined several RFCs related to using Frame Relay as a Layer 2 protocol in IP networks. (Cisco IOS documentation frequently refers to FR standards via FRF Implementation Agreements [IAs]—for instance, the FRF.12 fragmentation specification.) Finally, ANSI and ITU built on those standards to finalize U.S. national and international standards for Frame Relay.

This section briefly covers some of the more commonly known features of Frame Relay, as well as specific examples of some of the less commonly known features. This section does not attempt to cover all of Frame Relay's core concepts or terms, mainly because most engineers already understand Frame Relay well. So, make sure to review the definitions listed at the end of this chapter to fill in any gaps in your Frame Relay knowledge.

Frame Relay Data Link Connection Identifiers

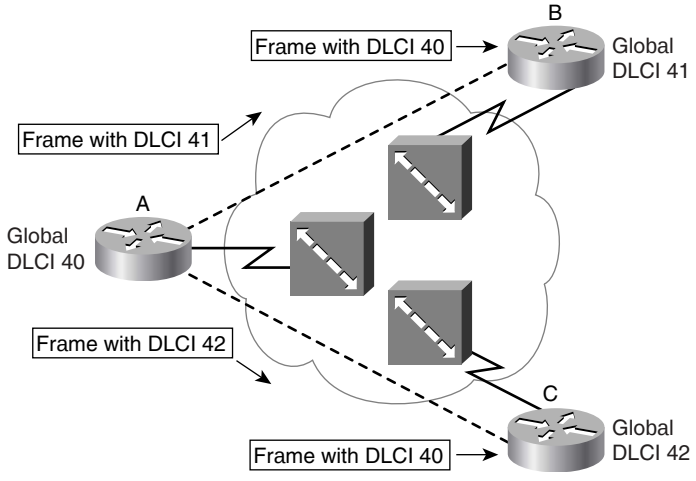
To connect two DTEs, an FR service uses a *virtual circuit (VC)* between pairs of routers. A router can then send an FR frame with the appropriate (typically) 10-bit *Data Link Connection Identifier (DLCI)* header field that identifies each VC. The intermediary FR switches forward the frame based on its DLCI, until the frame eventually exits the FR service out the access link to the router on the other end of the VC.

FR DLCIs are locally significant, meaning that a particular DLCI value only matters on a single link. As a result, the DLCI value for a frame may change as the frame passes through the network. The following five-step process shows the locally significant DLCI values for a VC in Figure 18-1:

1. Router A sends a frame with DLCI 41.
2. The FR service identifies the frame as part of the VC connecting Router A to Router B.
3. The FR service replaces the frame's DLCI field with a value of 40.

4. The FR service forwards the frame to Router B.
5. Router B sees the incoming DLCI as 40, identifying it as being from Router A.

Figure 18-1 *Comparing Local and Global Frame Relay DLCIs*



In practice, some providers use a convention called *global addressing*. The global DLCI convention simply allows humans to think of routers as having a single address, more akin to how MAC addresses are used. However, the addresses are still local, and a VC's DLCI may well change values as it passes through the network. For instance, the same VC from Router A to Router B in Figure 18-1 could use global addressing, listing Router A's DLCI as 40, and Router B's as 41. The logic based on the global addresses works like LANs. For example, for Router A to send a frame to Router B, Router A would send the frame to Router B's global address (41). Similarly, Router B would send frames to Router A's global address of 40 to send packets to Router A.

Local Management Interface

Local Management Interface (LMI) messages manage the local access link between the router and the Frame Relay switch. A Frame Relay DTE can send an LMI *Status Enquiry* message to the switch; the switch then replies with an LMI *Status* message to inform the router about the DLCIs of the defined VCs, as well as the status of each VC. By default, the LMI messages flow every 10 seconds. Every sixth message carries a full Status message, which includes more complete status information about each VC.

The LMI Status Enquiry (router) and Status (switch) messages function as a keepalive as well. A router considers its interface to have failed if the router ceases to receive LMI messages from the switch for a number (default 3) of keepalive intervals (default 10 seconds). As a result, FR LMI is actually enabled/disabled by using the **keepalive**/**no keepalive** interface subcommands on a Frame Relay interface.

Three LMI types exist, mainly because various vendors and standards organizations worked independently to develop Frame Relay standards. The earliest-defined type, called the Cisco LMI type, differs slightly from the later-defined ANSI and ITU types, as follows:

- The allowed DLCI values
- The DLCI used for sending LMI messages

Practically speaking, these issues seldom matter; by default, routers autosense the LMI type. If needed, the **frame-relay lmi-type type** interface subcommand can be used to set the LMI type on the access link. Table 18-2 lists the three LMI types, the **type** keyword values, along with some comparison points regarding LMI and permitted DLCIs.

Table 18-2 *Frame Relay LMI Types*

KEY POINT

LMI Type	Source Document	Cisco IOS lmi-type Parameter	Allowed DLCI Range (Number)	LMI DLCI
Cisco	Proprietary	**Cisco**	16–1007 (992)	1023
ANSI	T1.617 Annex D	**Ansi**	16–991 (976)	0
ITU	Q.933 Annex A	**q933a**	16–991 (976)	0

Frame Relay Headers and Encapsulation

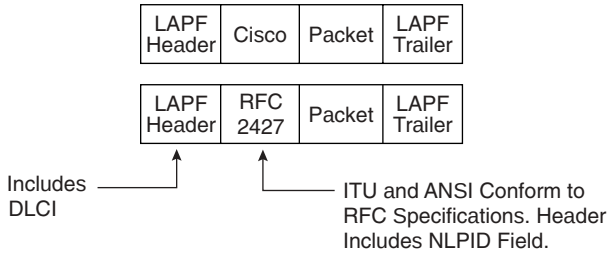
Routers create Frame Relay frames by using different consecutive headers. The first header is the ITU *Link Access Procedure for Frame-Mode Bearer Services (LAPF)* header. The LAPF header includes all the fields used by Frame Relay switches to deliver frames across the FR cloud, including the DLCI, DE, BECN, and FECN fields.

The Frame Relay encapsulation header follows the LAPF header, holding fields that are important only to the DTEs on the ends of a VC. For the encapsulation header, two options exist:

- The earlier-defined Cisco-proprietary header
- The IETF-defined RFC 2427 (formerly RFC 1490) encapsulation header

The **cisco** option works well with Cisco routers on each end of the VC, with the **ietf** option being required for multivendor interoperability. Both headers include a Protocol Type field to support multiple Layer 3 protocols over a VC; the most commonly used is the RFC 2427 *Network Layer Protocol ID (NLPID)* field. Figure 18-2 shows the general structure of the headers and trailers.

Figure 18-2 *Frame Relay Encapsulation Options*



Each VC uses the Cisco encapsulation header unless configured explicitly to use the IETF header. Three methods can be used to configure a VC to use the IETF-style header:

- Use the **encapsulation frame-relay ietf** interface subcommand, which changes that interface's default for each VC to IETF instead of cisco
- Use the **frame-relay interface-dlci** *number* **ietf** interface subcommand, overriding the default for this VC
- Use the **frame-relay map dlci . . . ietf** command, which also over-rides the default for this VC

For example, on an interface with ten VCs, seven of which need to use IETF encapsulation, the interface default could be changed to IETF using the **encapsulation frame-relay ietf** interface subcommand. Then, the **frame-relay interface-dlci** *number* **cisco** command could be used for each of the three VCs that require Cisco encapsulation.

Frame Relay Congestion: DE, BECN, and FECN

FR networks, like any other multiaccess network, create the possibility for congestion caused by speed mismatches. For instance, imagine an FR network with 20 remote sites with 256-kbps links, and one main site with a T1 link. If all 20 remote sites were to send continuous frames to the main site at the same time, about 5 Mbps of data would need to exit the FR switch over the 1.5-Mbps T1 connected to the main router, causing the output queue on the FR switch to grow. Similarly, when the main site sends data to any one remote site, it sends at T1 speed, potentially causing the egress queue connected to the remote 256-kbps access link to back up as well. Beyond those two cases, which are typically called *egress blocking*, queues can grow inside the core of the FR network as well.

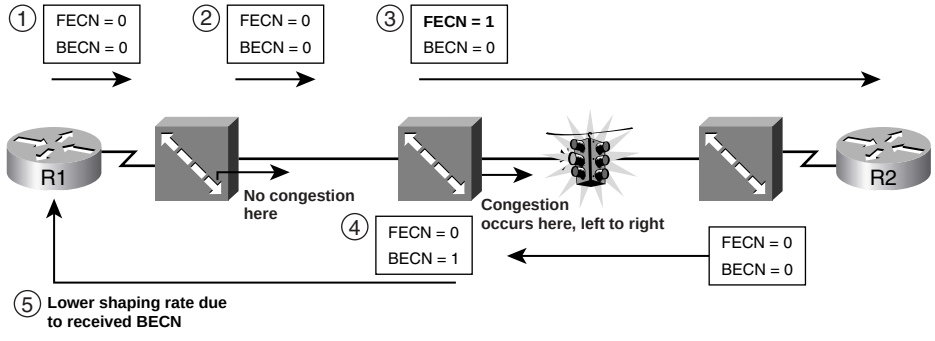
Frame Relay provides two methods of reacting to the inevitable congestion, as covered in the next two sections.

Adaptive Shaping, FECN, and BECN

Chapter 16, "Shaping and Policing," briefly covers the concept of adaptive traffic shaping, in which the shaper varies the shaping rate depending on whether the network is congested or not. To react to congestion that occurs somewhere inside the FR cloud, the router must receive some form of notice that the congestion is occurring. So, the FR LAPF header includes the *Forward Explicit Congestion Notification (FECN)* and *Backward Explicit Congestion Notification (BECN)* bits for signaling congestion on a particular VC.

FR switches use FECN and BECN to inform a router that a particular VC has experienced congestion. To do so, when a switch notices congestion caused by a VC, the switch sets the FECN bit in a frame that is part of that VC. The switch also tracks the VC that was congested so that it can look for the next frame sent over that VC, but going the opposite direction, as shown in step 4 of Figure 18-3. The switch then marks the BECN bit in that frame. The router receiving the frame with BECN set knows that a frame it sent experienced congestion, so the router can reduce its shaping rate. Figure 18-3 shows an example of the process.

Figure 18-3 *Basic Operation of FECN and BECN*



FECN can be set by the FR switches, but not by any of the routers, because the routers do not need to signal forward congestion. For example, if R1 thought congestion was occurring left to right in Figure 18-3, R1 could simply slow down its shaping rate. At the other end of the link, R2 is the destination of the frame, so it would never notice congestion for frames going left to right. So, only the switches need to set FECN.

BECN can be set by switches and by a router. Figure 18-3 shows a switch setting BECN on the next user frame. It can also send a Q.922 test frame, removing the need to wait on traffic sent over the VC, setting BECN in that frame. Finally, routers can be configured to watch for received frames with FECN set, reacting by returning a Q.922 test frame over that VC with the BECN bit set. This feature, sometimes called FECN reflection, is configured with the **shape fecn-adapt** (CB Shaping) or **traffic-shape fecn-adapt** (FRTS) command.

The Discard Eligibility Bit

When congestion occurs, queues begin to fill, and in some cases, frames must be tail-dropped from the queues. Switches can (but are not required to) examine the FR Discard Eligibility (DE) bit when frames need to be discarded, and purposefully discard frames with DE set instead of frames without DE set.

Both routers and switches can set the DE bit. Typically, a router makes the decision about setting the DE bit for certain frames, because the network engineer that controls the router is much more likely to know (and care) about which traffic is more important than other traffic. Marking DE can be performed with CB Marking, as covered in Chapter 14, "Classification and Marking," using the MQC **set fr-de** command.

Although routers typically mark DE, FR switches may also mark DE. For switches, the marking is typically done when the switch polices, but instead of discarding out-of-profile traffic, the switch marks DE. By doing so, downstream switches will be more likely to discard the marked frames that had already caused congestion.

Table 18-3 summarizes some of the key points regarding Frame Relay's FECN, BECN, and DE bits.

Table 18-3 *Frame Relay FECN, BECN, and DE Summary*

KEY POINT

Bit	Meaning When Set	Where Set
FECN	Congestion in the same direction as this frame	By FR switches in user frames
BECN	Congestion in the opposite direction of this frame	By FR switches or routers in user or Q.922 test frames
DE	This frame should be discarded before non-DE frames	By routers or switches in user frames

Frame Relay Configuration

This section completes the FR configuration coverage for this book. Earlier, Chapter 7, "IP Forwarding (Routing)," covered issues with mapping Layer 3 addresses to FR DLCIs, and Chapter 10, "OSPF," covered issues with using OSPF over FR. This section covers the basic configuration and operational commands, along with FR payload compression and FR LFI options.

Frame Relay Configuration Basics

Two of the most important details regarding Frame Relay configuration are the association of DLCIs with the correct interface or subinterface, and the mapping of L3 addresses to those DLCIs. Interesting, both features can be configured using the same two commands—the **frame-relay map** and **frame-relay interface-dlci** commands. Chapter 7 already covered the details of mapping L3 addresses to DLCIs using InARP and static mapping. (If you have not reviewed those

details since starting this chapter, it is probably a good time to do so.) This section focuses more on the association of DLCIs with a particular subinterface.

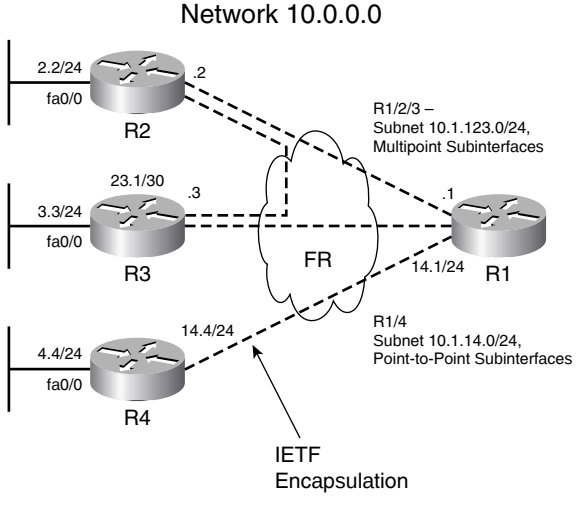
Although a router can learn each DLCI on the access link via LMI Status messages, these messages do not imply with which subinterface each DLCI should be used. To configure Frame Relay using subinterfaces, the DLCIs must be associated with the subinterface. Any DLCIs learned with LMI that are not associated with a subinterface are assumed to be used by the physical interface.

The more common method to make this association is to use the **frame-relay interface-dlci** subinterface subcommand. On point-to-point subinterfaces, only a single **frame-relay interface-dlci** command is allowed, whereas multipoint interfaces support multiple commands. The alternative method is to use the **frame-relay map** command. This command still maps Layer 3 addresses to DLCIs, but also implies an association of the configured DLCI with the subinterface under which the command is issued. And similar to **frame-relay interface-dlci** commands, only one **frame-relay map** command is allowed per point-to-point subinterface, per Layer 3 protocol. On multipoint subinterfaces, multiple commands are allowed per Layer 3 protocol.

Example 18-1 depicts a wide variety of Frame Relay configuration options, using **frame-relay interface-dlci** commands, and the related **show** commands. Based on Figure 18-4, this example implements the following requirements:

- R1 uses a multipoint subinterface to connect to R2 and R3.
- R1 uses a point-to-point subinterface to connect to R4.
- The VC between R1 and R4 uses IETF encapsulation.

Figure 18-4 *Sample FR Network for Configuration Examples*



Example 18-1 *Basic Frame Relay Configuration Example*

```
! R1 configuration begins the example. Subint .14 shows the IETF option used on
! the frame-relay interface-dlci command. Subint .123 has two DLCIs associated
! with it, for the VCs to R2 and R3.
interface Serial0/0/0
 encapsulation frame-relay
!
interface Serial0/0/0.14 point-to-point
 ip address 10.1.14.1 255.255.255.0
 frame-relay interface-dlci 104 IETF
!
interface Serial0/0/0.123 multipoint
 ip address 10.1.123.1 255.255.255.0
 frame-relay interface-dlci 102
 frame-relay interface-dlci 103
```

```
! R2 configuration comes next. R2 assigns the DLCI for the VC to R1 and R3 to the
! .123 subinterface. Note the routers' subint numbers do not have to match.
interface Serial0/0/0
 encapsulation frame-relay
!
interface Serial0/0/0.123 multipoint
 ip address 10.1.123.2 255.255.255.0
 frame-relay interface-dlci 101
 frame-relay interface-dlci 103
! R3 configuration follows the same conventions as does R2's and is not shown.
```

```
! R4's configuration follows next, with the encapsulation frame-relay ietf command
! setting the encapsulation for all the VCs on interface s0/0/0. Also note that
! the frequency of LMI enquiries was changed from the default (10) to 8 with the
! keepalive 8 command.
interface Serial0/0/0
 encapsulation frame-relay IETF
 keepalive 8
!
interface Serial0/0/0.1 point-to-point
 ip address 10.1.14.4 255.255.255.0
 frame-relay interface-dlci 101
```

```
! The show frame-relay pvc command shows statistics and status per VC. The next
! command (on R1) filters the output to just include the lines with PVC status.
R1# show frame-relay pvc ¦ incl PVC STATUS
DLCI = 100, DLCI USAGE = UNUSED, PVC STATUS = INACTIVE, INTERFACE = Serial0/0/0
DLCI = 102, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0.123
DLCI = 103, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0.123
DLCI = 104, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0.14
DLCI = 105, DLCI USAGE = UNUSED, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0
DLCI = 106, DLCI USAGE = UNUSED, PVC STATUS = INACTIVE, INTERFACE = Serial0/0/0
DLCI = 107, DLCI USAGE = UNUSED, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0
DLCI = 108, DLCI USAGE = UNUSED, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0
DLCI = 109, DLCI USAGE = UNUSED, PVC STATUS = INACTIVE, INTERFACE = Serial0/0/0
```

continues

Example 18-1 *Basic Frame Relay Configuration Example (Continued)*

```
DLCI = 110, DLCI USAGE = UNUSED, PVC STATUS = INACTIVE, INTERFACE = Serial0/0/0
! The next command lists stats for a single VC on R1, with DLCI 102, which is the
! VC to R2. Note the counters for FECN, BECN, and DE, as well as the in and out
! bit rates just for this VC.
R1# show frame-relay pvc 102
PVC Statistics for interface Serial0/0/0 (Frame Relay DTE)
DLCI = 102, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0.123

  input pkts 41             output pkts 54           in bytes 4615
  out bytes 5491            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 27         out bcast bytes 1587
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 00:29:37, last time pvc status changed 00:13:47
! The following output confirms that R1's link is using the Cisco LMI standard. Full
! LMI Status messages occur about every minute, with the Last Full Status message
! listed last. Note that the router sends Status Enquiries to the switch, with the
! switch sending Status messages; those counters should increment together.
R1# show frame-relay lmi
LMI Statistics for interface Serial0/0/0 (Frame Relay DTE) LMI TYPE = CISCO
  Invalid Unnumbered info 0         Invalid Prot Disc 0
  Invalid dummy Call Ref 0         Invalid Msg Type 0
  Invalid Status Message 0         Invalid Lock Shift 0
  Invalid Information ID 0         Invalid Report IE Len 0
  Invalid Report Request 0         Invalid Keep IE Len 0
  Num Status Enq. Sent 183         Num Status msgs Rcvd 183
  Num Update Status Rcvd 0         Num Status Timeouts 0
  Last Full Status Req 00:00:35     Last Full Status Rcvd 00:00:35
! The show interface command lists several details as well, including the interval
! for LMI messages (keepalive), LMI stats, LMI DLCI (1023), and stats for the FR
! broadcast queue. The broadcast queue holds FR broadcasts that must be replicated
! and sent over this VC, for example, OSPF LSAs.
R1# show int s 0/0/0
Serial0/0/0 is up, line protocol is up
! lines omitted for brevity
  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 274/0, interface broadcasts 228
! Lines omitted for brevity
! R3 is using ANSI LMI, which uses DLCI 0, as confirmed next.
```

Example 18-1 *Basic Frame Relay Configuration Example (Continued)*

```
R3# sh frame lmi | include LMI TYPE
LMI Statistics for interface Serial0/0/0 (Frame Relay DTE) LMI TYPE = ANSI
R3# sh int s 0/0/0 | include LMI DLCI
LMI DLCI 0  LMI type is ANSI Annex D  frame relay DTE
```

At the end of Example 18-1, note that R3 is using the ANSI LMI type. R3 could have configured the LMI type statically using the **frame-relay lmi-type** {**ansi** | **cisco** | **q933a**} command, under the physical interface. However, R3 omitted the command, causing R3 to take the default action of autosensing the LMI type.

Frame Relay Payload Compression

Cisco IOS software supports three options for payload compression on Frame Relay VCs: *packet-by-packet*, *data-stream*, and *Frame Relay Forum Implementation Agreement 9 (FRF.9)*. FRF.9 is the only standardized protocol of the three options. FRF.9 compression and data-stream compression function basically the same way; the only real difference is that FRF.9 implies compatibility with non-Cisco devices.

All three FR compression options use LZS as the compression algorithm, but one key difference relates to their use of compression dictionaries. LZS defines dynamic dictionary entries that list a binary string from the compressed data, and an associated smaller string that represents it during transmission—thereby reducing the number of bits used to send data. The table of short binary codes, and their longer associated string of bytes, is called a *dictionary*. The packet-by-packet compression method also uses LZS, but the compression dictionary is built for each packet, then discarded—hence the name packet-by-packet. The other two methods do not clear the dictionary after each packet. Table 18-4 lists the three FR compression options and their most important distinguishing features.

Table 18-4 *FR Payload Compression Feature Comparison*

Feature	Packet-by-Packet	FRF.9	Data-Stream
Uses LZS algorithm?	Yes	Yes	Yes
Same dictionary for all packets?	No	Yes	Yes
Cisco-proprietary?	Yes	No	Yes

FR payload compression configuration is configured per VC. The configuration varies depending on whether point-to-point subinterfaces are used. On point-to-point subinterfaces, the **frame-relay payload-compress** ***type*** subinterface command is used; otherwise, the **frame-relay map** command must be configured along with the **payload-compress** ***type*** option. Example 18-2 shows Frame

Relay compression configured in the same network as shown in Figure 18-4 and Example 18-1. The VC from R1 to R3 (multipoint subinterface) uses data-stream compression, and the VC from R1 to R4 uses FRF.9.

Example 18-2 *Frame Relay Data-Stream Compression*

```
! Below, the configuration added to R1's Example 18-1 configuration is shown.
! R3 uses a frame-relay map command as well, and R4 uses the same
! frame-relay payload-compress command.
interface Serial0/0/0.14 point-to-point
 frame-relay payload-compress frf9 stac
!
interface Serial0/0/0.123 multipoint
 frame-relay map ip 10.1.123.3 103 broadcast payload-compress data-stream stac
```

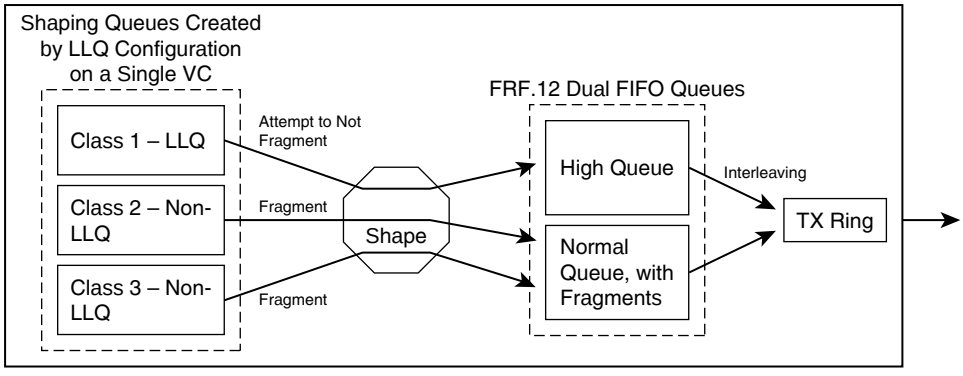
```
! Next, R1 sends 5000 200-byte pings to R4 to create traffic. R4 shows the pre- and
! post-compression stats in the show compress command.
R4# show compress
 Serial0/0/0 - DLCI: 101
         Software compression enabled
         uncompressed bytes xmt/rcv 1021536/1021536
         compressed bytes   xmt/rcv 178090/177820
         Compressed bytes sent:    178090 bytes   12 Kbits/sec  ratio: 5.736
         Compressed bytes recv:    177820 bytes   12 Kbits/sec  ratio: 5.744
         1  min avg ratio xmt/rcv 3.506/3.301
         5  min avg ratio xmt/rcv 3.506/3.301
         10 min avg ratio xmt/rcv 3.506/3.301
         no bufs xmt 0 no bufs rcv 0
         resyncs 0
         Additional Stac Stats:
         Transmit bytes:  Uncompressed =        0 Compressed =     142922
         Received bytes:  Compressed =     142652 Uncompressed =        0
```

Frame Relay Fragmentation

Frame Relay Forum IA 12, or FRF.12, defines a standard method of performing LFI over a Frame Relay PVC. Cisco IOS supports two methods for configuring FRF.12. The legacy FRF.12 configuration requires FRTS to be configured, and requires a queuing tool to be applied to the shaped packets. (Example 16-7 in Chapter 16 shows an FRTS **map-class shape-with-LLQ** command that shapes and applies LLQ.)

Figure 18-5 shows the overall logic of how FRF.12 interleaves packets using LFI, when configured using legacy FRF.12 configuration. IOS creates a 2-queue software queuing system on the physical interface. Any packets leaving the FRTS LLQ go into the “high” Dual FIFO queue, with the packets and fragments from other queuing going into the Dual FIFO “normal” queue. On the interface, IOS treats the Dual FIFO queue as a priority queue, which causes interleaving.

Figure 18-5 *Interface Dual FIFO Queues with FRTS Plus FRF.12*



NOTE All packets can be fragmented, but Cisco rightfully suggests choosing a fragment size so that the packets typically placed into the LLQ PQ will not be fragmented. Only packets from the shaping LLQ are placed into the Dual FIFO interface high queue, and only those packets are interleaved.

To configure legacy FRF.12, the **frame-relay fragment size** command is added to the FRTS map class on both ends of the VC. For example, in Example 16-7, the **frame-relay fragment 120** command could be added to the shape-with-LLQ map class, with the same configuration on the router on the other end of the VC, to enable FRF.12. Note that because fragmentation of any kind implies that an additional fragmentation header is used, fragmentation must be added on both ends of the link or VC.

The second method of configuring FRF.12 is called *Frame Relay Fragmentation at the Interface*, and was added to Cisco IOS Software Release 12.2(13)T. This method does not require FRTS; the **frame-relay fragment** command simply sits directly on the physical interface. If no queuing tool is configured on the interface, the router creates Dual FIFO queuing on the interface, interleaving all non-fragmented packets between fragments of other packets. Optionally, configuration of a queuing tool that has a PQ feature (for example, LLQ) can be used instead, causing packets in the PQ to be immediately interleaved. Example 18-3 shows a sample configuration using the same router, R1, from the first two examples in this chapter. In this case, FRF.12 has been enabled on s0/0/0, with a fragment size of 120.

Example 18-3 *FRF.12 on the Interface—Configuration*

```
! No FRTS configuration exists—simply the frame-relay fragment 120 end-to-end
! command. Note that LLQ is not enabled in this case, so nonfragmented packets
! will be interleaved using Dual FIFO.
R1# show run int s 0/0/0
interface Serial0/0/0
 encapsulation frame-relay
 frame-relay fragment 120 end-to-end
! Next, fragmentation stats are listed.
```

continues

Example 18-3 *FRF.12 on the Interface—Configuration (Continued)*

```
R1# show frame-relay fragment 104
interface               dlci frag-type  size in-frag    out-frag   dropped-frag
Se0/0/0.14              104  end-to-end 120  2759       2762       0
! The show queueing command (yes, IOS misspells it) lists statistics for the Dual
! FIFO queuing system added to the interface when FRF.12 is configured.
R1# show queueing int s0/0/0
Interface Serial0/0/0 queueing strategy: priority

Output queue utilization (queue/count)
    high/354 medium/0 normal/1422 low/0
```

Table 18-5 summarizes the key points regarding both styles of FRF.12 configuration.

Table 18-5 *Comparing Legacy and Interface FRF.12*

KEY POINT

Feature	Legacy FRF.12	FRF.12 on the Interface
Requires FRTS?	Yes	No
Interleaves by feeding Dual FIFO interface high queue from a shaping PQ?	Yes	No
Interleaves by using either Dual FIFO or a configured LLQ policy-map on the physical interface.	No	Yes
Config mode for the **frame-relay fragment** command	**map-class**	Physical interface

In addition to FRF.12, Cisco IOS supports two other methods of LFI over Frame Relay, including FRF.11-c. This fragmentation method works only on Voice over Frame Relay (VoFR) VCs. With this tool, voice frames are never fragmented, and voice frames are always interleaved, without requiring any particular queuing tool. Once a VoFR VC has been configured, the LFI configuration is identical to the legacy style of FRF.12 configuration.

The last type of FR LFI uses MLP over Frame Relay; it also happens to be the only option for Frame Relay-to-ATM Service Interworking. MLP over FR uses PPP headers instead of the Cisco or RFC 2427 header shown in Figure 18-2, thereby enabling many PPP features supported by the PPP headers. MLP and LFI configuration would simply need to be added to that configuration to achieve LFI.

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most Cisco Press *Exam Certification Guides*, this book does not repeat information covered in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 18-6 summarizes the key standards mentioned in the chapter.

Table 18-6 *Protocols and Standards for Chapter 18*

Topic	Standard
Frame Relay Encapsulation	RFC 2427
Frame Relay Compression	FRF.9
Frame Relay LFI	FRF.12, FRF.11-c
Frame Relay Service Interworking	FRF.8

Table 18-7 lists the Cisco IOS commands covered in this chapter.

Table 18-7 *Command Reference for Chapter 18*

Command	Mode and Function
frame-relay payload-compression {**packet-by-packet** | **frf9 stac** | **data-stream stac**}	Subinterface mode; defines the type of FR compression
encapsulation frame-relay [**cisco** | **ietf**]	Interface mode; enables FR, and chooses one of two encapsulation types
frame-relay broadcast-queue *size byte-rate packet-rate*	Interface mode; sets the FR broadcast queue size and rates
frame-relay fragment *fragment_size* [**switched**]	Map-class mode; enables fragmentation with fragments of the defined size
frame-relay fragment *fragment-size* **end-to-end**	Interface mode; enables interface FR fragmentation, based on size
frame-relay interface-dlci *dlci* [**ietf** | **cisco**] [**ppp** *virtual-template-name*]	Subinterface mode; associates a DLCI with the subinterface, and sets the encapsulation
frame-relay inverse-arp [*protocol*] [*dlci*]	Interface mode; enables InARP, per Layer 3 protocol and/or DLCI

continues

Table 18-7 *Command Reference for Chapter 18 (Continued)*

Command	Mode and Function
frame-relay lmi-type {**ansi** | **cisco** | **q933a**}	Interface mode; statically configures the LMI type
frame-relay map *protocol protocol-address* {*dlci* | **vc-bundle** *vc-bundle-name*}[**broadcast**] [**ietf** | **cisco**] [**payload-compression** {**packet-by-packet** | **frf9 stac** | **data-stream stac**]	Subinterface mode; maps Layer 3 protocol addresses of neighboring routers to DLCIs along with other settings associated with the PVC
keepalive *time-interval*	Interface mode; for FR, enables LMI messages every time interval
protocol ***protocol*** {*protocol-address* | **inarp**} [[**no**] **broadcast**]	PVC mode; maps a Layer 3 address to the PVC under which the command is issued
show compress	Displays compression statistics
show frame-relay fragment [**interface** *interface* [*dlci*]]	Displays fragmentation statistics
show frame-relay map	Displays mapping for physical and multipoint subinterfaces

Table 18-8 lists some of the ANSI and ITU standards for Frame Relay.

Table 18-8 *Frame Relay Protocol Specifications*

What the Specification Defines	ITU Document	ANSI Document
Data-link specifications, including LAPF header/trailer	Q.922 Annex A (Q.922-A)	T1.618
PVC management, LMI	Q.933 Annex A (Q.933-A)	T1.617 Annex D (T1.617-D)
SVC signaling	Q.933	T1.617
Multiprotocol encapsulation (originated in RFC 1490/2427)	Q.933 Annex E (Q.933-E)	T1.617 Annex F (T1.617-F)

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then,

simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

FRF, VC, PVC, SVC, DTE, DCE, LMI, access rate, access link, FRF.9, FRF.5, FRF.8, Service Interworking, FRF.12, FRF.11-c, VoFR, LAPF, NLPID, DE, FECN, BECN, Dual FIFO, LZS, DLCI

Further Reading

- *Troubleshooting Remote Access Networks*, by Dr. Plamen Nedeltchev
- *ISDN and Broadband ISDN with Frame Relay and ATM*, by Dr. William Stallings

Part VI: IP Multicast

Chapter 19 Introduction to IP Multicasting

Chapter 20 IP Multicast Routing

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- IP Multicast
  - — IGMP/CGMP
  - — Addressing

CHAPTER 19

Introduction to IP Multicasting

IP multicast concepts and protocols are an important part of the CCIE Routing and Switching written exam. Demand for IP multicast applications has increased dramatically over the last several years. Almost all major campus networks today use some form of multicasting. This chapter covers why multicasting is needed, the fundamentals of multicast addressing, and how multicast traffic is distributed and controlled over a LAN.

"Do I Know This Already?" Quiz

Table 19-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 19-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Why Do You Need Multicasting?	1	
Multicast IP Addresses	2–4	
Managing Distribution of Multicast Traffic	5–8	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

6. Which of the following reasons for using IP multicasting are valid for one-to-many applications?

a. Multicast applications use connection-oriented service.

b. Multicast uses less bandwidth than unicast.

c. A multicast packet can be sent from one source to many destinations.

d. Multicast eliminates traffic redundancy.

7. Which of the following statements is true of a multicast address?
   a. Uses a Class D address that can range from 223.0.0.0 to 239.255.255.255
   b. Uses a subnet mask ranging from 8 bits to 24 bits
   c. Can be permanent or transient
   d. Can be entered as an IP address on an interface of a router only if the router is configured for multicasting
8. Which of the following multicast addresses are reserved and not forwarded by multicast routers?
   a. 224.0.0.1 and 224.0.0.13
   b. 224.0.0.9 and 224.0.1.39
   c. 224.0.0.10 and 224.0.1.40
   d. 224.0.0.5 and 224.0.0.6
9. From the following pairs of Layer 3 multicast addresses, select a pair that will use the same Ethernet multicast MAC address of 0x0100.5e4d.2643.
   a. 224.67.26.43 and 234.67.26.43
   b. 225.77.67.38 and 235.77.67.38
   c. 229.87.26.43 and 239.87.26.43
   d. 227.77.38.67 and 238.205.38.67
10. From the following statements, select the true statement(s) regarding IGMP Query messages and IGMP Report messages.
    a. Hosts, switches, and routers originate IGMP Membership Report messages.
    b. Hosts, switches, and routers originate IGMP Query messages.
    c. Hosts originate IGMP Query messages and routers originate IGMP Membership messages.
    d. Hosts originate IGMP Membership messages and routers originate IGMP Query messages.
    e. Hosts and switches originate IGMP Membership messages and routers originate IGMP Query messages.

11. Seven hosts and a router on a multicast LAN network are using IGMPv2. Hosts 5, 6, and 7 are members of group 226.5.6.7, and the other 4 hosts are not. Which of the following answers is/are true about how the router will respond when Host 7 sends an IGMPv2 Leave message for the group 226.5.6.7?

    a. Sends an IGMPv2 General Query to multicast destination address 224.0.0.1

    b. Sends an IGMPv2 Group-Specific Query to multicast destination address 224.0.0.1

    c. Sends an IGMPv2 General Query to multicast destination address 226.5.6.7

    d. Sends an IGMPv2 Group-Specific Query to multicast destination address 226.5.6.7.

    e. First, sends an IGMPv2 Group-Specific Query to multicast destination address 226.5.6.7, and then sends an IGMPv2 General Query to multicast destination address 224.0.0.1

12. Which of the following statements is/are true about IGMP?

    a. IGMPv1, IGMPv2, and IGMPv3 use a fixed Maximum Response Time of 10 seconds.

    b. If more than one multicast router is connected to a LAN subnet, IGMPv1 elects the router with the lowest IP address as a querier, while IGMPv2 elects the router with the highest IP address as a querier.

    c. IGMPV2 is backward compatible with IGMPv1 but IGMPv3 is not backward compatible with IGMPv2 and IGMPv1.

    d. Leave messages are available only in IGMPv2; IGMPv1 and IGMPv3 cannot use Leave messages.

    e. All of these answers are correct.

    f. None of these answers is correct.

13. Which of the following statements is/are true regarding CGMP and IGMP snooping?

    a. CGMP and IGMP snooping are used to constrain the flooding of multicast traffic in LAN switches.

    b. CGMP is a Cisco-proprietary protocol and uses the well-known Layer 2 multicast MAC address 0x0100.0cdd.dddd.

    c. IGMP snooping is preferable in a mixed-vendor environment; however, if implemented using Layer 2–only LAN switches, it can cause a dramatic reduction in switch performance.

    d. CGMP is simple to implement, and in CGMP only routers send CGMP messages, while switches only listen for CGMP messages.

    e. All of these answers are correct.

    f. None of these answers is correct.

Foundation Topics

Why Do You Need Multicasting?

"Necessity is the mother of all invention," a saying derived from Plato's *Republic*, holds very true in the world of technology. In the late 1980s, Dr. Steve Deering was working on a project that required him to send a message from one computer to a group of computers across a Layer 3 network. After studying several routing protocols, Dr. Deering concluded that the functionality of the routing protocols could be extended to support "Layer 3 multicasting." This concept led to more research, and in 1991, Dr. Deering published his doctoral thesis, "Multicast Routing in a Datagram Network," in which he defined the components required for IP multicasting, their functions, and their relationships with each other.

The most basic definition of IP multicasting is as follows:

> Sending a message from a single source to selected multiple destinations across a Layer 3 network in one data stream.

If you want to send a message from one source to one destination, you could send a unicast message. If you want to send a message from one source to all the destinations on a local network, you could send a broadcast message. However, if you want to send a message from one source to selected multiple destinations spread across a routed network in one data stream, the most efficient method is IP multicasting.

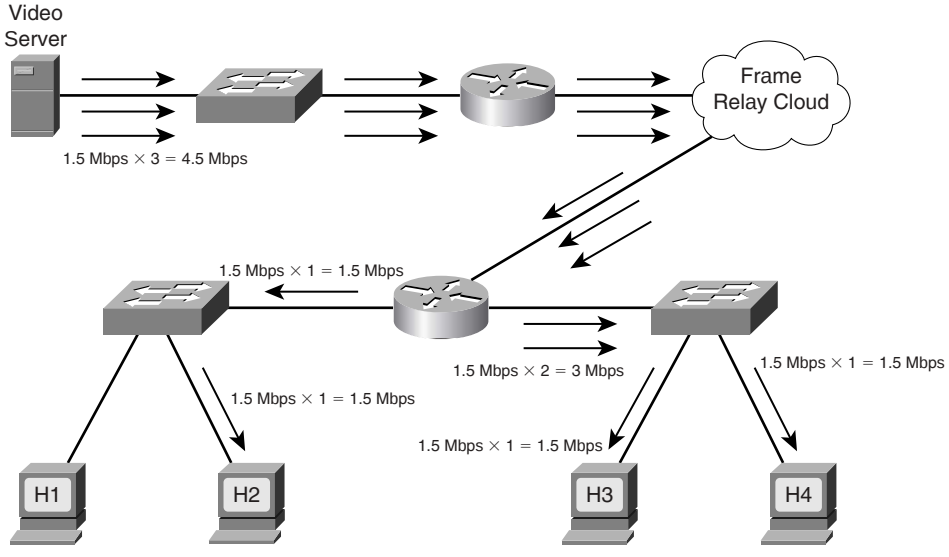
Demand for multicast applications is increasing with the advent of such applications as audio and video web content; broadcasting TV programs, radio programs, and concerts over the Internet; communicating stock quotes to brokers; transmitting a corporate message to employees; and transmitting data from a centralized warehouse to a chain of retail stores. Success of one-to-many multicast applications has created a demand for the second generation of multicast applications that are referred to as "many-to-many" and "many-to-few," in which there are many sources of multicast traffic. Examples of these types of applications include playing games on an intranet or the Internet and conducting interactive audio and video meetings. The primary focus of this chapter and the next chapter is to help you understand concepts and technologies required for implementing one-to-many multicast applications.

Problems with Unicast and Broadcast Methods

Why not use unicast or broadcast methods to send a message from one source to many destinations? Figure 19-1 shows a video server as a source of a video application and the video

data that needs to be delivered to a group of receivers—H2, H3, and H4—two hops away across a WAN link.

Figure 19-1 *Unicast*



The unicast method requires that the video application send one copy of each packet to every group member's unicast address. To support full-motion, full-screen viewing, the video stream requires about 1.5 Mbps of bandwidth for each receiver. If only a few receivers exist, as shown in Figure 19-1, this method works fine but still requires *n* * 1.5 Mbps of bandwidth, where *n* is the number of receiving hosts.

Figure 19-2 shows that as the number of receivers grows into the hundreds or thousands, the load on the server to create and send copies of the same data also increases, and replicated unicast transmissions consume a lot of bandwidth within the network. For 100 users, as indicated in the upper-left corner of Figure 19-2, the bandwidth required to send the unicast transmission increases to 150 Mbps. For 1000 users, the bandwidth required would increase to 1.5 Gbps.

You can see from Figure 19-2 that the unicast method is not scalable. Figure 19-3 shows that the broadcast method requires transmission of data only once, but it has some serious issues. First, as shown in Figure 19-3, if the receivers are in a different broadcast domain from the sender, routers need to forward broadcasts. However, forwarding broadcasts might be the worst possible solution, because broadcasting a packet to all hosts in a network can waste bandwidth and increase processing load on all the network devices if only a small group of hosts in the network actually needs to receive the packet.

Figure 19-2 *Unicast Does Not Scale to Large Numbers of Receivers*

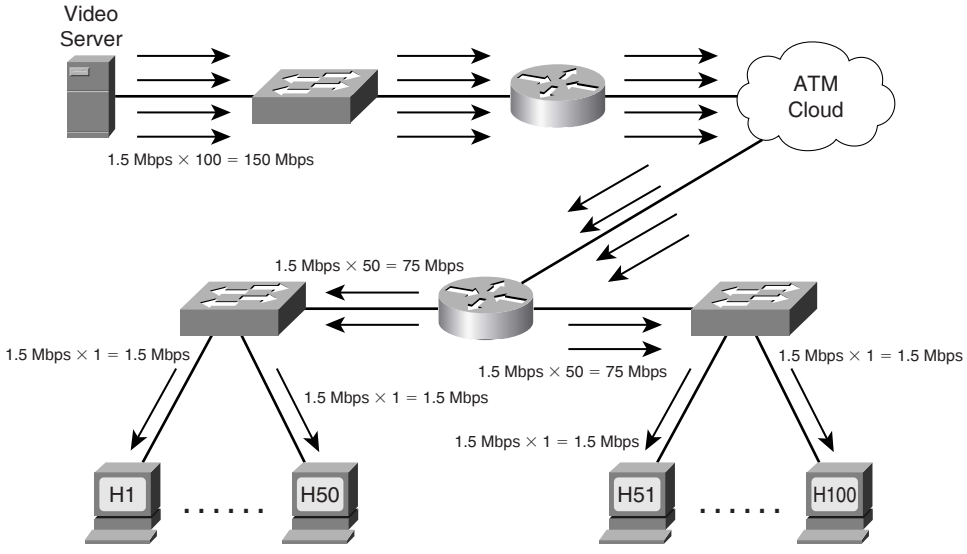
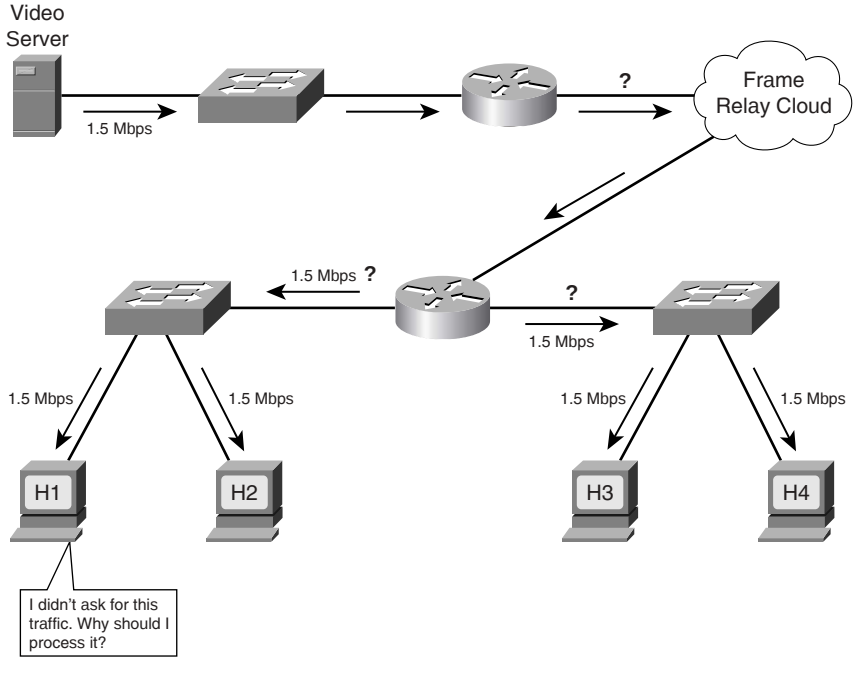


Figure 19-3 *Broadcast Wastes Bandwidth and Increases Processing Load on CPU*



How Multicasting Provides a Scalable and Manageable Solution

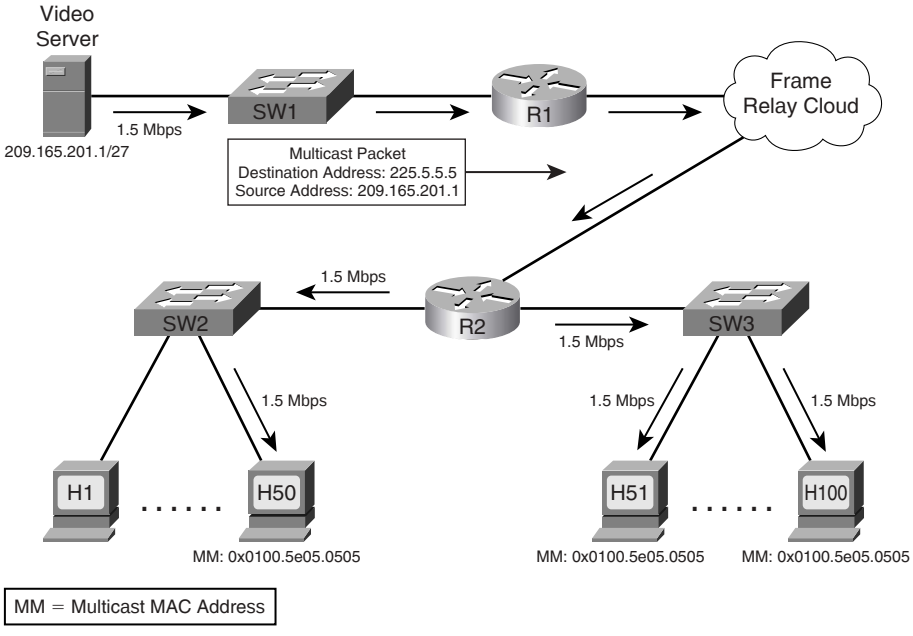
The six basic requirements for supporting multicast across a routed network are as follows:

- A designated range of Layer 3 addresses that can only be used by multicast applications must exist. A network administrator needs to install a multicast application on a multicast server using a Layer 3 multicast address from the designated range.
- A multicast address must be used only as a destination IP address, and specifically not as a source IP address. Unlike a unicast IP packet, a destination IP address in a multicast packet does not specify a recipient's address, but rather signifies that the packet is carrying multicast traffic for a specific multicast application.
- The multicast application must be installed on all the hosts in the network that need to receive the multicast traffic for the application. The application must be installed using the same Layer 3 multicast address that was used on the multicast server. This is referred to as *launching an application* or *joining a group*.
- All hosts that are connected to a LAN must use a standard method to calculate a Layer 2 multicast address from the Layer 3 multicast address and assign it to their network interface cards (NICs). For example, if multiple routers are connected to an Ethernet segment and all of them are using the OSPF routing protocol, all the routers on their Ethernet interfaces will also be listening to the Layer 2 multicast address 0x0100.5e00.0005 in addition to their Burned-In Addresses (BIAs). This Layer 2 multicast address 0x0100.5e00.0005 is calculated from the multicast Layer 3 address 224.0.0.5, which is reserved for the OSPF routing protocol.
- There must be a mechanism by which a host can dynamically indicate to the connected router whether it would like to receive the traffic for the installed multicast application. The Internet Group Management Protocol (IGMP) provides communication between hosts and a router connected to the same subnet. The Cisco Group Management Protocol (CGMP) or IGMP snooping helps switches learn which hosts have requested to receive the traffic for a specific multicast application and to which switch ports these hosts are connected.
- There must be a multicast routing protocol that allows routers to forward multicast traffic from multicast servers to hosts without overtaxing network resources. Some of the multicast routing protocols are Distance Vector Multicast Routing Protocol (DVMRP), Multicast Open Shortest Path First (MOSPF), and Protocol Independent Multicast dense mode (PIM-DM) and sparse mode (PIM-SM).

This chapter discusses the first five bulleted items, and Chapter 20, "IP Multicast Routing," covers the multicast routing protocols.

Figure 19-4 shows how multicast traffic is forwarded in a Layer 3 network. The purpose of this illustration is to give you an overview of how multicast traffic is forwarded and received by selected hosts.

Figure 19-4 *How Multicast Delivers Traffic to Selected Users*



Assume that a video multicast application was installed on the video server using the special Layer 3 multicast address 225.5.5.5. Hosts 1 to 49, located across a WAN link, are not interested at this time in receiving traffic for this application. Hosts 50 to 100 are interested in receiving traffic for this application and launch this application on their PCs. When the host launches the application, the host *joins the group*, which means that the host now wants to receive multicast packets sent to 225.5.5.5. Hosts 50 to 100 join group 225.5.5.5 and indicate to R2 their desire to receive traffic for this multicast application by using IGMP. The multicast application calculates the Layer 2 multicast address 0x0100.5e05.0505 from the Layer 3 multicast address 225.5.5.5, and NICs of hosts 50 to 100 are listening to this address in addition to their BIAs.

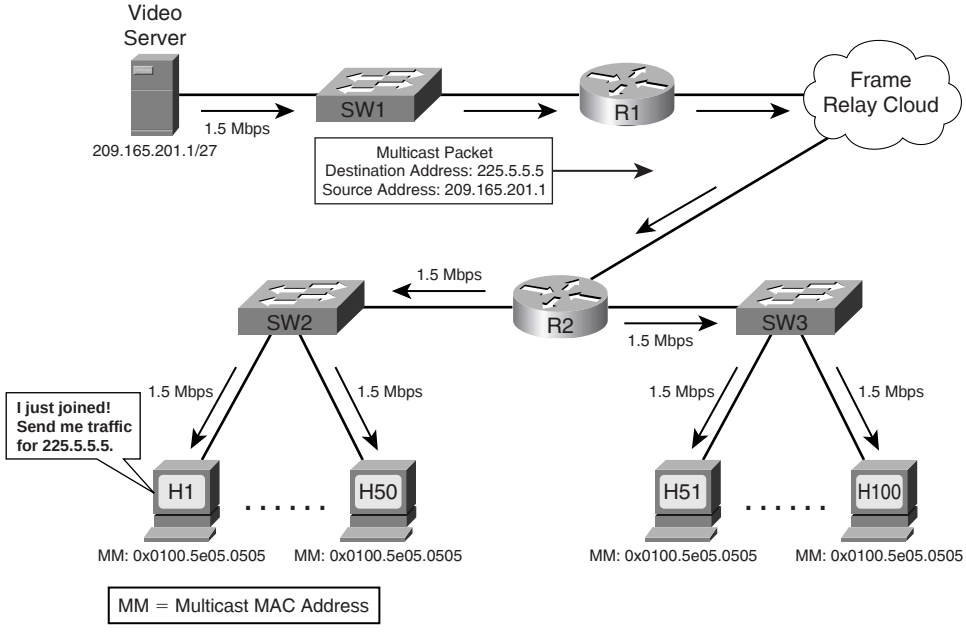
A multicast routing protocol is configured on R1 and R2 so that they can forward the multicast traffic. R2 has one WAN link connected to the Frame Relay cloud and two Ethernet links connected to two switches—SW2 and SW3. R2 knows that it has hosts on both Ethernet links that would like to receive multicast traffic for the group 225.5.5.5 because these hosts have indicated their desire to receive traffic for the group using IGMP. Both switches have also learned on which ports they have hosts that would like to receive the multicast traffic for this application by using either CGMP or IGMP snooping.

A multicast packet travels from the video server over the Ethernet link to R1, and R1 forwards a single copy of the multicast packet over the WAN link to R2. When R2 receives a multicast packet on the WAN link with the destination address 225.5.5.5, it makes a copy of the packet and

forwards a copy on each Ethernet link. Because it is a multicast packet for the group (application) 225.5.5.5, R2 calculates the Layer 2 destination multicast address of 0x0100.5e05.0505, and uses it as the destination MAC address on each packet it forwards to both switches. When the switches receive these packets, they forward them on appropriate ports to hosts. When the hosts receive the packets, their NICs compare the destination MAC address with the multicast MAC address they are listening to, and, because they match, inform the higher layers to process the packet.

You can see from Figure 19-4 that the multicast traffic is sent once over the WAN links and is received by the hosts that have requested it. Should additional hosts request to receive the same multicast traffic, neither the multicast server nor the network resources would incur any additional burden, as shown in Figure 19-5.

Figure 19-5 *Multicasting Is Scalable*



Assume that Hosts 1 to 49 have also indicated their desire to receive traffic for the multicast group 225.5.5.5 using IGMP. R2 is already forwarding the traffic to both switches. Either CGMP or IGMP snooping can help SW2 (shown in Figure 19-5) learn that hosts 1 to 49 have also requested the multicast traffic for the group, so that it can start forwarding the multicast traffic on ports connected to hosts 1 to 49. The additional 49 users are now receiving multicast traffic, and the load on the multicast server, load on other network devices, and demand for bandwidth on the WAN links remain the same. The load on SW2 shown in Figure 19-5 increases because it has to make 49 more copies of the multicast traffic and forward it on 49 more ports; however, it is now operating at the same level as the other switch. You can see that IP multicast is scalable.

Although multicast offers many advantages, it also has some disadvantages. Multicast is UDP-based and hence unreliable. Lack of TCP windowing and "slow start" mechanisms can result in network congestion. Some multicast protocol mechanisms occasionally generate duplicate packets and deliver packets out of order.

Multicast IP Addresses

Multicast applications always use a multicast IP address. This multicast address represents the multicast application and is referred to as a multicast *group*. Unlike a unicast IP address, which uniquely identifies a single IP host, a multicast address used as a destination address on an IP packet signifies that the packet is carrying traffic for a specific multicast application. For example, if a multicast packet is traveling over a network with a destination address 225.5.5.5, it is proclaiming to the network devices that "I am carrying traffic for the multicast application that uses multicast group address 225.5.5.5; do you want it?" A multicast address is never assigned to a network device, so it is never used as a source address. A source address on a multicast packet, or any IP packet, is always a unicast address.

Multicast Address Range and Structure

KEY POINT

The Internet Assigned Numbers Authority (IANA) has assigned class D IP addresses to multicast applications. The first 4 bits of the first octet for a class D address are always 1110. IP multicast addresses range from 224.0.0.0 through 239.255.255.255. As these addresses are used to represent multicast groups (applications) and not hosts, there is no need for a subnet mask for multicast addresses because they are not hierarchical. In other words, there is only one requirement for a multicast address: the first 4 bits of the first octet must be 1110. The last 28 bits are unstructured.

Well-Known Multicast Addresses

IANA controls the assignment of IP multicast addresses. To preserve multicast addresses, IANA is reluctant to assign individual IP multicast addresses to new applications without a good technical justification. However, IANA has assigned individual IP multicast addresses to popular network protocols.

IANA has assigned several ranges of multicast IP addresses for specific types of reasons. Those types are as follows:

KEY POINT

- Permanent multicast groups, in the range 224.0.0.0–224.0.1.255
- Addresses used with Source-Specific Multicast (SSM), in the range 232.0.0.0–232.255.255.255
- GLOP addressing, in the range 233.0.0.0–233.255.255.255
- Private multicast addresses, in the range 239.0.0.0–239.255.255.255

This section provides some insights into each of these four types of reserved IP multicast addresses. The rest of the multicast addresses are referred to as *transient* groups, which are covered later in this chapter in the section "Multicast Addresses for Transient Groups."

Multicast Addresses for Permanent Groups

IANA has reserved two ranges of permanent multicast IP addresses. The main distinction between these two ranges of addresses is that the first range is used for packets that should not be forwarded by routers, and the second group is used when packets should be forwarded by routers.

The range of addresses used for local (not routed) purposes is 224.0.0.0 through 224.0.0.255. These addresses should be somewhat familiar from the routing protocol discussions earlier in the book; for example, the 224.0.0.5 and 224.0.0.6 IP addresses used by OSPF fit into this first range of permanent addresses. Other examples include the IP multicast destination address of 224.0.0.1, which specifies that all multicast-capable hosts on a local network segment should examine this packet. Similarly, the IP multicast destination address of 224.0.0.2 on a packet specifies that all multicast-capable routers on a local network segment should examine this packet.

The range of permanent group addresses used when the packets should be routed is 224.0.1.0 through 224.0.1.255. This range includes 224.0.1.39 and 224.0.1.40, which are used by Cisco-proprietary Auto-Rendezvous Point (Auto-RP) protocols (covered in Chapter 20). Table 19-2 shows some of the well-known addresses from the permanent address range.

Table 19-2 *Some Well-Known Reserved Multicast Addresses*

KEY POINT

Address	Usage
224.0.0.1	All multicast hosts
224.0.0.2	All multicast routers
224.0.0.4	DVMRP routers
224.0.0.5	All OSPF routers
224.0.0.6	OSPF designated routers
224.0.0.9	RIPv2 routers
224.0.0.10	EIGRP routers
224.0.0.13	PIM routers
224.0.0.22	IGMPv3
224.0.0.25	RGMP
224.0.1.39	Cisco-RP-Announce
224.0.1.40	Cisco-RP-Discovery

Multicast Addresses for Source-Specific Multicast Applications and Protocols

IANA has allocated the range 232.0.0.0 through 232.255.255.255 for SSM applications and protocols. The purpose of these applications is to allow a host to select a source for the multicast group. SSM makes multicast routing efficient, allows a host to select a better-quality source, and helps network administrators minimize multicast denial-of-service (DoS) attacks.

NOTE Only IGMPv3-capable hosts can use the SSM feature. IGMPv3 is a new protocol. At the time of this writing, a very limited number of IGMPv3 applications were available. Hence, use of these addresses is minimal.

Multicast Addresses for GLOP Addressing

IANA has reserved the range 233.0.0.0 through 233.255.255.255 (RFC 2770), called GLOP addressing, on an experimental basis. It can be used by anyone who owns a registered autonomous system number (ASN) to create 256 global multicast addresses that can be owned and used by the entity. IANA reserves addresses to ensure global uniqueness of addresses; for similar reasons, each autonomous system should be using an assigned unique ASN.

By using a value of 233 for the first octet, and by using the ASN for the second and third octets, a single autonomous system can create globally unique multicast addresses as defined in the GLOP addressing RFC. For example, the autonomous system using registered ASN 5663 could covert ASN 5663 to binary (0001011000011111). The first 8 bits, 00010110, equals 22 in decimal notation, and the last 8 bits, 00011111, equals 31 in decimal notation. Mapping the first 8 bits to the second octet and the last 8 bits to the third octet in the 233 range addresses, the entity who owns the ASN 5663 is automatically allocated the address range 233.22.31.0 through 233.22.31.255.

NOTE GLOP is not an acronym and does not stand for anything. One of the authors of RFC 2770, David Meyer, started referring to this range of addresses as "GLOP" addressing, and since then the range has been identified by the name GLOP addressing.

Multicast Addresses for Private Multicast Domains

The last of the reserved multicast address ranges mentioned here is the range of *administratively scoped* addresses. IANA has assigned the range 239.0.0.0 through 239.255.255.255 (RFC 2365) for use in private multicast domains, much like the IP unicast ranges defined in RFC 1918, namely 10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16. IANA will not assign these administratively scoped multicast addresses to any other protocol or application. Network administrators are free to use multicast addresses in this range; however, they must configure their multicast routers to ensure that multicast traffic in this address range does not leave their multicast domain boundaries.

Multicast Addresses for Transient Groups

When an enterprise wants to use globally unique unicast addresses, it needs to get a block of addresses from its ISP or from IANA. However, when an enterprise wants to use a multicast address for a global multicast application, it can use any multicast address that is not part of the well-known permanent multicast address space covered in the previous sections. These remaining multicast addresses are called *transient groups* or *transient multicast addresses*. This means that the entire Internet must share the transient multicast addresses; they must be dynamically allocated when needed and must be released when no longer in use.

Because these addresses are not permanently assigned to any application, they are called "transient." Any enterprise can use these multicast addresses without requiring any registration or permission from IANA, but the enterprise is expected to release these multicast addresses after their use. At the time of this writing (mid 2005), there is no standard method available for using the transient multicast addresses. However, a great deal of work is being done by IETF to define and implement a standard method for dynamically allocating multicast addresses.

Summary of Multicast Address Ranges

Table 19-3 summarizes various multicast address ranges and their use.

Table 19-3 *Multicast Address Ranges and Their Use*

KEY POINT

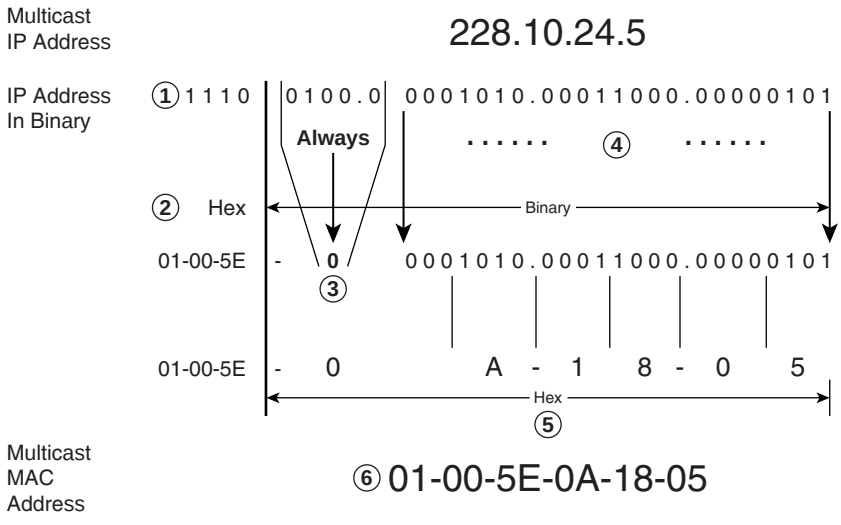
Multicast Address Range	Usage
224.0.0.0 to 239.255.255.255	This range represents the entire IPv4 multicast address space. It is reserved for multicast applications.
224.0.0.0 to 224.0.0.255	This range is part of the permanent groups. Addresses from this range are assigned by IANA for network protocols on a local segment. Routers do not forward packets with destination addresses used from this range.
224.0.1.0 to 224.0.1.255	This range is also part of the permanent groups. Addresses from this range are assigned by IANA for the network protocols that are forwarded in the entire network. Routers forward packets with destination addresses used from this range.
232.0.0.0 to 232.255.255.255	This range is used for SSM applications.
233.0.0.0 to 233.255.255.255	This range is called the GLOP addressing. It is used for automatically allocating 256 multicast addresses to any enterprise that owns a registered ASN.
239.0.0.0 to 239.255.255.255	This range is used for private multicast domains. These addresses are called administratively scoped addresses.
Remaining ranges of addresses in the multicast address space	Addresses from these ranges are called transient groups. Any enterprise can allocate a multicast address from the transient groups for a global multicast application, and should release it when the application is no longer in use.

Mapping IP Multicast Addresses to MAC Addresses

Assigning a Layer 3 multicast address to a multicast group (application) automatically generates a Layer 2 multicast address. Figure 19-6 shows how a multicast MAC address is calculated from a Layer 3 multicast address. The MAC address is formed using an IEEE-registered OUI of 01005E, then a binary 0, and then the last 23 bits of the multicast IP address. The method is identical for Ethernet and Fiber Distributed Data Interface (FDDI).

Figure 19-6 *Calculating a Multicast Destination MAC Address from a Multicast Destination IP Address*

KEY POINT



To understand the mechanics of this process, use the following six steps, which are referenced by number in Figure 19-6:

Step 1 Convert the IP address to binary. Notice the first 4 bits; they are always 1110 for any multicast IP address.

Step 2 Replace the first 4 bits 1110 of the IP address with the 6 hexadecimal digits (or 24 bits) 01-00-5E as multicast OUI, in the total space of 12 hexadecimal digits (or 48 bits) for a multicast MAC address.

Step 3 Replace the next 5 bits of the binary IP address with one binary 0 in the multicast MAC address space.

Step 4 Copy the last 23 bits of the binary IP address in the last 23-bit space of the multicast MAC address.

Step 5 Convert the last 24 bits of the multicast MAC address from binary to 6 hexadecimal digits.

Step 6 Combine the first 6 hexadecimal digits 01-00-5E with the last 6 hexadecimal digits, calculated in Step 5, to form a complete multicast MAC address of 12 hexadecimal digits.

Unfortunately, this method does not provide a unique multicast MAC address for each multicast IP address, because only the last 23 bits of the IP address are mapped to the MAC address. For example, the IP address 238.10.24.5 produces exactly the same MAC address, 0x01-00-5E-0A-18-05, as 228.10.24.5. In fact, because 5 bits from the IP address are always mapped to 0, 2^5 (32) different class D IP addresses produce exactly the same MAC address. IETF points out that the chances of two multicast applications on the same LAN producing the same MAC address are very low. If it happens accidentally, a packet from a different IP multicast application can be identified at Layer 3 and discarded; however, network administrators should be careful when they implement multicast applications so that they can avoid using IP addresses that produce identical MAC addresses.

Managing Distribution of Multicast Traffic with IGMP

Refer to Figure 19-4. Assume that R2 has started receiving multicast traffic from the server. R2 has to make a decision about forwarding this traffic on the Ethernet links. R2 needs to know the answers to the following questions:

- Is there any host connected to any of my Ethernet links that has shown interest in receiving this traffic?
- If none of the hosts has shown any interest in receiving this traffic, why should I forward it on the Ethernet links and waste bandwidth?
- If any host have shown interest in receiving this traffic, where are they located? Are they connected to one of my Ethernet links or to both?

As you can see, a mechanism is required for hosts and a local router to communicate with each other. The Internet Group Management Protocol (IGMP) was designed to enable communication between a router and connected hosts.

Not only do routers need to know out which LAN interface to forward multicast packets, but switches also need to know on which ports they should forward the traffic. By default, if a switch receives a multicast frame on a port, it will flood the frame throughout the VLAN, just like it would do for a broadcast or unknown unicast frame. The reason is that switches will never find a multicast MAC address in their Content Addressable Memory (CAM) table, because a multicast MAC address is never used as a source address.

A switch's decision to flood multicast frames means that if any host or hosts in a VLAN request to receive the traffic for a multicast group, all the remaining hosts in the same VLAN, whether they have requested to receive the traffic for the multicast group or not, will receive the multicast traffic. This behavior is contrary to one of the major goals of multicast design, which is to deliver multicast traffic to only those hosts that have requested it, while maximizing bandwidth efficiency.

To forward traffic more efficiently in Figure 19-4, SW2 and SW3 need to know the answers to the following questions:

- Should I forward this multicast traffic on all the ports in this VLAN, or only on specific ports?
- If I should forward this multicast traffic on specific ports of a VLAN, how will I find those port numbers?

Three different tools, namely CGMP, IGMP snooping, and RGMP, allow switches to optimize their multicast forwarding logic by answering these kinds of questions. These topics are covered in more depth later in the chapter. For now, this section focuses on how routers and hosts use IGMP to make sure the router knows whether it should forward multicasts out the router's LAN interfaces.

Joining a Group

Before a host can receive any multicast traffic, a multicast application must be installed and run on that host. The process of installing and running a multicast application is referred to as *launching an application* or *joining a multicast group*. After a host joins a group, the host software calculates the multicast MAC address and its NIC then starts listening to the multicast MAC address, in addition to its BIA.

Before a host (or a user) can join a group, the user needs to know what groups are available and how to join them. For enterprise-scale multicast applications, the user may simply find a link on a web page and click it, prompting the user's multicast client application to start working with the correct multicast address—totally hiding the multicast address details. Alternately, for an internally developed multicast application, the multicast address can be pre-configured on the client application. For example, a user might be required to log on to a server and authenticate with a name and a password; if the user is authenticated, the multicast application automatically installs on the user's PC, which means the user has joined the multicast group. When the user no longer wants to use the multicast application, the user must leave the group. For example, the user may simply close the multicast application to leave the group.

The process by which a human discovers which multicast IP address to listen for and join can be a challenge, particularly for multicast traffic on the Internet. The problem is similar to when you have a satellite or digital cable TV system at home—you might have literally thousands of channels, but finding the channel that has the show you want to watch might require a lot of surfing through the list of channels and time slots. For IP multicast, a user needs to discover what applications they may want to use, and the multicast IP addresses used by the application(s). A lot of work remains to be done in this area, but some options are available. For example, online TV program guides and web-based schedules advertise events that will use multicast groups, and specify who to contact if you want to see the event, lecture, or concert. Tools like Session Description Protocol (SDP) and Service Advertising Protocol (SAP) also describe multicast events and advertise them. However, a detailed discussion of the different methods, their limitations, and procedures for using them are beyond the scope of this book. The rest of the discussion in this section assumes that hosts have somehow learned about a multicast group.

Internet Group Management Protocol

IGMP has evolved from the Host Membership Protocol, described in Dr. Steve Deering's doctoral thesis, to IGMPv1 (RFC 1112), to IGMPv2 (RFC 2236), to the latest, IGMPv3 (RFC 3376). IGMP messages are sent in IP datagrams with IP protocol number 2, with the IP Time-to-Live (TTL) field set to 1. IGMP packets only pass over a LAN, and are not forwarded by routers, due to their TTL field values.

The two most important goals of IGMP are as follows:

KEY POINT

- To inform a local multicast router that a host wishes to receive multicast traffic for a specific group
- To inform local multicast routers that a host wishes to leave a multicast group (in other words, the host is no longer interested in receiving the multicast group traffic)

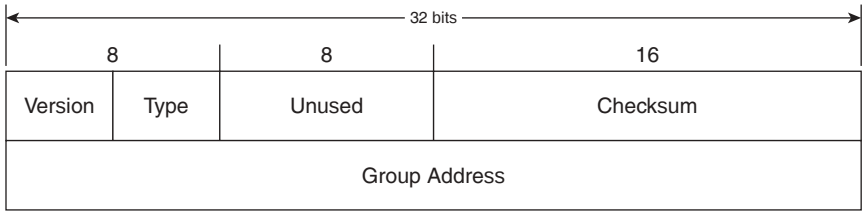
Multicast routers use IGMP to maintain information for each router interface about which multicast group traffic they should forward, and the hosts that want to receive it.

The following sections examine IGMPv1 and IGMPv2 in detail and introduce important features of IGMPv3. In the figures that show the operation of IGMP, Layer 2 switches are not shown because IGMP is used for communication between hosts and routers. Later in the chapter, the sections "Cisco Group Management Protocol," "IGMP Snooping," and "Router-Port Group Management Protocol" discuss the operation of multicasting at Layer 2.

IGMP Version 1

Figure 19-7 shows the 8-octet format of an IGMPv1 message.

Figure 19-7 *IGMPv1 Message Format*



The IGMPv1 message has five fields:

- **Version**—4-bit field that is always set to 1.
- **Type**—4-bit field that is either of two message types defined by IGMPv1:
  - **Type 1, Host Membership Query**—Used only by routers.
  - **Type 2, Host Membership Report**—Used only by hosts.

- **Unused**—8-bit field that contains zeros when sent and is ignored when received.
- **Checksum**—Carries the 16-bit checksum computed by the source for the IGMP message. The receiving device examines the checksum value, and if it does not match with its own calculated value, the receiver discards the frame.
- **Group Address**—Set to 0.0.0.0 when a router sends a Membership Query, and set to the multicast group address when a host sends a Membership Report.

Note that when you combine the Version and Type fields, the hexadecimal value of the first byte of an IGMPv1 Host Membership Query message would be 0x11 and an IGMPv1 Host Membership Report would be 0x12. Later, these values will be compared with the values of IGMPv2.

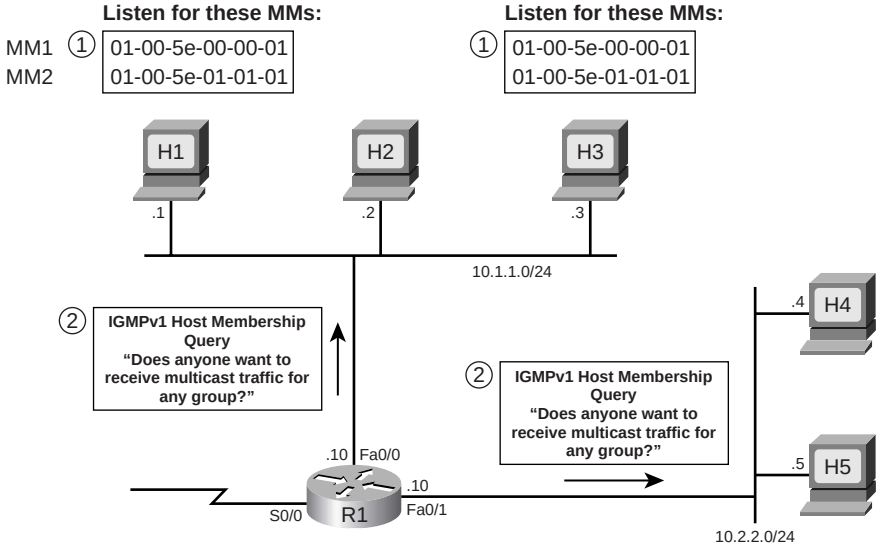
NOTE The values and names of various timers used in the following sections of IGMPv1 are not explicitly defined in RFC 1112; however, they are defined in RFC 2236 for IGMPv2. To maintain consistency and clarify differences in values of timers used in two versions, the same timers names are used in the IGMPv1 and IGMPv2 sections.

IGMPv1 Host Membership Query Functions

Multicast routers use the IGMPv1 Host Membership Query message to determine whether it has any host on any of its LAN interfaces that wants multicast traffic for any group. The IGMPv1 Host Membership Query message is sent only by multicast routers on LAN interfaces. For example, Figure 19-8 shows the IGMPv1 Host Membership Query process. It lists two steps, with the second step being the router sending the Query.

Figure 19-8 *IGMPv1 Host Membership Query Process*

KEY POINT



The details of the two steps are as follows:

1. Hosts H1 and H3 join multicast group 226.1.1.1. The Join causes these hosts to prepare to receive messages sent to both 226.1.1.1 (the joined group) and 224.0.0.1 (the address to which IGMPv1 Queries will be sent). Multicast hosts must listen to the well-known 224.0.0.1 multicast group address to participate in IGMP and, as a result, to receive multicast queries sent by the router. The Join causes these hosts to calculate the two multicast MAC (MM) addresses, 01-00-5e-01-01-01 (from 226.1.1.1) and 01-00-5e-00-00-01 (from 224.0.0.1), and then listen for frames sent to these two MMs.

KEY POINT

2. R1 sends an IGMPv1 Host Membership Query out each LAN interface, looking for any host interested in receiving packets for any multicast group. R1 periodically sends IGMPv1 Queries on each LAN interface, by default, every 60 seconds. This time period is called the *Query Interval*. R1's Queries use a destination IP address and MAC address of 224.0.0.1 and 01-00-5e-00-00-01, with the source IP address and MAC address of R1's interface IP address and BIA, respectively. After sending IGMPv1 Queries, R1 expects any host that has joined group 226.1.1.1, or any other group, to reply with an IGMPv1 Report. The IGMPv1 Queries also use a TTL of 1, preventing the packet from being routed.

   The IGMPv1 Query message's Group Address field (see Figure 19-7) is always 0.0.0.0. By sending the IGMPv1 Query message with the Group Address 0.0.0.0, the router is asking the hosts on each LAN, "Does anyone want to receive multicast traffic for *any group*?"

At this point, router R1 still does not know if any hosts need to receive any multicast traffic. The next section covers how the hosts respond with IGMP Report messages to inform R1 of their interest in multicast packets.

IGMPv1 Host Membership Report Functions

KEY POINT

Hosts use IGMPv1 Host Membership Report message to reply to IGMP Queries and inform the router(s) of their desire to receive multicasts. Multicast hosts use IGMPv1 Host Membership Report messages to communicate to a local router for which multicast groups they want to receive traffic.

In IGMPv1, a host sends an IGMPv1 Host Membership Report under the following two conditions:

- When a host receives an IGMPv1 Query from a local router, it is supposed to send an IGMPv1 Host Membership Report for all the multicast groups for which it wants to receive multicast traffic. This Report is called an IGMPv1 Solicited Host Membership Report.
- When a host joins a new group, the host immediately sends an IGMPv1 Host Membership Report to inform a local router that it wants to receive multicast traffic for the group it has just joined without waiting to receive an IGMPv1 Query. This Report is called an IGMPv1 Unsolicited Host Membership Report.

The operations of IGMPv1 Solicited Host Membership Report and IGMPv1 Unsolicited Host Membership Report are explained in the following sections.

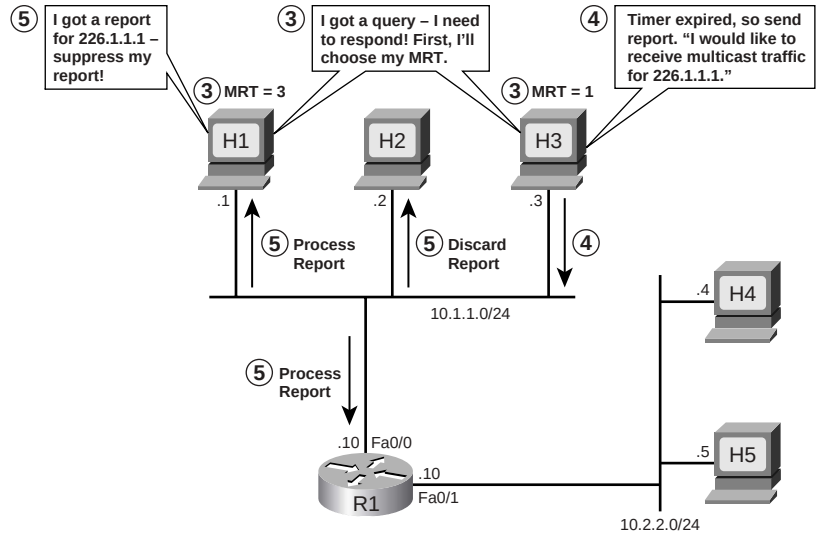
> **NOTE** The terms *IGMPv1 Solicited Host Membership Report* and *IGMPv1 Unsolicited Host Membership Report* are not defined in IGMPv1 RFC 1112. They are used in this book to specify whether the IGMPv1 Report was sent in response to a Query (solicited) or simply when the client application comes up (unsolicited).

IGMPv1 Solicited Host Membership Report

Figure 19-9 shows operation of the IGMPv1 Solicited Host Membership Report process and the Report Suppression mechanism. Figure 19-9 picks up the example from Figure 19-8, in which router R1 had sent an IGMPv1 Query.

Figure 19-9 *IGMPV1 Solicited Host Membership Report and Report Suppression Processes*

KEY POINT



If many hosts have launched multicast applications and if all of them respond to the Host Membership Query with the Host Membership Report, too many redundant reports would be sent to the router. This would waste bandwidth and unnecessarily increase the processing load on the router. A multicast router needs to receive only one report for each application on each of its LAN interfaces. A multicast router would forward the traffic for a multicast application on a LAN interface whether one user has requested to receive the traffic or 200 users have requested to receive the traffic for the group.

The Report Suppression mechanism helps to solve these problems. It uses the IGMPv1 Maximum Response Time (MRT) timer to suppress many of the unnecessary IGMPv1 Reports. This timer is called the *Query Response Interval*. It is fixed at 10 seconds and cannot be configured to a different value in IGMPv1. In other words, in IGMPv1, when any host receives an IGMPv1 Query, it has a maximum of 10 seconds to send the IGMPv1 Report if it wants to receive multicast traffic for any application. Each host that wants to send the IGMPv1 Solicited Host Membership Report picks randomly a time between 0 and 10 seconds and starts a timer. When this timer expires, the host is supposed to send a report. However, if a host receives a report sent by another host for the same multicast group for which it was planning to send the report, it does not send the report. This is called *Report Suppression* and is designed to reduce redundant reports. The unit of measurement for the MRT is 1/10 of a second. For example, a 3-second MRT is expressed as 30.

The following three steps, referenced in Figure 19-9 and a continuation of the steps referenced in Figure 19-8, describe the sequence of events for the IGMPv1 Solicited Host Membership Report and Report Suppression mechanism:

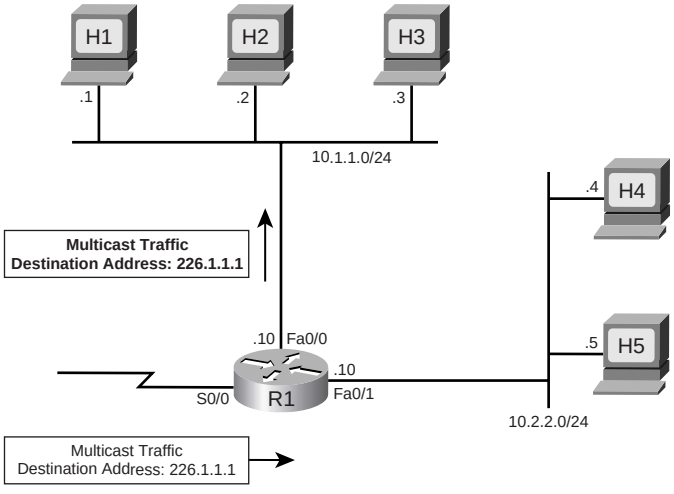
3. Hosts H1 and H3 would like to send IGMPv1 Solicited Host Membership Reports. Assume that H1 and H3 have received an IGMPv1 Query (as shown in step 2 of Figure 19-8). Because both H1 and H3 have joined the group 226.1.1.1, they need to send an IGMPv1 Solicited Host Membership Report. Further assume that H1 and H3 have randomly picked a Maximum Response Time (MRT) of 3 seconds and 1 second, respectively.

4. H3's timer expires in 1 second; it prepares and sends the IGMPv1 Solicited Host Membership Report with the TTL value of 1. H3 uses the destination IP address 226.1.1.1 and the source IP address 10.1.1.3, the destination MAC address 01-00-5e-01-01-01 calculated from the Layer 3 address 226.1.1.1, and its BIA address as the source address.

5. Hosts H1, H2, and R1 receive the IGMPv1 Solicited Host Membership Report, but only H1 and R1 process the Report. The NIC of H2 discards the frame sent by H3 because it is not listening to the address 01-00-5e-01-01-01. H1 realizes that H3 has already made a request to the router to forward the traffic for the same multicast group 226.1.1.1. Therefore, H1 suppresses its own Report and does not send it. By using the Group Address of 226.1.1.1, H3 is telling the multicast router, "I would like to receive multicast traffic for group 226.1.1.1."

R1 has now received the IGMPv1 Solicited Host Membership Report on its fa0/0 interface requesting traffic for multicast group 226.1.1.1, but it has not received a Host Membership Report on its fa0/1 interface. Figure 19-10 shows that R1 has started forwarding multicast traffic for group 226.1.1.1 on its fa0/0 interface.

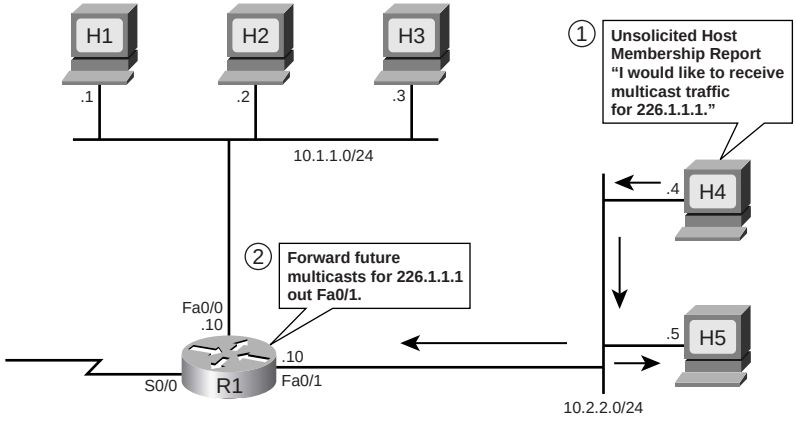
Figure 19-10 *R1 Forwarding Traffic for Group 226.1.1.1 on Its Fa0/0 Interface*



IGMPv1 Unsolicited Host Membership Report

In IGMPv1, a multicast router sends IGMP Host Membership Query messages by default every 60 seconds (Query Interval) on each of its LAN interfaces to determine whether any host wants to receive multicast traffic for any group. However, a host does not have to receive the Host Membership Query message from the router to send a Host Membership Report. A host can send an IGMPv1 Unsolicited Host Membership Report anytime a user launches a multicast application. This feature reduces the waiting time for a host to receive traffic for a multicast group. For example, Figure 19-11 shows that a user has launched a multicast application that uses 226.1.1.1 on H4. H4 sends an IGMPv1 Unsolicited Host Membership Report, which will be received by R1 on its fa0/1 interface, and R1 will then start forwarding traffic for 226.1.1.1 on its fa0/1 interface.

Figure 19-11 *H4 Sends IGMPv1 Unsolicited Host Membership Report*



IGMPv1 Leave Mechanism

Unfortunately, IGMPv1 does not have an explicit "Leave" mechanism that hosts can use to inform local routers that they no longer want to receive multicast traffic. For example, assume that a user sitting at H4 shown in Figure 19-11 receives group 226.1.1.1 traffic for a few minutes, but then decides to select another group or closes the application window. H4 has quietly left the group and R1 does not know about it. R1 will continue forwarding traffic for 226.1.1.1 on its fa0/1 interface for (up to) 3 minutes (default,) even if nobody wants to receive it, wasting bandwidth. After R1 sends its Host Membership Query message three times, by default 60 seconds apart, and nobody responds with an IGMPv1 Solicited Host Membership Report for 226.1.1.1, R1 concludes that no hosts want to receive traffic for 226.1.1.1 on its fa0/1 interface. This 3-minute timer is called the *Group Membership Interval*.

Thus, after H4 leaves group 226.1.1.1, R1 continues forwarding traffic for 226.1.1.1 for 3 minutes on the subnet. R1 then learns that no host wants traffic for 226.1.1.1 and stops forwarding these packets on its fa0/1 interface.

IGMPv1 Querier

For redundancy, it might be desirable to have multiple multicast routers connected to the same subnet. However, making all of them send Queries every minute is a waste of bandwidth. It is important to have only one preferred router and assign that router the responsibility of sending Query messages and forwarding (or stopping) multicast traffic on the subnet. If the first preferred router is not available, the second router should take the responsibility. RFC 1112 does not specify how the IGMPv1 preferred router should be selected. Instead, IGMPv1 depends on the Layer 3 IP multicast routing protocol to solve this problem by electing a designated router for the subnet. Multicast routing protocols are covered in Chapter 20.

IGMP Version 2

One of the primary reasons for developing IGMPv2 was to provide a better Leave mechanism to shorten the leave latency. IGMPv2 has the following new features:

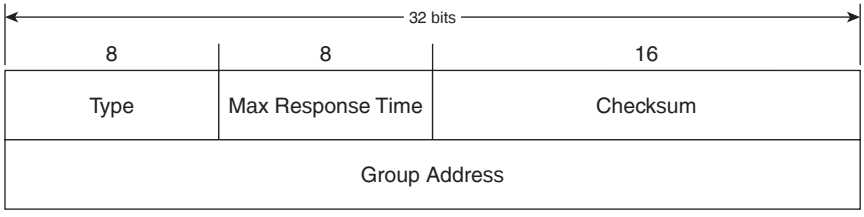
KEY POINT

- **Leave Group messages**—Provide hosts with a method for notifying routers that they wish to leave the group.
- **Group-Specific Query messages**—Permit the router to send a query for a specific group instead of all groups.
- **Maximum Response Time field**—A new field in Query messages that permits the router to specify the MRT. This field allows for tuning the response time for the Host Membership Report. This feature can be useful when a large number of groups are active on a subnet and you want to decrease the burstiness of the responses by spreading the responses over a longer period of time.

- **Querier election process**—Provides the method for selecting the preferred router for sending Query messages when multiple routers are connected to the same subnet.

Figure 19-12 shows the 8-octet format of an IGMPv2 message. Note that only the second field in the header, the Maximum Response Time field, differs from the IGMPv1 message shown in Figure 19-7. However, many other values are defined for the Type field.

Figure 19-12 *IGMPV2 Message Format*



IGMPv2 has four fields, which are defined as follows:

- **Type**—8-bit field that is one of four message types defined by IGMPv2:

KEY POINT

  - **Membership Query (Type code = 0x11)**—Used by multicast routers to discover the presence of group members on a subnet. A General Membership Query message sets the Group Address field to 0.0.0.0 exactly like IGMPv1 does. A Group-Specific Query sets the Group Address field to the address of the group being queried. It is sent by a router after it receives the IGMPv2 Leave Group message from a host. It is used to determine whether a specific multicast group has any remaining members on a subnet.
  - **Version 1 Membership Report (Type code = 0x12)**—Used by IGMPv2 hosts for backward compatibility with IGMPv1.
  - **Version 2 Membership Report (Type Code = 0x16)**—Sent by a group member to inform the router that at least one group member is present on the subnet.
  - **Leave Group (Type code = 0x17)**—Sent by a group member if it was the last member to send a Membership Report, to inform the router that it is leaving the group.

- **Maximum Response Time**—8-bit field included only in Query messages. The units are 1/10 of a second, with 100 (10 seconds) being the default. The values range from 1 to 255 (.1 to 25.5 seconds).

- **Checksum**—Carries the 16-bit checksum computed by the source. The IGMP checksum is computed over the whole IP payload, not just over the first 8 octets, even though IGMPv2 messages are only 8 bytes in length.

- **Group Address**—Set to 0.0.0.0 in General Query messages and to the group address in Group-Specific messages. Membership Report messages carry the address of the group being reported in this field; Leave Group messages carry the address of the group being left in this field.

IGMPv2 supports complete backward compatibility with IGMPv1. Note that the IGMPv2 Type codes 0x11 and 0x12 match the type codes for IGMPv1 for the Membership Query and Membership Report messages. This enables IGMPv2 hosts and routers to recognize IGMPv1 messages when other IGMPv1 hosts or routers are on the network.

IGMPv2 helps reduce surges in IGMPv2 Solicited Report messages sent by hosts in response to IGMPv2 Query messages by allowing the network administrator to change the Query Response Interval. IGMPv1 Query messages do not have a Maximum Response Time field, so each host simply uses a default MRT of 10 seconds. However, the IGMPv2 Query message includes an MRT field, stating an MRT to be used by all IGMPv2 hosts on the LAN. By setting MRT slightly higher, the hosts' collective IGMPv2 Solicited Report messages are spread over a longer time period, resulting in more uniform consumption of subnet bandwidth and router resources.

The coverage of IGMPv1 earlier in this chapter described how a host can send an IGMP Report in response to a Query, or simply send a Report when the host's application first comes up. These processes work the same way with IGMPv2, with one small difference: the router acting as the IGMPv2 querier sends general IGMP Query messages every 125 seconds, instead of every 60 seconds, as with IGMPv1. Because the operations of IGMPv2 General Query messages and Report messages are identical to those of IGMPv1, they are not repeated in this section. (Refer to the earlier "IGMPv1 Host Membership Query Functions" and "IGMPv1 Host Membership Report Functions" sections for more details.)

IGMPv2 improves the Query/Report process as well by using the IGMPv2 Group-Specific Query. In IGMPv2, when a host leaves a group, it sends an IGMPv2 Leave message. When an IGMPv2 router receives a Leave message, instead of waiting for the normal Query Interval timer (125 seconds by default) to expire, the IGMPv2 router immediately sends a Group-Specific Query for that group. The Group-Specific Query only asks if there are any remaining hosts wanting packets for that single multicast group. As a result, the router quickly knows if there are any remaining hosts interested in that multicast group on that LAN.

The main advantage of IGMPv2 over IGMPv1 is IGMPv2's shorter leave latency. Recall from the "IGMPv1 Leave Mechanism" section that an IGMPv1 router takes by default 3 minutes to conclude that the last host on the subnet has left a group and no host on the subnet wants to receive traffic for the group. Meanwhile, the IGMPv1 router continues forwarding the group traffic on the subnet and wastes bandwidth. On the other hand, an IGMPv2 router concludes in 3 seconds that no host on the subnet wants to receive traffic for a group, stops forwarding it on the subnet, and does not waste bandwidth.

The functions of the IGMPv2 Leave message and IGMPv2 Group-Specific Query message are explained in detail in the next section.

IGMPv2 Leave Group and Group-Specific Query Messages

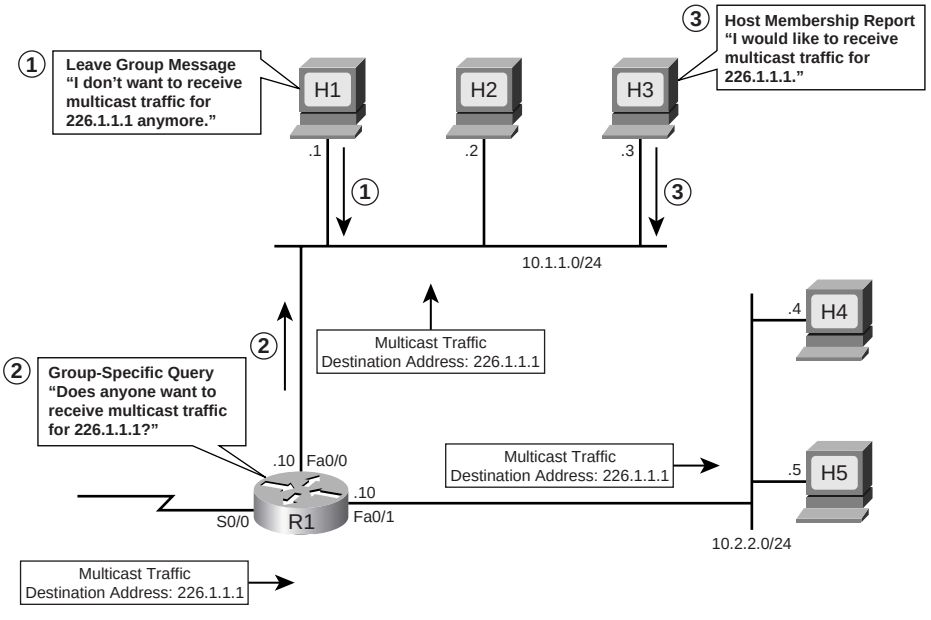
The IGMPv2 Leave Group message is used to significantly reduce the leave latency, while the IGMPv2 Group-Specific Query message prevents a router from incorrectly stopping the forwarding of packets on a LAN when a host leaves a group. As a result, both of these functions that were added for IGMPv2 work together.

> **NOTE** IGMPv2 RFC 2236 recommends that a host send a Leave Group message only if the leaving member was the last host to send a Membership Report in response to a Query. However, most IGMPv2 vendor operating systems have implemented the Leave Group processing by always sending a Leave Group message when any host leaves the group.

Figure 19-13 shows the operation of the IGMPv2 Leave process and the IGMP Group-Specific Query. In Figure 19-13, hosts H1 and H3 are currently members of group 226.1.1.1; H1 wants to leave the group.

Figure 19-13 *How Group-Specific Queries Work with the IGMPv2 Leave Process*

KEY POINT



The following three steps, referenced in Figure 19-13, describe the sequence of events for the IGMPv2 Leave mechanism when H1 leaves:

1. H1 sends an IGMPv2 Leave Group message. The destination address on the packet is 224.0.0.2, which is a well-known multicast address for All Multicast Routers to inform all routers on the subnet that, "I don't want to receive multicast traffic for 226.1.1.1 anymore."

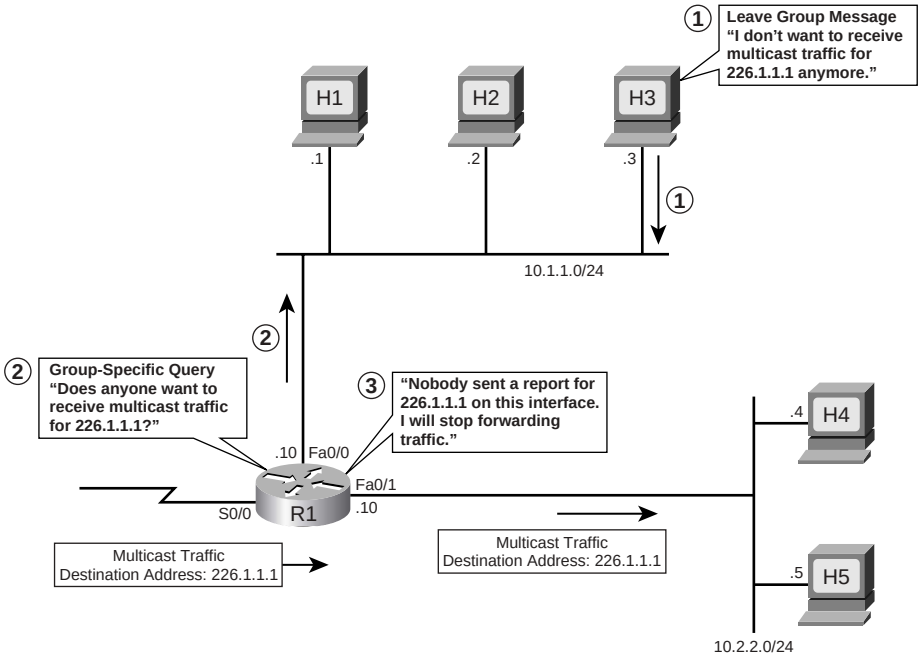
2. R1 sends a Group-Specific Query. Routers do not keep track of hosts that are members of the group, only the group memberships that are active. Because H1 has decided to leave 226.1.1.1, R1 can stop forwarding traffic for 226.1.1.1 on its fa0/0 interface if H1 is the last member of 226.1.1.1 on the interface. However, R1 needs to make sure that no other hosts off this interface still need to receive packets from group 226.1.1.1. Therefore, R1 sends a Group-Specific Query to determine whether any hosts are still members of 226.1.1.1. R1 uses 226.1.1.1 as the destination address on the packet so that only hosts that are members of this group will receive the message and respond. Through this message, R1 is asking any remaining hosts on the subnet, "Does anyone want to receive multicast traffic for 226.1.1.1?"

3. H3 sends a Membership Report. H3 is still a member of group 226.1.1.1. It hears the Group-Specific Query and responds with an IGMPv2 Membership Report to inform the routers on the subnet that it is still a member of group 226.1.1.1 and would like to keep receiving traffic for group 226.1.1.1.

NOTE The Report Suppression mechanism explained earlier for the General Group Query is also used for the Group-Specific Query.

IGMPv2 routers repeat the process of Step 2 in this example each time they receive a Leave message as shown in Step 1. In the previous example, the router (R1) did not stop sending traffic as a result of the process. In the next example, H3 is the only remaining member of group 226.1.1.1 on the subnet. Assume that now H3 also wants to leave the group, as shown in Figure 19-14.

Figure 19-14 *IGMPv2 Leave Process—No Response to the Group-Specific Query*



The following three steps, referenced in Figure 19-14, describe the sequence of events for the IGMPv2 Leave mechanism when H3 leaves:

1. H3 sends an IGMPv2 Leave Group message. The destination address on the packet is 224.0.0.2, to inform all routers on the subnet that, "I don't want to receive multicast traffic for 226.1.1.1 anymore."
2. R1 sends a Group-Specific Query. R1 receives the Leave Group message from H3. R1 sends a Group-Specific Query to determine whether any hosts are still members of group 226.1.1.1. R1 uses 226.1.1.1 as the destination address on the packet so that only hosts that are members of this group will receive the message and respond.
3. No reports are received, so R1 stops forwarding group traffic. Because there are now no remaining members of 226.1.1.1 on the subnet, R1 does not receive a response to the Group-Specific Query from any host. As a result, R1 stops forwarding multicasts for 226.1.1.1 out its fa0/1 interface.

Step 3 of this example provides a nice backdrop from which to describe the concepts of a *Last Member Query Interval* and a *Last Member Query Count*. These values determine how long it takes a router to believe that all hosts on a LAN have left a particular group. By default, routers use an MRT of 10 (1 second) for Group-Specific Queries; because a router should receive a response to a Group-Specific Query in that amount of time, the router uses the MRT value as the value of the Last Member Query Interval. So, the router uses the following process:

KEY POINT

1. Send a Group-Specific Query in response to an IGMP Leave.
2. If no Report is received within the Last Member Query Interval, repeat Step 1.
3. Repeat Step 1 the number of times defined by the value of the Last Member Query Count.

The Last Member Query Count is the number of consecutive Group-Specific Queries sent for the same group before the router concludes that there are no active members of the group on a subnet. The default value for the Last Member Query Count is 2. So, the leave latency is typically less than 3 seconds, compared to up to 3 minutes with IGMPv1.

IGMPv2 Querier

IGMPv2 defines a querier election process that is used when multiple routers are connected to a subnet. When IGMPv2 routers start, they each send an IGMPv2 General Query message to the well-known All Hosts group 224.0.0.1 using their interface address as the source address. When an IGMPv2 router receives a General Query message, it compares the source IP address of the General Query message with its own interface address. The router with the lowest IP address on the subnet is elected as the IGMP querier. The non-querier routers stop sending their queries but

monitor how frequently the querier is sending general IGMPv2 Queries. When the elected querier does not send a query for two consecutive Query Intervals plus one half of one Query Response Interval, it is considered to be dead and a new querier is elected. RFC 2236 refers to this time interval as *Other Querier Present Interval*. The default value for the Other Querier Present Interval is 255 seconds, because the default General IGMPv2 Query Interval is 125 seconds and the default Query Response Interval is 10 seconds.

IGMPv1 and IGMPv2 Interoperability

IGMPv2 is designed to be backward compatible with IGMPv1. RFC 2236 defines some special interoperability rules. The next few sections explore the following interoperability scenarios:

- **IGMPv2 Host and IGMPv1 Routers**—Defines how an IGMPv2 host should behave in the presence of an IGMPv1 router on the same subnet.
- **IGMPv1 Host and IGMPv2 Routers**—Defines how an IGMPv2 router should behave in the presence of an IGMPv1 host on the same subnet.
- **IGMPv1 and IGMPv2 Routers**—Defines how an IGMPv2 router should behave in the presence of an IGMPv1 router on the same subnet.

IGMPv2 Host and IGMPv1 Routers

When a host sends the IGMPv2 Report with the message type 0x16, which is not defined in IGMPv1, a version 1 router would consider 0x16 an invalid message type and ignore it. Therefore, a version 2 host must send IGMPv1 Reports when a version 1 router is active. But how does an IGMPv2 host detect the presence of an IGMPv1 router on the subnet?

IGMPv2 hosts determine if the querying router is an IGMPv1 or IGMPv2 host based on the value of the MRT field of the periodic general IGMP Query. In IGMPv1 Queries, this field is zero, whereas in IGMPv2 it is nonzero and represents the MRT value. When an IGMPv2 host receives an IGMPv1 Query, it knows that the IGMPv1 router is present on the subnet and marks the interface as an IGMPv1 interface. The IGMPv2 host then stops sending IGMPv2 messages.

Whenever an IGMPv2 host receives an IGMPv1 Query, it starts a 400-second *Version 1 Router Present Timeout* timer. This timer is reset whenever it receives an IGMPv1 Query. If the timer expires, which indicates that there are no IGMPv1 routers present on the subnet, the IGMPv2 host starts sending IGMPv2 messages.

IGMPv1 Host and IGMPv2 Routers

IGMPv2 routers can easily determine if any IGMPv1 hosts are present on a LAN based on whether any hosts send an IGMPv1 Report message (type 0x12) or IGMPv2 Report message (type 0x16). Like IGMPv1 routers, IGMPv2 routers send periodic IGMPv2 General Queries. An IGMPv1 host responds normally because IGMPv2 General Queries are very similar in format to IGMPv1 Queries—except for the second octet, which is ignored by IGMPv1 hosts. So, an IGMPv2 router will examine all Reports to find out if any IGMPv1 hosts exist on a LAN.

NOTE If IGMPv2 hosts are also present on the same subnet, they would send IGMPv2 Membership Reports. However, IGMPv1 hosts do not understand IGMPv2 Reports and ignore them; they do not trigger Report Suppression in IGMPv1 hosts. Therefore, sometimes an IGMPv2 router receives both an IGMPv1 Report and an IGMPv2 Report in response to a General Query.

While an IGMPv2 router knows that an IGMPv1 host is present on a LAN, the router ignores Leave messages and the Group-Specific Queries triggered by receipt of the Leave messages. This is necessary because if an IGMPv2 router responds to a Leave Group message with a Group-Specific Query, IGMPv1 hosts will not understand it and thus ignore the message. When an IGMPv2 router does not receive a response to its Group-Specific Query, it may erroneously conclude that nobody wants to receive traffic for the group and thus stop forwarding it on the subnet. So, with one (or more) IGMPv1 hosts listening for a particular group, the router essentially suspends the optimizations that reduce leave latency.

IGMPv2 routers continue to ignore Leave messages until the *IGMPv1-host-present countdown timer* expires. RFC 2236 defines that when IGMPv2 routers receive an IGMPv1 Report, they must set an IGMPv1-host-present countdown timer. The timer value should be equal to the Group Membership Interval, which defaults to 180 seconds in IGMPv1 and 260 seconds in IGMPv2. (Group Membership Interval is a time period during which if a router does not receive an IGMP Report, the router concludes that there are no more members of the group on a subnet.)

IGMPv1 and IGMPv2 Routers

RFC 2236 defines that when IGMPv1 and IGMPv2 routers are present on a subnet, a network administrator should manually configure all IGMPv2 routers to function as IGMPv1 routers.

Timers Used in IGMPv1 and IGMPv2

Table 19-4 summarizes important timers used in IGMPv1 and IGMPv2, their usage, and default values.

Table 19-4 *Important Timers Used in IGMPv1 and IGMPv2*

KEY POINT

Timer	Usage	Default Value for IGMPv1	Default Value for IGMPv2
Query Interval	A time period between General Queries sent by a router.	60 seconds	125 seconds
Query Response Interval	The maximum response time for hosts to respond to the periodic general Queries.	Fixed at 10 seconds	10 seconds , can be between .1 and 25.5 seconds
Group Membership Interval	A time period during which if a router doesn't receive an IGMP Report, the router concludes that there are no more members of the group on the subnet.	180 seconds	260 seconds
Other Querier Present Interval	A time period during which if the IGMPv2 non-querier routers don't receive an IGMP Query from the querier router, the non-querier routers conclude that the querier is dead.	—	255 seconds
Last Member Query Interval	The maximum response time inserted by IGMPv2 routers into the Group-Specific Queries and the time period between two consecutive Group-Specific Queries sent for the same group.	—	1 second
Version 1 Router Present Timeout	A time period during which if an IGMPv2 host doesn't receive an IGMPv1 Query, the IGMPv2 host concludes that there are no IGMPv1 routers present and starts sending IGMPv2 messages.	—	400 seconds

IGMP Version 3

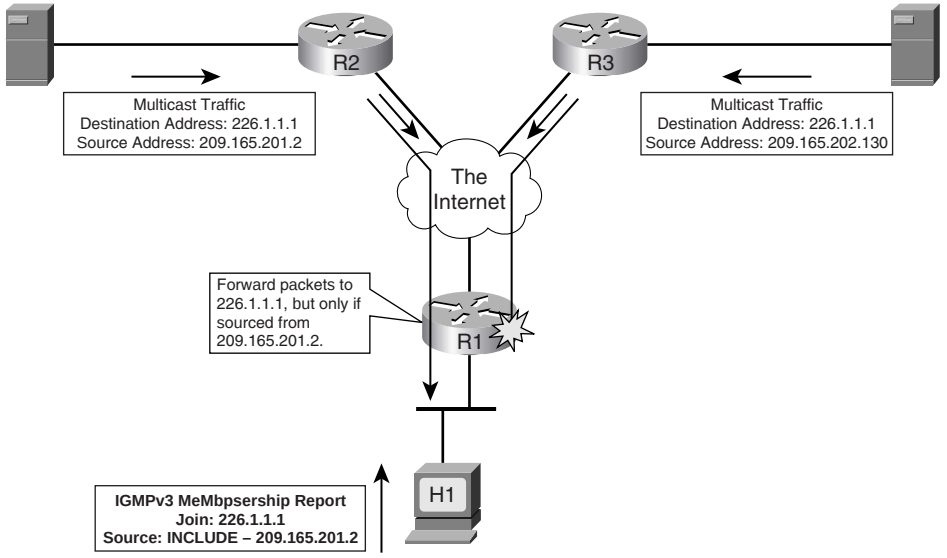
In October 2002, RFC 3376 defined specifications for IGMPv3. IGMPv3 is a major revision of the protocol and is very complex. To use the new features of IGMPv3, last-hop routers have to be updated, host operating systems have to be modified, and applications have to be specially designed and written. Unfortunately, at the time of this writing (mid-2005), a very limited number

of IGMPv3 applications are available. Therefore, this section does not examine IGMPv3 in detail; instead, it summarizes its major features.

In IGMPv1 and IGMPv2, when a host makes a request to join a group, a multicast router forwards the traffic for the group to the subnet regardless of the source IP address of the packets. For example, assume that a multimedia conference is in session. A group member decides to maliciously disturb the session by sending "bogus data or noise" by either talking or sending music to the same group. Although multimedia applications allow a user to mute any of the other members, it does not stop the unwanted traffic from being delivered to the host. If a group of hackers decides to flood a company's network with bogus high-bandwidth data using the same multicast group address that the company's employees have joined, it can create a DoS attack for the company by overwhelming low-speed links. Neither IGMPv1 nor IGMPv2 has a mechanism to prevent such an attack.

IGMPv3 allows hosts to filter incoming traffic based on the source IP addresses from which it is willing to receive packets, through a feature called *Source-Specific Multicast (SSM)*. IGMPv3 is designed to support source filtering. It allows a host to indicate interest in receiving packets only from specific source addresses, or from all but specific source addresses, sent to a particular multicast address. Figure 19-15 shows basic operation of the IGMPv3 Membership Report process.

Figure 19-15 *IGMPv3 Membership Report*



In Figure 19-15, the multicast traffic for the group 226.1.1.1 is available from two sources. R1 receives traffic from both the sources. H1 prepares an IGMPv3 Membership Report using the destination address 224.0.0.22, specially assigned by IANA for the IGMPv3 Membership Report. The message type is 0x22 (defined in RFC 3376), with a note "Source–INCLUDE—209.165.201.2," which means, "I would like to join multicast group 226.1.1.1, but only if the group traffic is coming from the source 209.165.201.2."

How does a host learn group source addresses? A lot of work remains to be done by application designers to develop SSM applications. Cisco has designed URL Rendezvous Directory (URD), and IGMP v3lite to use the new features of IGMPv3 until IGMPv3 applications are available and operating systems are updated. A detailed discussion of URD and IGMP v3lite is beyond the scope of this book. IGMPv3 is compatible with IGMPv1 and IGMPv2.

NOTE The following URL provides more information on URD and IGMP v3lite:
http://www.cisco.com/en/US/products/sw/iosswrel/ps1834/products_feature_guide09186a008008048a.html

Comparison of IGMPv1, IGMPv2, and IGMPv3

Table 19-5 compares the important features of IGMPv1, IGMPv2, and IGMPv3.

Table 19-5 *Comparison of IGMPv1, IGMPv2, and IGMPv3*

KEY POINT

Feature	IGMPv1	IGMPv2	IGMPv3
First Octet Value for the Query Message	0x11	0x11	0x11
Group Address for the General Query	0.0.0.0	0.0.0.0	0.0.0.0
Destination Address for the General Query	224.0.0.1	224.0.0.1	224.0.0.1
Default Query Interval	60 seconds	125 seconds	125 seconds
First Octet Value for the Report	0x12	0x16	0x22
Group Address for the Report	Joining multicast group address	Joining multicast group address	Joining multicast group address and source address
Destination Address for the Report	Joining multicast group address	Joining multicast group address	224.0.0.22
Is Report Suppression Mechanism Available?	Yes	Yes	No
Can Maximum Response Time Be Configured?	No, fixed at 10 seconds	Yes, 0 to 25.5 seconds	Yes, 0 to 53 minutes
Can a Host Send a Leave Group Message?	No	Yes	Yes
Destination Address for the Leave Group Message	—	224.0.0.2	224.0.0.22

continues

Table 19-5 *Comparison of IGMPv1, IGMPv2, and IGMPv3 (Continued)*

Feature	IGMPv1	IGMPv2	IGMPv3
Can a router send a Group-Specific Query?	No	Yes	Yes
Can a Host Send Source- and Group-Specific Reports?	No	No	Yes
Can a Router Send Source- and Group-Specific Queries?	No	No	Yes
Rule for Electing a Querier	None—depends on multicast routing protocol	Router with the lowest IP address on the subnet	Router with the lowest IP address on the subnet
Compatible with Other Versions of IGMP?	No	Yes, only with IGMPv1	Yes, with both IGMPv1 and IGMPv2

Multicast Listener Discovery Protocol

RFC 2710 defines specifications for the Multicast Listener Discovery (MLD) protocol. MLD is derived from IGMPv2 and is designed for IPv6. The operation of MLD is similar to IGMPv2. The major differences between IGMPv2 and MLD are as follows:

- All the multicast devices on a subnet use a special IPv6 link-local address as their source address in their communication to other multicast devices. The use of the link-local source address prevents the MLD packet from traveling beyond the local link.
- In MLD, when a host wants to leave a group, it sends a Done message. The Done message is similar to the IGMPv2 Leave message. It is addressed to the all-routers IPv6 link-local scope address, FF02::2.
- In MLD, the router Queries are called Multicast Listener Queries. The General Queries are addressed to the all-nodes IPv6 link-local scope address, FF02::1. When a router receives a Done message, it sends a Multicast-Address-Specific Query. Its function is similar to IGMPv2 Group-Specific Query.

LAN Multicast Optimizations

This final major section of this chapter introduces the basics of three tools that optimize the flow of multicast over a LAN. Specifically, this section covers the following topics:

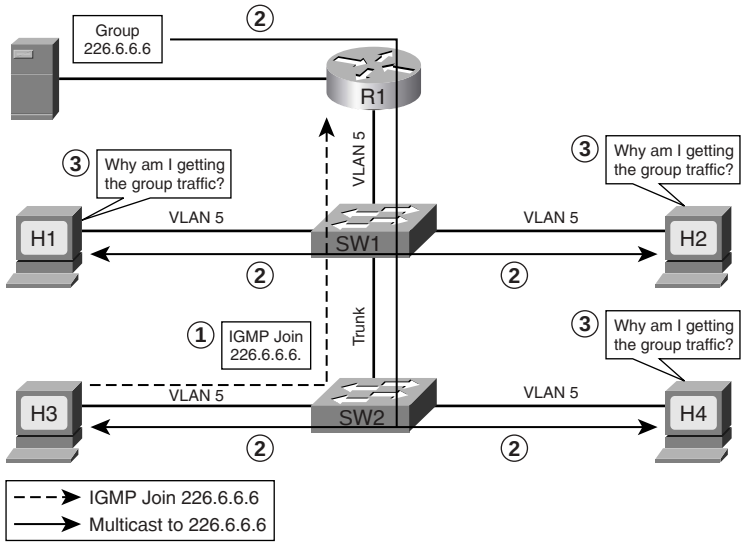
- Cisco Group Management Protocol (CGMP)

- IGMP snooping
- Router-Port Group Management Protocol (RGMP)

Cisco Group Management Protocol

IGMP helps routers to determine how to distribute multicast traffic. However, IGMP works at Layer 3, and switches do not understand IGMP messages. Switches, by default, flood multicast traffic to all the hosts in a broadcast domain, which wastes bandwidth. Figure 19-16 illustrates the problem.

Figure 19-16 *Switches Flood Multicast Traffic*



Hosts H1, H2, H3, H4, and R1 are all in the same broadcast domain of VLAN 5. The following three steps, referenced in Figure 19-16, describe the sequence of events when H3 sends an IGMP Join message:

1. H3 sends an IGMP Join message for group 226.6.6.6.
2. R1 forwards the group traffic to SW1. The destination MAC address on the frame is 0x0100.5e06.0606. SW1 cannot find this address in its CAM table because it is never used by any device as a source address. Therefore, SW1 starts forwarding the group traffic to H1, H2, and SW2 because the group traffic is for VLAN 5. Similarly, SW2 starts forwarding the group traffic to H3 and H4.
3. All the hosts, H1 to H4, receive the group traffic, but only H3 requested it. H3 requested the group traffic and has started receiving it. However, H1, H2, and H4 did not ask for the group traffic and they are flooded by switches with the group traffic.

In this illustration, only four hosts are shown in the broadcast domain of VLAN 5. What happens if a broadcast domain is flat and has hundreds of users? If a single host joins a multicast group, all

the hosts would be flooded with the group traffic whether they have requested the group traffic or not. The goal of multicasting is to deliver the group traffic to only those hosts that have requested it and maximize the use of bandwidth.

There are two popular methods for helping Layer 2 switches determine how to distribute the multicast traffic to hosts:

- CGMP, which is Cisco proprietary and discussed throughout the rest of this section.
- IGMP snooping, discussed in the next section.

CGMP, a Layer 2 protocol, is configured on both a Cisco router and switches and permits the router to communicate Layer 2 information it has learned from IGMP to switches. A multicast router knows the MAC addresses of the multicast hosts, and the groups to which they listen, based on IGMP communication with hosts. The goal of CGMP is to enable the router to communicate this information through CGMP messages to switches so that switches can dynamically modify their CAM table entries. Only the routers produce CGMP messages, while switches only listen to the CGMP messages. The destination address on the CGMP messages is always the well-known CGMP multicast MAC address 0x0100.0cdd.dddd. The use of the multicast destination MAC address on the CGMP messages forces switches to flood the message through all the ports so that all the switches in a network receive the CGMP messages. The important information in the CGMP messages is one or more pairs of MAC addresses:

- Group Destination Address (GDA)
- Unicast Source Address (USA)

The following five steps describe the general process of CGMP. Later, these steps are explained using a detailed example.

1. When a CGMP-capable router gets connected to the switch, it sends a CGMP Join message with the GDA set to zero and the USA set to its own MAC address. The CGMP-capable switch now knows that a multicast router is connected to the port on which it received the router's CGMP message. The router repeats the message every 60 seconds. A router can also tell the switch that it no longer participates in CGMP by sending a CGMP Leave message with the GDA set to zero and the USA set to its own MAC address.
2. When a host joins a group, it sends an IGMP Join message. Normally, a multicast router examines only Layer 3 information in the IGMP Join message and the router does not have to process any Layer 2 information. However, when CGMP is configured on a router, the router also examines the Layer 2 destination and source MAC addresses of the IGMP Join message. The source address is the unicast MAC address of the host that sent the IGMP Join message. The router then generates a CGMP Join message that includes the multicast MAC address associated with the multicast IP address (to the GDA field of the CGMP join) and the

unicast MAC address of the host (to the USA field of the CGMP message). The router sends the CGMP Join message using the well-known CGMP multicast MAC address 0x0100.0cdd.dddd as the destination address.

3. When switches receive a CGMP Join message, they search in their CAM tables for the port number associated with the host MAC address listed in the USA field. Switches create a new CAM table entry (or use an existing entry if it was already created before) for the multicast MAC address listed in the GDA field of the CGMP Join message, add the port number associated with the host MAC address listed in the USA field to the entry, and forward the group traffic on the port.

4. When a host leaves a group, it sends an IGMP Leave message. The router learns the host's unicast MAC address (USA) and the IP multicast group it has just left. Because the Leave messages are sent to the All Multicast Routers MAC address 0x0100.5e00.0002 and not to the multicast group address the host has just left, the router calculates the multicast MAC address (GDA) from the IP multicast group the host has just left. The router then generates a CGMP Leave message, copies the multicast MAC address it has just calculated in the GDA field and unicast MAC address in the USA field of the CGMP Leave message, and sends it to the well-known CGMP multicast MAC address.

5. When switches receive a CGMP Leave message, they again search for the port number associated with the host MAC address listed in the USA field. Switches remove this port from the CAM table entry for the multicast MAC address listed in the GDA field of the CGMP Leave message and stop forwarding the group traffic on the port.

Thus, CGMP helps switches send group traffic to only those hosts that want it, which helps to avoid waste bandwidth.

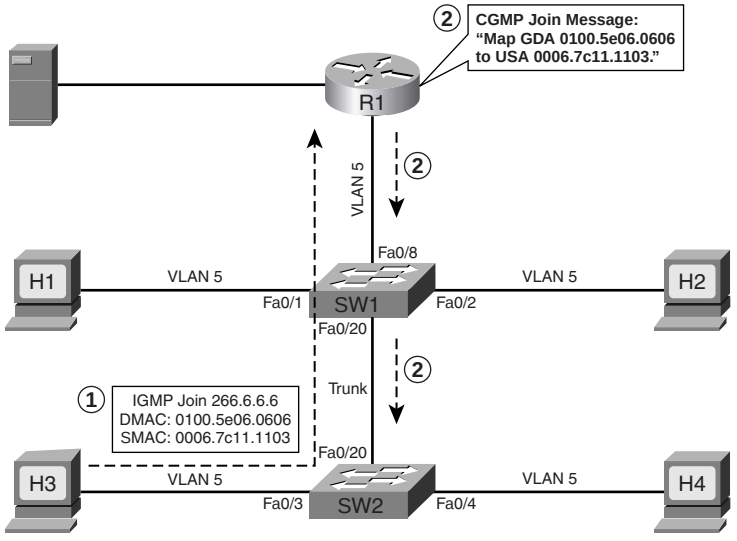
Figure 19-17, 19-18, and 19-19 show a complete example of how routers and switches use CGMP in response to a host joining and then leaving a group. Figure 19-17 begins the example by showing a router's reaction to an IGMP Report, which is to send an CGMP Join to the switches on a LAN. The following two steps, referenced in Figure 19-17, describe the sequence of events when H3 sends an IGMP Join message:

1. H3 sends an IGMP Join message for 226.6.6.6. At Layer 2, H3 uses 0x0100.5e06.0606 (the multicast MAC address associated with 226.6.6.6) as the destination address of a frame, and its own BIA 0x0006.7c11.1103 as the source MAC address.

2. R1 generates a CGMP Join message. When a CGMP-capable router receives an IGMP Join message, it generates a Layer 2 CGMP Join message. The destination address on the frame is the well-known multicast MAC address 0x0100.0cdd.dddd, which is understood only by Cisco switches but is forwarded by all switches. R1 sets the GDA to the group MAC address 0x0100.5e06.0606 and sets the USA to H3's MAC address 0x0006.7c11.1103, which communicates to switches that "A host with the USA 0x0006.7c11.1103 has requested

multicast traffic for the GDA 0x0100.5e06.0606, so map your CAM tables accordingly." This message is received by both switches.

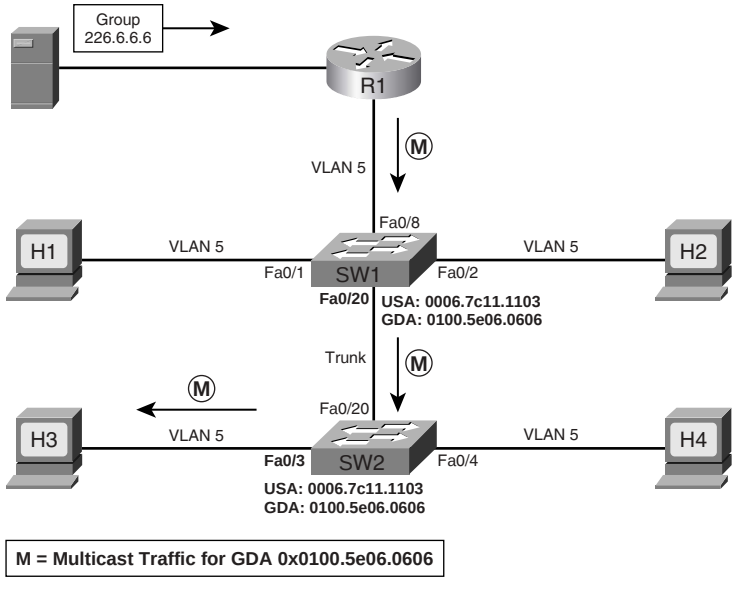
Figure 19-17 *CGMP Join Message Process*



SW1 and SW2 search their CAM table entries and find that a host with the USA 0x0006.7c11.1103 is located on their port number fa0/20 and fa0/3, respectively. Figure 19-18 shows that SW1 and SW2 have mapped the GDA 0x0100.5e06.0606 to their port numbers fa0/20 and fa0/3, respectively.

Figure 19-18 *Switches Map GDA to Port Numbers and Don't Flood All the Hosts in a Broadcast Domain*

KEY POINT

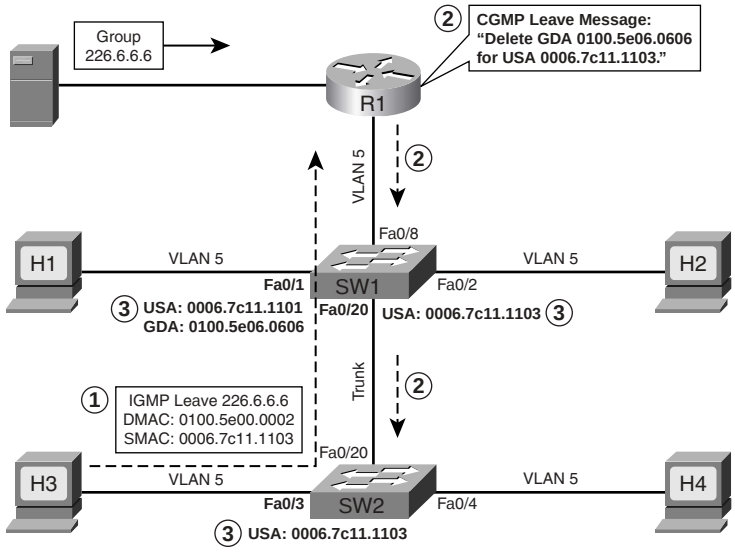


When R1 forwards multicast traffic with GDA 0x0100.5e06.0606 to SW1, as shown in Figure 19-18, SW1 searches its CAM table and notices that this traffic should be forwarded only on port fa0/20. Therefore, only SW2 receives the group traffic. Similarly, SW2 searches its CAM table and forwards the group traffic only on its port fa0/3, and only H3 receives the group traffic.

CGMP optimizes the forwarding of IGMP traffic as well. Although not shown in the figures, assume that H1 sends an IGMP Join message for 226.6.6.6. R1 will send another CGMP Join message, and SW1 will add the GDA 0x0100.5e06.0606 to its port fa0/1 also. When a router sends IGMP General Queries, switches forward them to host members who have joined any group, for example, H1 and H3. When hosts send IGMP Reports, switches forward them to the members of the group and the router.

The final step of the example, shown in Figure 19-19, demonstrates what happens when H3 leaves the group. Note that for this example, H1 has also joined the same multicast group.

Figure 19-19 *CGMP Leave Message Process*



The following three steps, referenced in Figure 19-19, describe the sequence of events when H3 sends an IGMP Leave message:

1. H3 sends an IGMP Leave message for 226.6.6.6. At Layer 2, H3 uses the All Multicast Routers MAC address 0x0100.5e00.0002 as the destination address and its own BIA 0x0006.7c11.1103 as the source address.

2. R1 generates a CGMP Leave message. When a CGMP-capable router receives an IGMP Leave message, it generates a Layer 2 CGMP Leave message. The destination address on the frame is the well-known multicast MAC address 0x0100.0cdd.dddd. R1 calculates the group

MAC address 0x0100.5e06.0606 from the Layer 3 address 226.6.6.6 and sets the GDA to that value. It sets the USA to H3's MAC unicast MAC address of 0x0006.7c11.1103. This Leave message communicates to switches that, "A host with the USA 0x0006.7c11.1103 does not want to receive multicast traffic for GDA 0x0100.5e06.0606, so update your CAM tables accordingly." This message is received by both switches.

3. Switches update their CAM table entries. SW1 and SW2 search their CAM table entries and find that a host with the USA 0x0006.7c11.1103 is located on their port numbers fa0/20 and fa0/3, respectively. Figure 19-20 shows that SW1 and SW2 have removed the GDA 0x0100.5e06.0606 from their port numbers fa0/20 and fa0/3, respectively.

H1 is still a member of the group 266.6.6.6, so R1 keeps forwarding the traffic with GDA 0x0100.5e06.0606 to SW1, as shown in Figure 19-19. SW1 searches its CAM table and finds that this traffic should be forwarded only on port fa0/1. Therefore, only H1 receives the group traffic.

Continuing the example further, now assume that H1 sends an IGMP Leave message for 226.6.6.6. R1 will send a Group-Specific Query for 226.6.6.6. Because no host is currently a member of this group, R1 does not receive any IGMP Membership Reports for the group. R1 sends the CGMP Leave message with the GDA set to the group MAC address and the USA set to 0. This message communicates to switches that, "No hosts are interested in receiving the multicast group traffic for the MAC address 0x0100.5e06.0606, so remove all the CAM table entries for this group."

Table 19-6 summarizes the possible combinations of the GDA and the USA in CGMP messages and the meanings of each. The first five messages have been discussed.

Table 19-6 *CGMP Messages*

KEY POINT

Type	Group Destination Address	Unicast Source Address	Meaning
Join	Group MAC	Host MAC	Add USA port to group
Leave	Group MAC	Host MAC	Delete USA port from group
Join	Zero	Router MAC	Learn which port connects to the CGMP router
Leave	Zero	Router MAC	Release CGMP router port
Leave	Group MAC	Zero	Delete the group from the CAM
Leave	Zero	Zero	Delete all groups from the CAM

The last Leave message in Table 19-6, Delete All Groups, is used by the router for special maintenance functions. For example, when the **clear ip cgmp** command is entered at the router for clearing all the CGMP entries on the switches, the router sends the CGMP Leave message with GDA set to zero and USA set to zero. When switches receive this message, they delete all group entries from the CAM tables.

IGMP Snooping

What happens if your network has non-Cisco switches? You cannot use CGMP because it is Cisco proprietary. IGMP snooping can be used for a multivendor switched network to control distribution of multicast traffic at Layer 2. IGMP snooping requires the switch software to eavesdrop on the IGMP conversation between multicast hosts and the router. The switch examines IGMP messages and learns the location of multicast routers and group members.

NOTE Many Cisco switches support IGMP snooping, including the 3550 switches used in the CCIE Routing and Switching lab exam.

The following three steps describe the general process of IGMP snooping. Later, these steps are explained in detail.

KEY POINT

1. To detect whether multiple routers are connected to the same subnet, Cisco switches listen to the following routing protocol messages to determine on which ports routers are connected:
   - IGMP General Query message with GDA 01-00-5e-00-00-01
   - OSPF messages with GDA 01-00-5e-00-00-05 or 01-00-5e-00-00-06
   - Protocol Independent Multicast (PIM) version 1 and Hot Standby Routing Protocol (HSRP) Hello messages with GDA 01-00-5e-00-00-02
   - PIMv2 Hello messages with GDA 01-00-5e-00-00-od
   - Distance Vector Multicast Routing Protocol (DVMRP) Probe messages with GDA 01-00-5e-00-00-04

   As soon as the switch detects router ports in a VLAN, they are added to the port list of all GDAs in that VLAN.
2. When the switch receives an IGMP Report on a port, its CPU looks at the GDA, creates an entry in the CAM table for the GDA, and adds the port to the entry. The router port is also added to the entry. The group traffic is now forwarded on this port and the router port. If other hosts send their IGMP Reports, the switch adds their ports to the group entry in the CAM table and forwards the group traffic on these ports.
3. Similarly, when the switch receives an IGMP Leave message on a port, its CPU looks at the GDA, removes the port from the group entry in the CAM table, and does not forward the group traffic on the port. The switch checks whether this is the last nonrouter port for the GDA. If it is not the last nonrouter port for the GDA, which means there is at least one host in the VLAN that wants the group traffic, the switch discards the Leave message; otherwise, it sends the Leave message to the router.

Thus, IGMP snooping helps switches send group traffic to only those hosts that want it and helps to avoid wasted bandwidth.

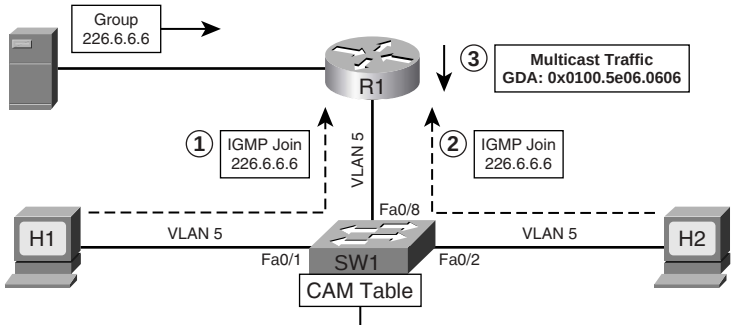
For efficient operations, IGMP snooping requires hardware filtering support in a switch so that it can differentiate between IGMP Reports and actual multicast traffic. The switch CPU needs to see IGMP Report messages (and Multicast Routing Protocol messages) because the IGMP snooping process requires the CPU. However, the forwarding of multicast frames does not require the CPU, instead requiring only a switch's forwarding ASICs. Older switches, particularly those that have no Layer 3 awareness, could not identify a packet as IGMP; these switches would have overburdened their CPUs by having to send all multicasts to the CPU. Most of today's more modern switches support enough Layer 3 awareness to recognize IGMP, so that IGMP snooping does not overburden the CPU.

NOTE CGMP was a popular Cisco switch feature in years past because IGMP implementations on some switches would have required too much work. Today, many of the Cisco current switch product offerings do not even support CGMP, in deference to IGMP snooping.

Figure 19-20 shows an example of the IGMP snooping process.

Figure 19-20 *Joining a Group Using IGMP Snooping and CAM Table Entries*

KEY POINT



Layer 3-Aware Switch

	Destination Address	Type of Packet	Ports
1	01-00-5e-06-06-06	**L3-IGMP**	0
	01-00-5e-06-06-06	**Non-IGMP**	**1, 8**
2	01-00-5e-06-06-06	**L3-IGMP**	0
	01-00-5e-06-06-06	**Non-IGMP**	**1, 2, 8**

The following three steps, referenced in Figure 19-20, describe the sequence of events when H1 and H2 send IGMP Join messages:

1. H1 sends an IGMP Join message for 226.6.6.6. At Layer 2, H1 uses the multicast MAC address 0x0100.5e06.0606 (the MAC for group 226.6.6.6) as the destination address and uses its own BIA 0x0006.7c11.1101 as the source address. SW1 receives the packet on its

fa0/1 port and, noticing that it is an IGMP packet, forwards the packet to the switch CPU. The CPU uses the information to set up a multicast forwarding table entry, as shown in the CAM table that includes the port numbers 0 for CPU, 1 for H1, and 8 for R1. Notice that the CAM table lists two entries for the same destination MAC address 0x0100.5e06.0606—one for the IGMP frames for port 0, and the other for the non-IGMP frames for ports 1 and 8. The CPU of the switch instructs the switching engine to not forward any non-IGMP frames to port 0, which is connected to the CPU.

2. H2 sends an IGMP Join message for 226.6.6.6. At Layer 2, H2 uses the multicast MAC address 0x0100.5e06.0606 as the destination address and uses its own BIA 0x0006.7c11.1102 as the source address. SW1 receives the packet on its fa0/2 port, and its switching engine examines the packet. The process of analyzing the packet, as described in Step 1, is repeated and the CAM table entries are updated as shown.

3. Router R1 forwards the group traffic. R1 is receiving multicast traffic for group 226.6.6.6 and starts forwarding the traffic to SW1. SW1 starts receiving the multicast traffic on its port fa0/8. The switching engine would examine the packet and determine that this is a non-IGMP packet, search its CAM table, and determine that it should forward the packet on ports fa0/1 and fa0/2.

Compared to CGMP, IGMP snooping is less efficient in maintaining group information. In Figure 19-21, when R1 periodically sends IGMP General Queries to the All Hosts group, 224.0.0.1 (GDA 0x0100.5e00.0001), SW1 intercepts the General Queries and forwards them through all ports in VLAN 5. In CGMP, due to communication from the router through CGMP messages, the switch knows exactly on which ports multicast hosts are connected and, therefore, forwards IGMP General Queries only on those ports. Also, in IGMP snooping, when hosts send IGMP Reports, the switch must intercept them to maintain GDA information in the CAM table. As a result, the hosts do not receive each other's IGMP Report, which breaks the Report Suppression mechanism and forces each host to send an IGMP Report. However, the switch sends only one IGMP Report per group to the router. In CGMP, the switch does not have to intercept IGMP Reports, because maintaining group information in the switch is not dependent on examining IGMP packets from hosts; instead, the switch uses CGMP messages from the router.

Figure 19-21 shows the Leave process for IGMP snooping.

The following three steps, referenced in Figure 19-21, describe the sequence of events when H1 and H2 send IGMP Leave messages:

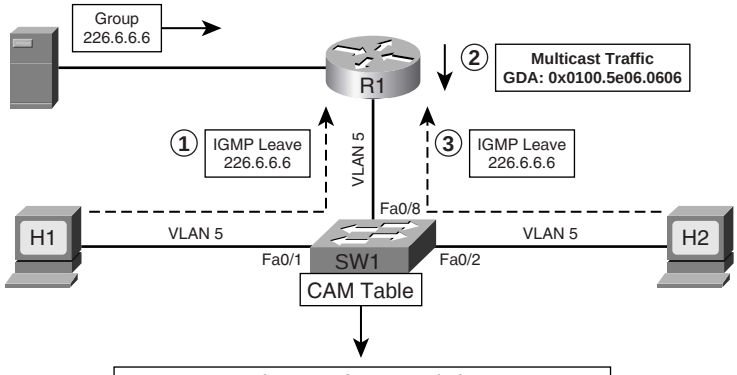
1. H1 sends an IGMP Leave message for 226.6.6.6, but SW1 does not forward it to router R1 in this case. At Layer 2, H1 uses the All Multicast Routers MAC address 0x0100.5e00.0002 as the destination address and uses its own BIA 0x0006.7c11.1101 as the source address. SW1 captures the IGMP Leave message on its fa0/1 port and its switching engine examines the packet. The switch sends an IGMP General Query on port fa0/1 to determine whether there are any other hosts that are members of this group on the port. (This feature was designed to protect other hosts if they are connected to the same switch port using a hub.) If an IGMP

Report is received on port fa0/1, the switch discards the Leave message received from H1. Because, in this example, there is only one host connected to port fa0/1, the switch does not receive any IGMP Report and deletes the port fa0/1 from the CAM table entry, as shown in Figure 19-22. H2 connected with port fa0/2 is still a member of the group and its port number is in the CAM table entry. Hence, SW1 does not forward the IGMP Leave message to the router.

2. Router R1 continues forwarding the group traffic. R1 continues forwarding multicast traffic for group 226.6.6.6 to SW1 because R1 did not even know that H1 left the group. Based on the updated CAM table entry for the group shown in Figure 19-22, SW1 now forwards this traffic only on port fa0/2.

3. H2 sends an IGMP Leave message for 226.6.6.6, and SW1 does forward it to router R1 in this case. At Layer 2, H2 uses the All Multicast Routers MAC address 0x0100.5e00.0002 as the destination address and uses its own BIA 0x0006.7c11.1102 as the source address. Again, SW1 captures the IGMP Leave message on its fa0/2 port and its switching engine examines the packet. The switch sends an IGMP General Query on port fa0/2 to determine whether there are any other hosts that are members of this group on the port. Because, in this example, there is only one host connected to port fa0/2, the switch does not receive any IGMP Report and deletes the port fa0/2 from the CAM table entry. After SW1 deletes the port, it realizes that this was the last nonrouter port for the CAM table entry for Ox0100.5e06.0606. Therefore, SW1 deletes the CAM table entry for this group, as shown in Figure 19-22, and forwards the IGMP Leave message to R1. R1 sends an IGMP Group-Specific Query and, when no hosts respond, stops forwarding traffic for 226.6.6.6 toward SW1.

Figure 19-21 *Leaving a Group Using IGMP Snooping and CAM Table Entries*



Layer 3-Aware Switch

	Destination Address	Type of Packet	Ports
1	01-00-5e-06-06-06	**L3-IGMP**	0
	01-00-5e-06-06-06	**Non-IGMP**	**2, 8**
3	01-00-5e-06-06-06	**L3-IGMP**	0

IGMP snooping becomes more complicated when multiple multicast routers are used and many LAN switches are interconnected via high-speed trunks. Also, CGMP and IGMP snooping control distribution of multicast traffic only on ports where hosts are connected. They do not provide any control mechanism for ports where routers are connected. The next section briefly examines how Router-Port Group Management Protocol (RGMP) helps switches control distribution of multicast traffic on ports where routers are connected.

Router-Port Group Management Protocol

RGMP is a Layer 2 protocol that enables a router to communicate to a switch which multicast group traffic the router does and does not want to receive from the switch. By being able to restrict the multicast destinations that a switch forwards to a router, a router can reduce its overhead. In fact, RGMP was designed to help routers reduce overhead when they are attached to high-speed LAN backbones.

Although RGMP is Cisco proprietary, oddly enough it cannot work concurrently with Cisco-proprietary CGMP. When RGMP is enabled on a router or a switch, CGMP is silently disabled; if CGMP is enabled on a router or a switch, RGMP is silently disabled. Note also that while it is proprietary, RGMP is published as informational RFC 3488.

RGMP works well in conjunction with IGMP snooping. In fact, IGMP snooping would typically learn the ports of all multicast routers by listening for IGMP and multicast routing protocol traffic. In some cases, some routers may not want all multicast traffic, so RGMP provides a means to reduce the unwanted traffic. The subtle key to the need for RGMP when using IGMP snooping is to realize this important fact about IGMP snooping:

> IGMP snooping helps switches control distribution of multicast traffic on ports where multicast hosts are connected, but it does not help switches control distribution of multicast traffic on ports where multicast routers are connected.

For example, consider the simple network shown in Figure 19-22. SW2 has learned of routers R3 and R4 with IGMP snooping, so it forwards multicasts sent to all multicast groups out to both R3 and R4.

As you can see from Figure 19-22, R3 really needs to receive traffic only for group A, and R4 needs to receive traffic only for group B. However, IGMP snooping causes the switch to forward all multicast packets to each router. To combat that problem, RGMP can be used by a router to tell the switch to only forward packets for particular multicast groups. For example, Figure 19-23 shows the same network as Figure 19-22, but with RGMP snooping. In this case, RGMP Join messages are enabled in both the routers and the switch, with the results shown in Figure 19-23.

Figure 19-22 *IGMP Snooping Without RGMP*

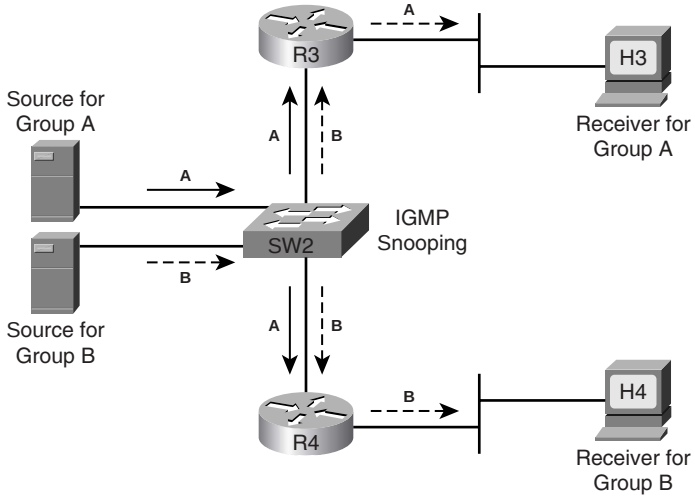


Figure 19-23 *More Efficient Forwarding with RGMP Added to IGMP Snooping*

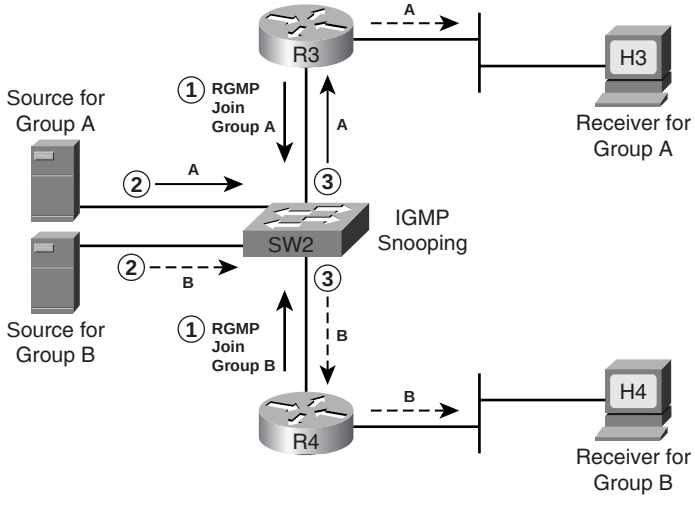


Figure 19-23 shows the following three main steps, with the first step really showing the RGMP function with the RGMP Join message. The Join message allows a router to identify the groups for which the router wants to receive traffic.

1. R3 sends an RGMP Join for group A, and R4 sends an RGMP Join for group B. As a result, SW2 knows to forward multicasts for group A only to R3, and for group B only to R4.

2. The sources send a packet to groups A and B, respectively.
3. SW2 forwards the traffic for group A only to R3, and the packets for group B only to R4.

While Figure 19-23 shows just one example and one type of RGMP message, RGMP includes four different messages. All the RGMP messages are generated by a router and are sent to the multicast IP address 224.0.0.25. The following list describes the four RGMP messages:

KEY POINT

- When RGMP is enabled on a router, the router sends RGMP Hello messages by default every 30 seconds. When the switch receives an RGMP Hello message, it stops forwarding all multicast traffic on the port on which it received the Hello message.
- When the router wants to receive traffic for a specific multicast group, the router sends an RGMP Join *G* message, where *G* is the multicast group address, to the switch. When the switch receives an RGMP Join message, it starts forwarding the requested group traffic on the port on which it received the Hello message.
- When the router does not want to receive traffic for a formerly RGMP-joined specific multicast group, the router sends an RGMP Leave *G* message, where *G* is the multicast group address, to the switch. When the switch receives an RGMP Leave message, it stops forwarding the group traffic on the port on which it received the Hello message.
- When RGMP is disabled on the router, the router sends an RGMP Bye message to the switch. When the switch receives an RGMP Bye message, it starts forwarding all IP multicast traffic on the port on which it received the Hello message.

NOTE The following URL provides more information on RGMP:
http://www.cisco.com/en/US/products/hw/switches/ps700/products_tech_note09186a008011c11b.shtml

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most Cisco Press *Exam Certification Guides*, this book does not repeat information listed in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 19-7 lists some of the key protocols and facts regarding IGMP.

Table 19-7 *Protocols and Standards for Chapter 19*

Name	Standard
GLOP Addressing in 233/8	RFC 2770
Administratively Scoped IP Multicast	RFC 2365
IGMP version 0	RFC 988
Host Extensions for IP Multicasting [IGMPv1]	RFC 1112
Internet Group Management Protocol, Version 2	RFC 2236
Internet Group Management Protocol, Version 3	RFC 3376
Multicast Listener Discovery (MLD) for IPv6	RFC 2710
Cisco Systems Router-Port Group Management Protocol (RGMP)	RFC 3488

Configuring multicasting on a Cisco router is relatively easy. You must first configure a multicast routing protocol on a Cisco router. The multicast routing protocols are covered in the next chapter, which also presents all the important configuration commands in the "Foundation Summary" section.

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

multicasting, multicast address range, multicast address structure, permanent multicast group, source-specific addresses, GLOP addressing, administratively scoped addresses, transient multicast group, multicast MAC address, joining a group, IGMP, IGMPv1 Host Membership Query, MRT, IGMPv1 Host Membership Report, Report Suppression mechanism, IGMPv2 Host Membership Query, IGMPv2 Leave, IGMPv2 Group-Specific Query, IGMPv2 Host Membership Report, SSM, IGMPv3 Host Membership Query, IGMPv3 Host Membership Report, querier election, MLD, CGMP, IGMP snooping, RGMP

Further Reading

Beau Williamson, *Developing IP Multicast Networks*, Volume I, Cisco Press, 2000.

References in This Chapter

- Beau Williamson, *Developing IP Multicast Networks*, Volume I, Cisco Press, 2000 (Chapter 3):
  - IGMPv1 Querier, page 61
  - IGMP Version 2, page 64
  - IGMPv1–IGMPv2 Interoperability, pages 73–76
- Cisco Systems, Inc.:
  - Multicast in a Campus Network: CGMP and IGMP Snooping (Document ID 10559), http://www.cisco.com/en/US/products/hw/switches/ps700/products_tech_note09186a00800b0871.shtml#snooping_ov
  - Router-Port Group Management Protocol, http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120limit/120s/120s10/dtrgmp.htm

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- IP Multicast
  - — Distribution Trees
  - — PIM-SM Mechanics
  - — Rendezvous Points
  - — RPF

CHAPTER 20

IP Multicast Routing

In Chapter 19, "Introduction to IP Multicasting," you learned how a multicast router communicates with hosts and then decides whether to forward or stop the multicast traffic on a subnet. But how does a multicast router receive the group traffic? How is the multicast traffic forwarded from a source so that all the group users receive it? This chapter provides answers to those questions.

This chapter first defines the multicast routing problem by identifying the difference between unicast and multicast routing. It then provides an overview of the basic design concepts of multicast routing protocols, and shows how they solve multicast routing problems. Next, the chapter covers the operations of the Protocol Independent Multicast routing protocol in dense mode (PIM-DM) and sparse mode (PIM-SM). The chapter also covers the basic functions of Distance Vector Multicast Routing Protocol (DVMRP) and Multicast OSPF (MOSPF).

"Do I Know This Already?" Quiz

Table 20-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 20-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Multicast Routing Basics	1	
Dense-Mode Routing Protocols	2–4	
Sparse-Mode Routing Protocols	5–8	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

4. When a multicast router receives a multicast packet, which one of the following tasks will it perform first?

   a. Examine the IP multicast destination address on the packet, consult the multicast routing table to determine the next-hop address, and forward the packet through appropriate interface(s).

   b. Depending on the multicast routing protocol configured, either forward the packet on all the interfaces or forward the packet on selected interfaces except the one on which the packet was received.

   c. Determine the interface this router would use to send packets to the source of the packet, and decide whether the packet arrived in that interface or not.

   d. Send a Prune message to its upstream neighbor if it does not have any directly connected group members or active downstream routers.

5. A PIM router receives a PIM Assert message on a LAN interface. Which of the following statements is (are) true about the response of the router?

   a. The router does not have to take any action.

   b. If the router is configured with the PIM-DM routing protocol, it will process the Assert message; otherwise, it will ignore it.

   c. If the router is configured with the PIM-SM routing protocol, it will process the Assert message; otherwise, it will ignore it.

   d. The router will send a PIM Assert message.

6. When a PIM-DM router receives a Graft message from a downstream router after it has sent a Prune message to its upstream router for the same group, which of the following statements is (are) true about its response?

   a. It will send a Graft message to the downstream router and a Prune message to the upstream router.

   b. It will send a Prune message to the downstream router and a Graft message to the upstream router.

   c. It will re-establish adjacency with the upstream router.

   d. It will send a Graft message to the upstream router.

7. On router R1, the **show ip mroute 239.5.130.24** command displays **Serial2, Prune/Dense, 00:01:34/00:01:26** for the (S, G) entry under the outgoing interface list. Which of the following statements provide correct interpretation of this information?

   a. Router R1 has sent a Prune message on its Serial2 interface to its upstream router 1 minute and 34 seconds ago.

   b. Router R1 will send a Graft message on its Serial2 interface to its upstream router after 1 minute and 26 seconds.

   c. Router R1 received a Prune message on its Serial2 interface from its downstream router 1 minute and 34 seconds ago.

   d. Router R1 will send a Prune message on its Serial2 interface to its upstream router after 1 minute and 26 seconds.

   e. Router R1 will forward the traffic for the group on its Serial2 interface after 1 minute and 26 seconds.

8. From the following statements, select the true statement(s) regarding when a PIM-SM RP router will send the unicast PIM Register-Stop messages to the first-hop DR.

   a. If the RP has no need for the traffic

   b. If the RP is already receiving traffic on the shared tree

   c. When the RP begins receiving multicast traffic via SPT from the source

   d. When the RP sends multicast traffic via SPT to the downstream router

9. R1, a PIM-SM router, sends an (S,G) RP-bit Prune to its upstream neighbor. Assume that all the PIM-SM routers in the network are using the Cisco default **spt-threshold** value. Which of the following statements is (are) true about the status of different routers in the PIM-SM network at this time?

   a. At R1, the root-path tree and shortest-path tree diverge.

   b. R1 is switching over from shortest-path tree to root-path tree.

   c. R1 is switching over from root-path tree to shortest-path tree.

   d. At R1, the RPF neighbor for the (S,G) entry is different from the RPF neighbor of the (*, G) entry.

10. In a PIM-SM LAN network using Auto-RP, one of the routers is configured to send Cisco-RP-Announce and Cisco-RP-Discovery messages. All the routers show all the interfaces with correct PIM neighbors in sparse mode. However, the network administrator is puzzled by inconsistent RP mapping information shown on many routers. Some routers show correct RP mappings, but many leaf routers do not show any RP mappings. Which of the following statements represent(s) the most likely cause(s) for the above problem?

- **a.** The links between the leaf routers and the mapping agent are congested.
- **b.** All the interfaces of all the routers are configured with the command **ip pim sparse-mode**.
- **c.** The leaf routers are configured with a static RP address using an **override** option.
- **d.** The RPF check on the leaf routers is failing.

11. PIM-SM router R1 has two interfaces listed, s0/0 and fa0/0, in its (*,G) entry for group 227.7.7.7 in its multicast routing table. Assuming nothing changes in that (*,G) entry in the next 10 minutes, which of the following could be true?

- **a.** R1 is sending PIM Join messages toward the RP.
- **b.** R1 does not need to send Join messages toward the RP as long as the RP is continuing to forward multicasts for group 227.7.7.7 to R1.
- **c.** R1 is receiving PIM Join messages periodically on one or both of interfaces s0/0 and fa0/0.
- **d.** R1 is receiving IGMP Report messages periodically on interface fa0/0.
- **e.** The RP has been sending PIM Prune messages to R1 periodically, but R1 has been replying with PIM Reject messages because it still needs to receive the packets.

Foundation Topics

Multicast Routing Basics

The main function of any routing protocol is to help routers forward a packet in the right direction, causing the packet to keep moving closer to its desired destination, ultimately reaching its destination. To forward a unicast packet, a router examines the packet's destination address, finds the next-hop address from the unicast routing table, and forwards the packet through the appropriate interface. A unicast packet is forwarded along a single path from the source to the destination.

The top part of Figure 20-1 shows how a router can easily make a decision about forwarding a unicast packet by consulting its unicast routing table. However, when a router receives a multicast packet, as shown at the bottom of Figure 20-1, it cannot forward the packet because multicast IP addresses are not listed in the unicast routing table. Also, routers often have to forward multicast packets out multiple interfaces to reach all receivers. These requirements make the multicast forwarding process more complex than unicast forwarding.

Figure 20-1 *Multicast Routing Problem*

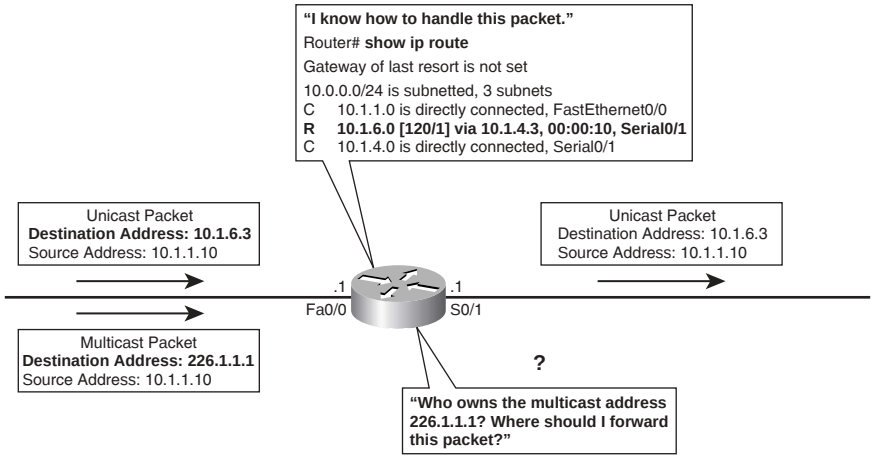


Figure 20-1 shows that the router has received a multicast packet with the destination address 226.1.1.1. The destination address represents a dynamically changing group of recipients, not any one recipient's address. How can the router find out where these users are? Where should the router forward this packet?

An analogy may help you to understand better the difficulty of multicast routing. Assume that you want to send party invitations through the mail, but instead of creating dozens of invitations, you

create only one. Before mailing the invitation, you put a destination address on it, "This envelope contains my party invitation," and then drop it in a mailbox. When the postal system examines the destination address on your envelope, where should it deliver your envelope? And because it is only one invitation, does the postal system need to make copies? Also, how can the postal system figure out to which addresses to deliver the copies? By contrast, if IP multicast were the post office, it would know who you want to invite to the party, know where they are located, and make copies of the invitation and deliver them all to the correct addresses.

The next few sections discuss solutions for forwarding multicast traffic and controlling the distribution of multicast traffic in a routed network.

Overview of Multicast Routing Protocols

Routers can forward a multicast packet by using either a *dense-mode multicast routing protocol* or a *sparse-mode multicast routing protocol*. This section examines the basic concepts of multicast forwarding using dense mode, the reverse-path-forwarding (RPF) check, and multicast forwarding using sparse mode, all of which help to solve the multicast routing problem.

Multicast Forwarding Using Dense Mode

Dense-mode routing protocols assume that the multicast group application is so popular that every subnet in the network has at least one receiver wanting to receive the group traffic. Therefore, the design of a dense-mode routing protocol instructs the router to forward the multicast traffic on all the configured interfaces, with some exceptions to prevent looping. For example, a multicast packet is never forwarded out the interface on which it was received. Figure 20-2 shows how a dense-mode routing protocol receives a multicast on one interface, and then forwards copies out all other interfaces.

Figure 20-2 *R1 Forwarding a Multicast Packet Using a Dense-Mode Routing Protocol*

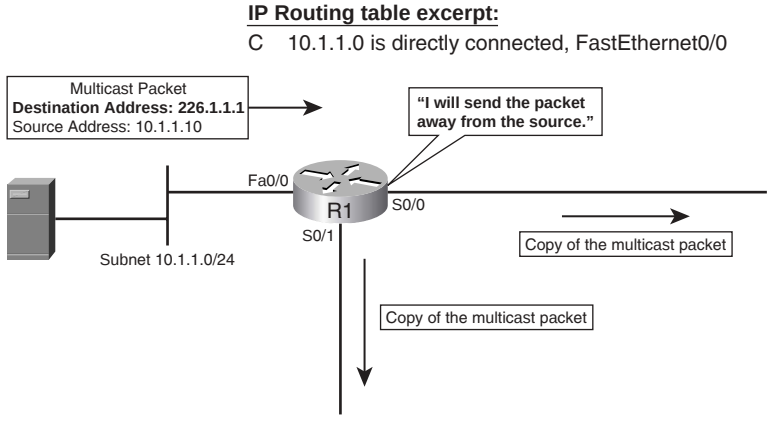


Figure 20-2 shows the dense-mode logic on R1, with R1 *flooding* copies of the packet out all interfaces except the one on which the packet was received. Although Figure 20-2 shows only one router, other routers can receive these multicasts and repeat the same process. All subnets will receive a copy of the original multicast packet.

Dense-mode protocols assume that all subnets need to receive a copy of the packets; however, dense-mode protocols do allow routers to ask to not receive traffic sent to a particular multicast group. Dense-mode routers typically do not want to receive multicast packets for a particular group if both of the following are true:

KEY POINT

- The router does not have any active downstream routers that need packets for that group.
- The router does not know of any hosts on directly connected subnets that have joined that group.

When both of these conditions are true, the router needs to inform its upstream router not to send traffic for the group, which it does by using a special message called a Prune message. The mechanics of how dense-mode routers communicate with each other is discussed in detail under the PIM-DM section later in this chapter.

DVMRP, PIM-DM, and MOSPF are the dense-mode routing protocols discussed in this chapter, with most of the attention being paid to PIM-DM.

Reverse-Path-Forwarding Check

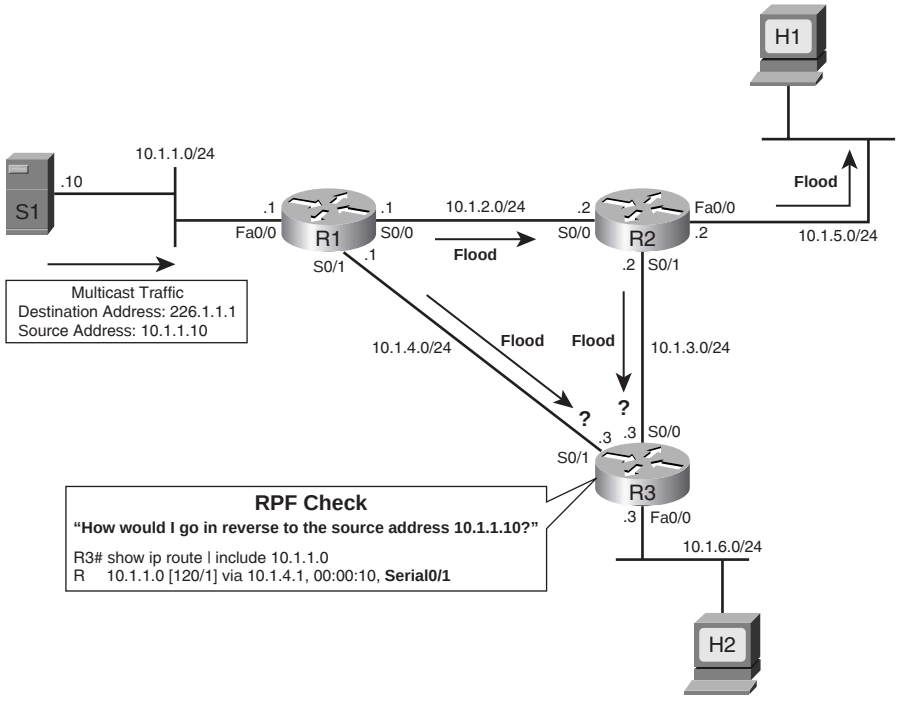
Routers cannot simply use logic by which they receive a multicast packet and then forward a copy of it out all other interfaces, without causing multicast packets to loop around the internetwork. To prevent such loops, routers do not forward multicasts out the same interface on which they were received. Multicast routers use a *reverse-path-forwarding (RPF) check* to prevent loops. The RPF check adds this additional step to a dense-mode router's forwarding logic:

KEY POINT

Look at the source IP address of the multicast packet. If my route that matches the source lists an outgoing interface that is the actual interface on which the packet was received, the packet passes the RPF check. If not, do not replicate and forward the packet.

Figure 20-3 shows an example in which R3 uses the RPF check on two separate copies of the same original multicast packet. Host S1 sends a multicast packet, with R1 flooding it to R2 and R3. R2 receives its copy, and floods it as well. As a result, R3 receives the same packet from two routers: on its s0/0 interface from R2 and on its s0/1 interface from R1. Without the RPF check, R3 would forward the packet it got from R1 to R2, and vice versa, and begin the process of looping packets. With this same logic, R1 and R2 also keep repeating the process. This duplication creates multicast routing loops and generates multicast storms that waste bandwidth and router resources.

Figure 20-3 *R3 Performs the RPF Check*



A multicast router does not forward any multicast packet unless the packet passes the RPF check. In Figure 20-3, R3 has to decide whether it should accept the multicast packets coming from R1 and R2. R3 makes this decision by performing the RPF check, described in detail as follows:

1. R3 examines the source address of each incoming multicast packet, which is 10.1.1.10. The source address is used in the RPF check of Step 2.
2. R3 determines the reverse path interface based on its route used to forward packets to 10.1.1.10. In this case, R3's route to 10.1.1.0/24 is matched, and it lists an outgoing interface of s0/1, making s0/1 R3's RPF interface for IP address 10.1.1.10.
3. R3 compares the reverse path interface determined in Step 2 with the interface on which the multicast packet arrived. If they match, it accepts the packet and forwards it; otherwise, it drops the packet. In this case, R3 floods the packet received on s0/1 from R1, but it ignores the packet received on s0/0 from R2.

The RPF check implements a strategy by which routers accept packets that arrive over the shortest path, and discard those that arrive over longer routes. Multicast routing protocols cannot use the destination address to help routers forward a packet, because that address represents the group traffic. So, multicast routing protocols use the RPF check to determine whether the packet arrived at the router using the shortest-path route from the source to the router. If it did, multicast routing

protocols accept the packet and forward it; otherwise, they drop the packet and thereby avoid routing loops and duplication.

Different multicast routing protocols determine their RPF interfaces in different ways, as follows:

- Distance Vector Multicast Routing Protocol (DVMRP) maintains a separate multicast routing table and uses it for the RPF check.
- Protocol Independent Multicast (PIM) and Core-Based Tree (CBT) generally use the unicast routing table for the RPF check, as shown in Figure 20-3.
- PIM and CBT can also use the DVMRP route table, the Multiprotocol Border Gateway Protocol (MBGP) route table, or statically configured multicast route(s) for the RPF check.
- Multicast OSPF does not use the RPF check, because it computes both forward and reverse shortest-path source-rooted trees by using the Dijkstra algorithm.

Multicast Forwarding Using Sparse Mode

A dense-mode routing protocol is useful when a multicast application is so popular that you need to deliver the group traffic to almost all the subnets of a network. However, if the group users are located on a few subnets, a dense-mode routing protocol will still flood the traffic in the entire internetwork, wasting bandwidth and resources of routers. In those cases, a sparse-mode routing protocol, such as PIM-SM, could be used to help reduce waste of network resources.

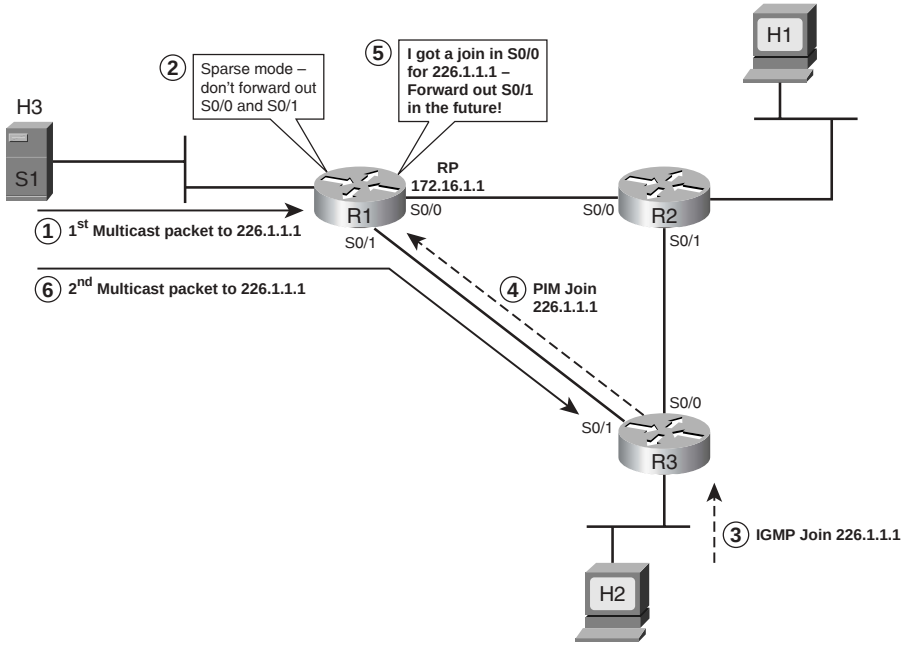
The fundamental difference between dense-mode and sparse-mode routing protocols relates to their default behavior. By default, dense-mode protocols keep forwarding the group traffic unless a downstream router sends a message stating that it does not want that traffic. Sparse-mode protocols do not forward the group traffic to any other router unless it receives a message from that router requesting copies of packets sent to a particular multicast group. A downstream router requests to receive the packets only for one of two reasons:

KEY POINT

- The router has received a request to receive the packets from some downstream router.
- A host on a directly connected host has sent an IGMP Join message for that group.

Figure 20-4 shows an example of what must happen with PIM-SM before a host (H2 in this case) can receive packets sent by host S1 to multicast group address 226.1.1.1. The PIM sparse-mode operation begins with the packet being forwarded to a special router called the *rendezvous point (RP)*. Once the group traffic arrives at an RP, unlike the dense-mode design, the RP does not automatically forward the group traffic to any router; the group traffic must be specifically requested by a router.

Figure 20-4 *R1 Forwarding a Multicast Packet Using a Sparse-Mode Routing Protocol*



NOTE Throughout this chapter, the solid arrowed lines in the figures represent multicast packets, with dashed arrowed lines representing PIM and IGMP messages.

Before you look at the numbered steps in Figure 20-4, consider the state of this internetwork. PIM-SM is configured on all the routers, R1 is selected as an RP, and in all three routers, the IP address 172.16.1.1 of R1 is configured statically as the RP address. Usually, a loopback interface address is used as an RP address and the loopback network is advertised in the unicast routing protocol so that all the routers learn how to locate an RP. At this point, R1, as the RP, may receive multicast packets sent to 226.1.1.1, but it will not forward them.

The following list describes the steps shown in Figure 20-4:

1. Host S1 sends a multicast to the RP, with destination address 226.1.1.1.
2. R1 chooses to ignore the packet, because no routers or local hosts have told the RP (R1) that they want to receive copies of multicast packets.
3. Host H2 sends an IGMP Join message for group 226.1.1.1.
4. R3 sends a PIM Join message to the RP (R1) for address 26.1.1.1.
5. R1's logic now changes, so future packets sent to 226.1.1.1 will be forwarded by R1 out s0/1 to R3.
6. Host S1 sends a multicast packet to 226.1.1.1, and R1 forwards it out s0/1 to R3.

In a PIM-SM network, it is critical for all the routers to somehow learn the IP address of an RP. One option in a small network is to statically configure the IP address of an RP in every router. Later in the chapter, the section "Dynamically Finding RPs and Using Redundant RPs" covers how routers can dynamically discover the IP address of the RP.

The example in Figure 20-4 shows some of the savings in using a sparse-mode protocol like PIM-SM. R2 has not received any IGMP Join messages on its LAN interface, so it does not send any request to the RP to forward the group traffic. As a result, R1 does not waste link bandwidth on the link from R1 to R2. R3 will not forward multicasts to R2 either in this case.

NOTE In Figure 20-4, R3 first performs its RPF check by using the IP address of the RP rather than the IP address of the source of the packet, because it is receiving the group traffic from the RP. If the RPF check succeeds, R3 forwards the traffic on its LAN.

Multicast Scoping

Multicast scoping confines the forwarding of multicast traffic to a group of routers, for administrative, security, or policy reasons. In other words, multicast scoping is the practice of defining boundaries that determine how far multicast traffic will travel in your network. The following sections discuss two methods of multicast scoping:

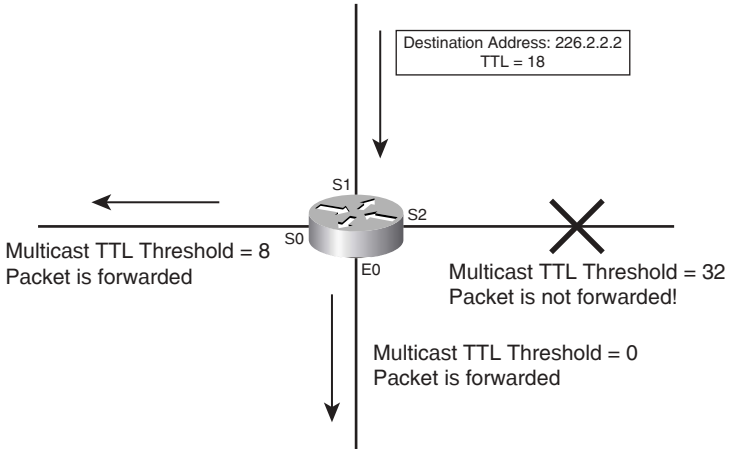
- TTL scoping
- Administrative scoping

TTL Scoping

With TTL scoping, routers compare the TTL value on a multicast packet with a configured TTL value on each outgoing interface. A router forwards the multicast packet only on those interfaces whose configured TTL value is less than or equal to the TTL value of the multicast packet. In effect, TTL scoping resets the TTL value at which the router discards multicasts from the usual value of 0 to some higher number. Figure 20-5 shows an example of a multicast router with various TTL threshold values configured on its interfaces.

In Figure 20-5, a multicast packet arrives on the s1 interface with a TTL of 18. The router decreases the packet's TTL by 1 to 17. Assume that the router is configured with a dense-mode routing protocol on all four interfaces and the RPF check succeeds—in other words, the router will want to forward a copy of the packet on each interface. The router compares the remaining TTL of the packet, which is now 17, with the TTL threshold of each outgoing interface. If the packet's TTL is higher than or equal to the interface TTL, it forwards a copy of the packet on that interface; otherwise, it does not forward it. On a Cisco router, the default TTL value on all the interfaces is 0.

Figure 20-5 *Multicast Scoping Using TTL Thresholds*



On the s0 and s2 interfaces in Figure 20-5, the network administrator has configured the TTL as 8 and 32, respectively. A copy of the packet is forwarded on the s0 and e0 interfaces because their TTL thresholds are less than 17. However, the packet is not forwarded on the s2 interface because its TTL threshold is 32, which is higher than 17.

TTL scoping has some weaknesses. First, it is difficult to implement in a large and complex network, because estimating correct TTL thresholds on many routers and many interfaces so that the network correctly confines only the intended sessions becomes an extremely demanding task. Another problem with TTL scoping is that a configured TTL threshold value on an interface applies to all multicast packets. If you want flexibility for some multicast sessions, you have to manipulate the applications to alter the TTL values when packets leave the servers.

Administrative Scoping

Recall from Chapter 19 that administratively scoped multicast addresses are private addresses in the range 239.0.0.0 to 239.255.255.255. They can be used to set administrative boundaries to limit the forwarding of multicast traffic outside of a domain. It requires manual configuration. You can configure and apply a filter on a router's interface so that multicast traffic with group addresses in the private address range is not allowed to enter or exit the interface.

NOTE This chapter assumes that you have read Chapter 19 or are thoroughly familiar with the operation of IGMP; if neither is true, read Chapter 19 before continuing with this chapter.

Dense-Mode Routing Protocols

There are three dense-mode routing protocols:

- Protocol Independent Multicast Dense Mode (PIM-DM)

- Distance Vector Multicast Routing Protocol (DVMRP)
- Multicast Open Shortest Path First (MOSPF)

This section covers the operation of PIM-DM in detail and provides an overview of DVMRP and MOSPF.

Operation of Protocol Independent Multicast Dense Mode

Protocol Independent Multicast (PIM) defines a series of protocol messages and rules by which routers can provide efficient forwarding of multicast IP packets. PIM previously existed as a Cisco-proprietary protocol, although it has been offered as an experimental protocol via RFCs 2362, 3446, and 3973. The PIM specifications spell out the rules mentioned in the earlier examples in this chapter—things like the RPF check, the PIM dense-mode logic of flooding multicasts until routers send Prune messages, and the PIM Sparse-mode logic of not forwarding multicasts anywhere until a router sends a Join message. This section describes the PIM-DM protocols in more detail.

PIM gets its name from its ability to use the unicast IP routing table for its RPF check—independent of whatever unicast IP routing protocol(s) was used to build the unicast routing table entries. In fact, the name "PIM" really says as much about the two other dense-mode protocols—DVMRP and MOSPF—as it does about PIM. These other two protocols do not use the unicast IP routing table for their RPF checks, instead building their own independent tables. PIM simply relies on the unicast IP routing table, independent of which unicast IP routing protocol built a particular entry in the routing table.

Forming PIM Adjacencies Using PIM Hello Messages

KEY POINT PIM routers form adjacencies with neighboring PIM routers for the same general reasons, and with the same general mechanisms, as many other routing protocols. PIMv2, the current version of PIM, sends Hello messages every 30 seconds (default) on every interface on which PIM is configured. By receiving Hellos on the same interface, routers discover neighbors, establish adjacency, and maintain adjacency. PIMv2 Hellos use IP protocol number 103 and reserved multicast destination address 224.0.0.13, called the All-PIM-Routers multicast address. The Hello messages contain a Holdtime value, typically three times the sender's PIM Hello interval. If the receiver does not receive a Hello message from the sender during the Holdtime period, it considers the sending neighbor to be dead.

NOTE The older version, PIMv1, does not use Hellos, instead using a PIM Query message. PIMv1 messages are encapsulated in IP packets with protocol number 2 and use the multicast destination address 224.0.0.2.

As you will see in the following sections, establishing and maintaining adjacencies with directly connected neighbors is very important for the operation of PIM. A PIM router sends other PIM messages only on interfaces on which it has known active PIM neighbors.

Source-Based Distribution Trees

Dense-mode routing protocols are suitable for dense topology in which there are many multicast group members relative to the total number of hosts in a network. When a PIM-DM router receives a multicast packet, it first performs the RPF check. If the RPF check succeeds, the router forwards a copy of the packet to all the PIM neighbors except the one on which it received the packet. Each PIM-DM router repeats the process and floods the entire network with the group traffic. Ultimately, the packets are flooded to all leaf routers that have no downstream PIM neighbors.

The logic described in the previous paragraph actually describes the concepts behind what PIM calls a *source-based distribution tree*. It is also sometimes called a *shortest-path tree (SPT)*, or simply a *source tree*. The tree defines a path between the source host that originates the multicast packets and all subnets that need to receive a copy of the multicasts sent by that host. The tree uses the source as the root, the routers as the nodes in the tree, and the subnets connected to the routers as the branches and leaves of the tree. Figure 20-3, earlier in the chapter, shows the concept behind an SPT.

The configuration required on the three routers in Figure 20-3 is easy—just add the global command **ip multicast-routing** on each router and the interface command **ip pim dense-mode** on all the interfaces of all the routers.

KEY POINT PIM-DM might have a different source-based distribution tree for each combination of source and multicast group, because the SPT will differ based on the location of the source and the locations of the hosts listening for each multicast group address. The notation (S,G) refers to a particular SPT, or to an individual router's part of a particular SPT, where S is the source's IP address and G is the multicast group address. For example, the (S,G) notation for the example in Figure 20-3 would be written as (10.1.1.10, 226.1.1.1).

Example 20-1 shows part of the (S,G) SPT entry on R3, from Figure 20-3, for the (10.1.1.0, 226.1.1.1) SPT. Host S1 is sending packets to 226.1.1.1, and host H2 sends an IGMP Join message for the group 226.1.1.1. Example 20-1 shows a part of R3's multicast routing table, as displayed using the **show ip mroute** command.

Example 20-1 *Multicast Route Table Entry for the Group 226.1.1.1 for R3*

```
(10.1.1.10/32, 226.1.1.1), 00:00:12/00:02:48, flags: CT
 Incoming interface: Serial0/1, RPF nbr 10.1.4.1
 Outgoing interface list:
   FastEthernet0/0, Forward/Dense, 00:00:12/00:00:00
```

The interpretation of the information shown in Example 20-1 is as follows:

- The first line shows that the (S, G) entry for (10.1.1.10/32, 226.1.1.1) has been up for 12 seconds, and that if R3 does not forward an (S, G) packet in 2 minutes and 48 seconds, it will expire. Every time R3 forwards a packet using this entry, the timer is reset to 3 minutes.
- The C flag indicates that R3 has a directly connected group member for 226.1.1.1. The T flag indicates that the (S, G) traffic is forwarded on the shortest-path tree.
- The incoming interface for the group 226.1.1.1 is s0/1 and the RPF neighbor (the next-hop IP address to go in the reverse direction toward the source address 10.1.1.10) is 10.1.4.1.
- The group traffic is forwarded out on the fa0/0 interface. This interface has been in the forwarding state for 12 seconds. The second timer is listed as 00:00:00, because it cannot expire with PIM-DM, as this interface will continue to forward traffic until pruned.

NOTE The multicast routing table flags mentioned in this list, as well as others, are summarized in Table 20-6 in the "Foundation Summary" section of this chapter.

The next two sections show how PIM-DM routers use information learned from IGMP to dynamically expand and contract the source-based distribution trees to satisfy the needs of the group users.

NOTE According to PIM-DM specifications, multicast route tables only need (S,G) entries. However, for each (S,G) entry, a Cisco router creates a (*,G) entry as a parent entry, for design efficiency. The (*,G) entry is not used for forwarding the multicast traffic for a group that uses PIM-DM. Therefore, for simplicity and clarity, the (*,G) entries are not shown in the examples that use PIM-DM. Had you built the same network as illustrated in Figure 20-3, and configured PIM-DM, the (*,G) entries would also be listed in the **show ip mroute** command output.

Prune Message

PIM-DM creates a new SPT when a source first sends multicast packets to a new multicast group address. The SPT includes all interfaces except RPF interfaces, because PIM-DM assumes that all hosts need to receive a copy of each multicast packet. However, some subnets may not need a copy of the multicasts, so PIM-DM defines a process by which routers can remove interfaces from an SPT by using PIM Prune messages.

For example, in Figure 20-3, hosts H1 and H2 need a copy of the multicast packets sent to 226.1.1.1. However, as shown, when R2 gets the multicast from R1, R2 then forwards the multicasts to R3. As it turns out, R3 is dropping the packets for the group traffic from 10.1.1.1, sent to 226.1.1.1, because those packets fail R3's RPF check. In this case, R3 can cause R2 to remove its s0/1 interface from its outgoing interface list for (10.1.1.10, 226.1.1.1) by sending a

Prune message to R2. As a result, R2 will not forward the multicasts to R3, thereby reducing the amount of wasted bandwidth.

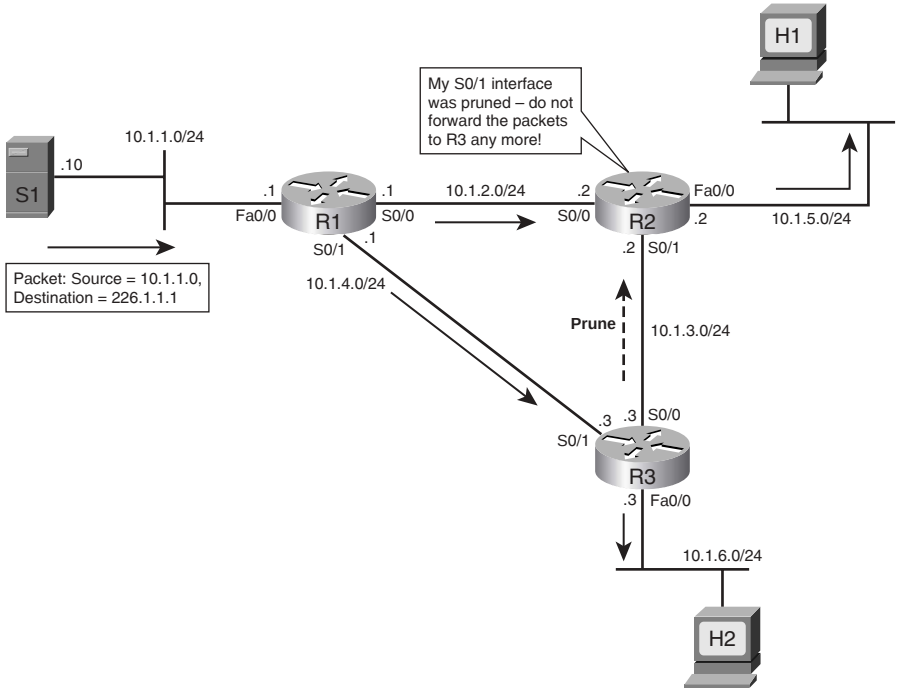
NOTE The term *outgoing interface list* refers to the list of interfaces in a forwarding state, listed for an entry in a router's multicast routing table.

The following is a more formal definition of a PIM Prune message:

KEY POINT The PIM Prune message is sent by one router to a second router to cause the second router to remove the link on which the Prune is received from a particular (S,G) SPT.

Figure 20-6 shows the same internetwork and example as Figure 20-3, but with R3's Prune messages sent to R2.

Figure 20-6 *R3 Sends a Prune Message to R2*



As a result of the Prune message from R3 to R2, R2 will prune its s0/1 interface from the SPT for (10.1.1.10,226.1.1.1). Example 20-2 shows the multicast route table entry for R2 in Figure 20-6, with the line that shows the pruned state highlighted.

Example 20-2 *Multicast Route Table Entry for the Group 226.1.1.1 for R2*

```
(10.1.1.10/32, 226.1.1.1), 00:00:14/00:02:46, flags: CT
 Incoming interface: Serial0/0, RPF nbr 10.1.2.1
 Outgoing interface list:
   FastEthernet0/0, Forward/Dense, 00:00:14/00:00:00
   Serial0/1, Prune/Dense, 00:00:08/00:02:52
```

Most of the information shown in Example 20-2 is similar to the information shown in Example 20-1. Notice the Serial0/1 information shown under the outgoing interface list. It shows that this interface was pruned 8 seconds ago because R3 sent a Prune message to R2. This means that, at this time, R2 is not forwarding traffic for 226.1.1.1 on its s0/1 interface.

Because PIM-DM's inherent tendency is to flood traffic through an internetwork, the pruned s0/1 interface listed in example 20-2 will be changed back to a forwarding state after 2 minutes and 52 seconds. In PIM-DM, when a router receives a Prune message on an interface, it starts a (default) 3-minute Prune timer, counting down to 0. When the Prune timer expires, the router changes the interface to a forwarding state again. If the downstream router does not want the traffic, it can again send a Prune message. This feature keeps a downstream router aware that the group traffic is available on a particular interface from the upstream neighbor.

NOTE PIMv2 offers a better solution to maintaining the pruned state of an interface, using State Refresh messages. These messages are covered later in the chapter, in the section "Steady-State Operation and the State Refresh Message."

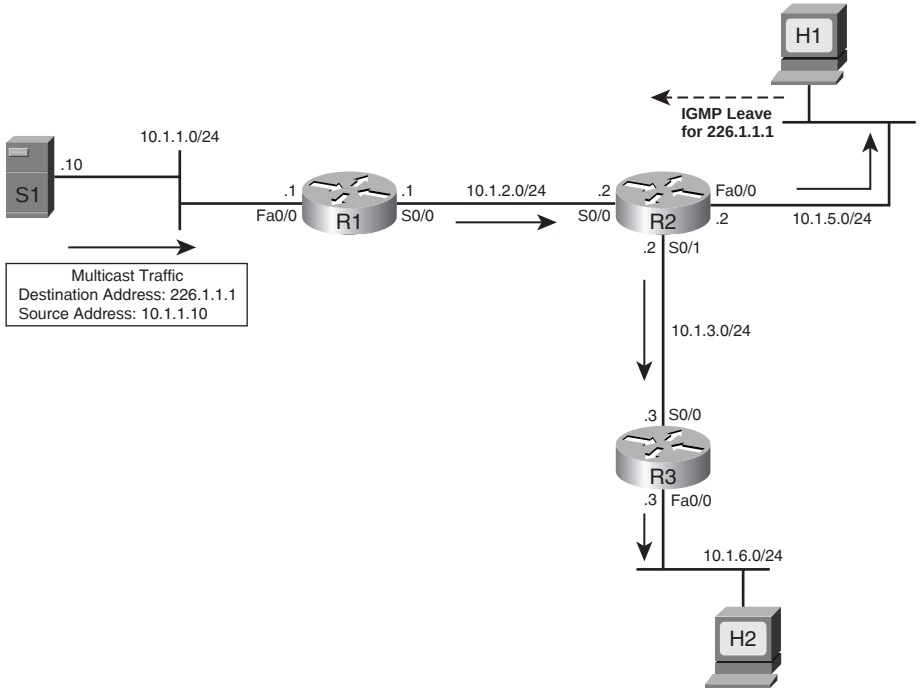
Note that a multicast router can have more than one interface in the outgoing interface list, but it can have only one interface in the incoming interface list. The only interface in which a router will receive and process multicasts from a particular source is the RPF interface. Routers still perform an RPF check, with the incoming interface information in the beginning of the **show ip mroute** output stating the RPF interface and neighbor.

PIM-DM: Reacting to a Failed Link

When links fail, or any other changes affect the unicast IP routing table, PIM-DM needs to update the RPF interfaces based on the new unicast IP routing table. Because the RPF interface may change, (S,G) entries may also need to list different interfaces in the outgoing interface list. This section describes an example of how PIM-DM reacts.

Figure 20-7 shows an example in which the link between R1 and R3, originally illustrated in Figure 20-6, has failed. After the unicast routing protocol converges, R3 needs to update its RPF neighbor IP address from 10.1.4.1 (R1) to 10.1.3.2 (R2). Also in this case, H1 has issued an IGMP Leave message.

Figure 20-7 *Direct Link Between R1 and R3 Is Down and Host H1 Sends an IGMP Leave Message*



Example 20-3 shows the resulting multicast route table entry for R3 in Figure 20-7. Note that the RPF interface and neighbor IP address has changed to point to R2.

Example 20-3 *Multicast Route Table Entry for the Group 226.1.1.1 for R3*

```
(10.1.1.10/32, 226.1.1.1), 00:02:16/00:01:36, flags: CT
  Incoming interface: Serial0/0, RPF nbr 10.1.3.2
  Outgoing interface list:
     FastEthernet0/0, Forward/Dense, 00:02:16/00:00:00
```

Example 20-3 shows how R3's view of the (10.1.1.10,226.1.1.1) SPT has changed. However, R2 had pruned its s0/1 interface from that SPT, as shown in Figure 20-6. So, R2 needs to change its s0/1 interface back to a forwarding state for SPT (10.1.1.10, 226.1.1.1). Example 20-4 shows the resulting multicast route table entry for (10.1.1.10, 226.1.1.1) in R2.

Example 20-4 *Multicast Route Table Entry for the Group 226.1.1.1 for R2*

```
(10.1.1.10/32, 226.1.1.1), 00:03:14/00:02:38, flags: T
  Incoming interface: Serial0/0, RPF nbr 10.1.2.1
  Outgoing interface list:
  Serial0/1, Forward/Dense, 00:02:28/00:00:00
```

NOTE R2 changed its s0/1 to a forwarding state because of a PIM Graft message sent by R3. The upcoming section "Graft Message" explains the details.

In Example 20-4, notice the outgoing interface list for R2. R2 has now removed interface fa0/0 from the outgoing interface list and stopped forwarding traffic on the interface because it received no response to the IGMP Group-Specific query for group 226.1.1.1. As a result, R2 has also removed the C flag (C meaning "connected") from its multicast routing table entry for (10.1.1.10, 226.1.1.1). Additionally, R2 forwards the traffic on its s0/1 interface toward R3 because R3 is still forwarding traffic on its fa0/0 interface and has not yet sent a Prune message to R2.

Rules for Pruning

This section explains two key rules that a PIM-DM router must follow to decide when it can request a prune. Before explaining another example of how PIM-DM reacts to changes in an internetwork, a couple of new multicast terms must be defined. To simplify the wording, the following statements define *upstream router* and *downstream router* from the perspective of a router named R1.

KEY POINT

- R1's upstream router is the router from which R1 receives multicast packets for a particular SPT.
- R1's downstream router is a router to which R1 forwards some multicast packets for a particular SPT.

For example, R1 is R2's upstream router for the packets that S1 is sending to 226.1.1.1 in Figure 20-7. R3 is R2's downstream router for those same packets, because R2 sends those packets to R3.

PIM-DM routers can choose to send a Prune message for many reasons, one of which was covered earlier with regard to Figure 20-6. The main reasons are summarized here:

KEY POINT

- When receiving packets on a non-RPF interface.
- When a router realizes that both of the following are true:
  - — No locally connected hosts in a particular group are listening for packets.
  - — No downstream routers are listening for the group.

This section shows the logic behind the second reason for sending prunes. At this point in the explanation of Figures 20-6 and 20-7, the only host that needs to receive packets sent to 226.1.1.1 is H2. What would the PIM-DM routers in this network do if H2 leaves group 226.1.1.1? Figure 20-8 shows just such an example, with H2 sending an IGMP Leave message for group 226.1.1.1. Figure 20-8 shows how PIM-DM uses this information to dynamically update the SPT.

Figure 20-8 *R3 and R2 Sending Prune Messages*

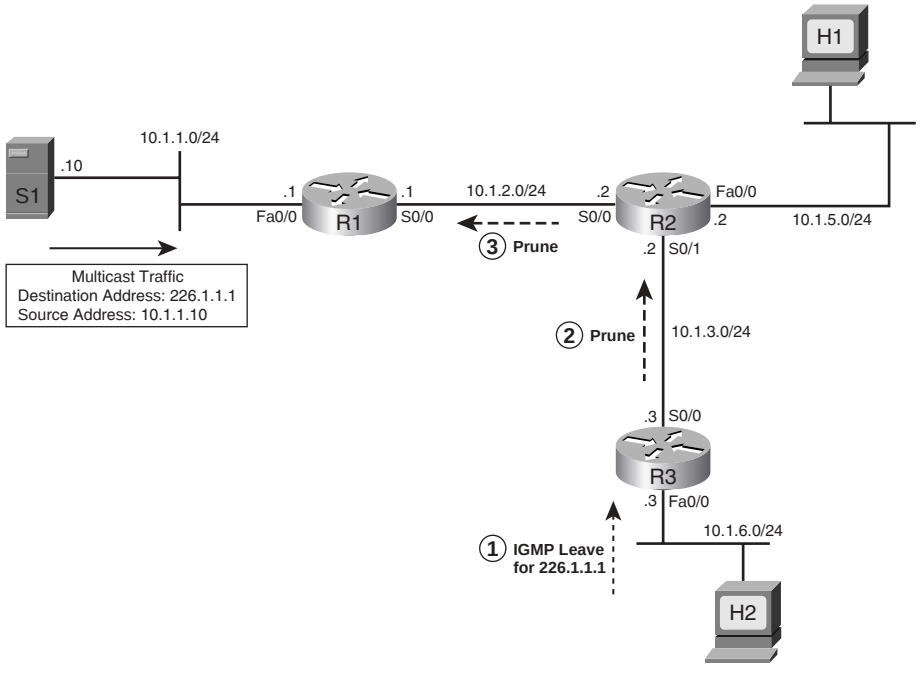


Figure 20-8 shows three steps, with the logic in Steps 2 and 3 being similar but very important:

1. H2 leaves the multicast group by using an IGMP Leave message.
2. R3 uses an IGMP Query to confirm that no other hosts on the LAN want to receive traffic for group 226.1.1.1. So, R3 sends a Prune, referencing the (10.1.1.20, 226.1.1.1) SPT, out its RPF interface R2.
3. R2 does not have any locally connected hosts listening for group 226.1.1.1. Now, its only downstream router has sent a Prune for the SPT with source 10.1.1.10, group 226.1.1.1. Therefore, R2 has no reason to need packets sent to 226.1.1.1 any more. So, R2 sends a Prune, referencing the (10.1.1.20, 226.1.1.1) SPT, out its RPF interface R1.

After the pruning is complete, both R3 and R2 will not be forwarding traffic sent to 226.1.1.1 from source 10.1.1.10. In the routers, the **show ip mroute** command shows that fact using the P (prune) flag, which means that the router has completely pruned itself from that particular (S,G) SPT.

Example 20-5 shows R3's command output with a null outgoing interface list.

Example 20-5 *Multicast Route Table Entry for the Group 226.1.1.1 for R3*

```
(10.1.1.10/32, 226.1.1.1), 00:03:16/00:01:36, flags: PT
  Incoming interface: Serial0/0, RPF nbr 10.1.3.2
    Outgoing interface list: Null
```

After all the steps in Figure 20-8 have been completed, R1 also does not need to send packets sent by 10.1.1.10 to 226.1.1.1 out any interfaces. After receiving a Prune message from R2, R1 has also updated its outgoing interface list, which shows that there is only one outgoing interface and that it is in the pruned state at this time. Example 20-6 shows the details.

Example 20-6 *Multicast Route Table Entry for the Group 226.1.1.1 for R1*

```
(10.1.1.10/32, 226.1.1.1), 00:08:35/00:02:42, flags: CT
  Incoming interface: FastEthernet0/0, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0, Prune/Dense, 00:00:12/00:02:48
```

Of particular interest in the output, R1 has also set the C flag, but for R1 the C flag does not indicate that it has directly connected group members. In this case, the combination of a C flag and an RPF neighbor of 0.0.0.0 indicates that the connected device is the source for the group.

KEY POINT In reality, there is no separate Prune message and Join message; instead, PIM-DM and PIM-SM use a single message called a Join/Prune message. A Prune message is actually a Join/Prune message with a group address listed in the Prune field, and a Join message is a Join/Prune message with a group address listed in the Join field.

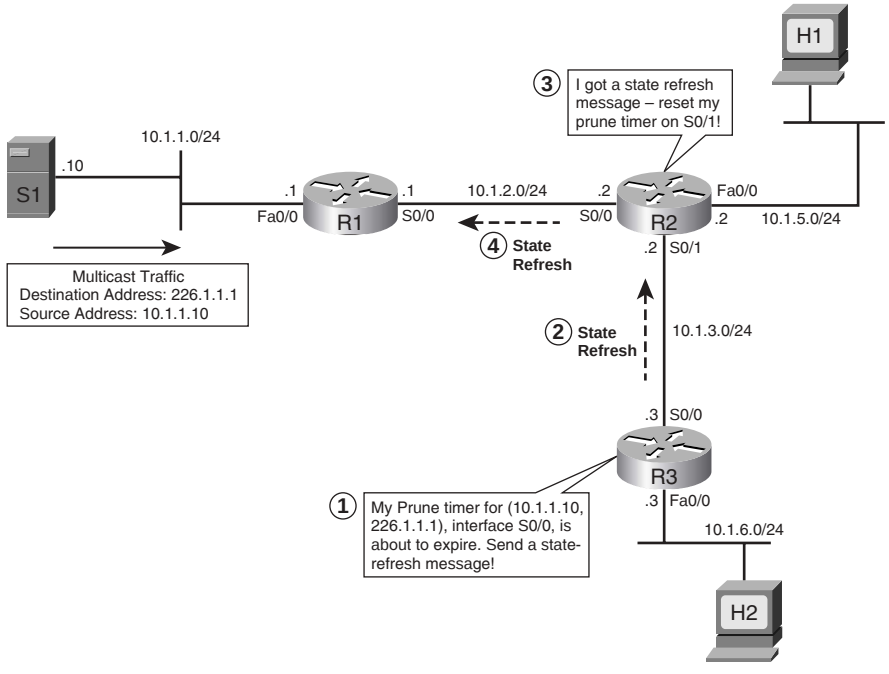
Steady-State Operation and the State Refresh Message

As mentioned briefly earlier in the chapter, with PIM-DM, an interface stays pruned only for 3 minutes by default. Prune messages list a particular source and group (in other words, a particular (S,G) SPT). Whenever a router receives a Prune message, it finds the matching (S,G) SPT entry and marks the interface on which the Prune message was received as "pruned." However, it also sets a Prune timer, default 3 minutes, so that after 3 minutes, the interface is placed into a forwarding state again.

So, what happens with PIM-DM and pruned links? Well, the necessary links are pruned, and 3 minutes later they are added back. More multicasts flow, and the links are pruned. Then they are added back. And so on. So, when Cisco created PIM V2 (published as experimental RFC 3973), it included a feature called *state refresh*. State Refresh messages can prevent this rather inefficient behavior in PIM-DM version 1 of pruning and automatically unpruning interfaces.

Figure 20-9 shows an example that begins with the same state as the network described at the end of the preceding section, "Rules for Pruning," where the link between R1 and R2 and the link between R2 and R3 have been pruned. Almost 3 minutes have passed, and the links are about to be added to the SPT again due to the expiration of the Prune timers.

Figure 20-9 *How PIM-DM Version 2 Uses State Refresh Messages*



The PM State Refresh message can be sent, just before a neighbor's Prune timer expires, to keep the interface in a pruned state. In Figure 20-9, the following steps do just that:

1. R3 monitors the time since it sent the last Prune to R2. Just before the Prune timer expires, R3 decides to send a State Refresh message to R2.
2. R3 sends the State Refresh message to R2, referencing SPT (10.1.1.10, 226.1.1.1).
3. R2 reacts by resetting its Prune timer for the interface on which it received the State Refresh message.
4. Because R2 had also pruned itself by sending a Prune message to R1, R2 also uses State Refresh messages to tell R1 to leave its s0/0 interface in a pruned state.

As long as R3 keeps sending a State Refresh message before the Prune timer on the upstream router (R2) expires, the SPT will remain stable, and there will not be the periodic times of flooding of more multicasts for that (S,G) tree.

Graft Message

When new hosts join a group, routers may need to change the current SPT for a particular (S,G) entry. With PIM-DM, one option could be to wait on the pruned links to expire. For example, in Figure 20-9, R3 could simply quit sending State Refresh messages, and within 3 minutes at most,

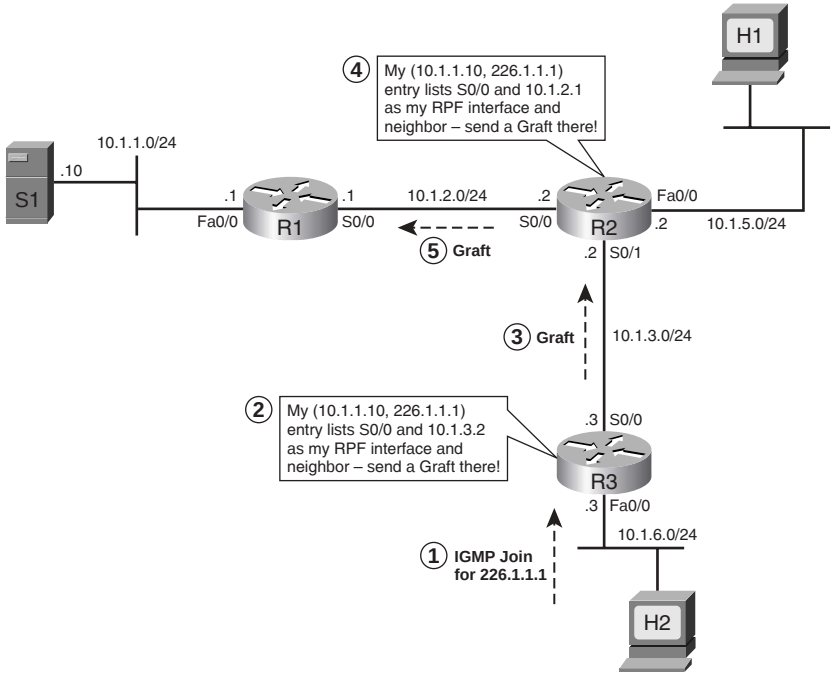
R3 would be receiving the multicast packets for some (S,G) SPT again. However, waiting on the (default) 3-minute Prune timer to expire is not very efficient. To allow routers to "unprune" a previously pruned interface from an SPT, PIM-DM includes the *Graft* message, which is defined as follows:

KEY POINT

A router sends a Graft message to an upstream neighbor—a neighbor to which it had formerly sent a Prune message—causing the upstream router to put the link back into a forwarding state (for a particular (S,G) SPT).

Figure 20-10 shows an example that uses the same ongoing example network. The process shown in Figure 20-10 begins in the same state as described at the end of the preceding section, "Steady-State Operation and the State Refresh Message." Neither host H1 nor H2 has joined group 226.1.1.1, and R2 and R3 have been totally pruned from the (10.1.1.10, 226.1.1.1) SPT. Referring to Figure 20-10, R1's s0/0 interface has been pruned from the (S,G) SPT, so R2 and R3 are not receiving the multicasts sent by server S1 to 226.1.1.1. The example then begins with host H2 joining group 226.1.1.1 again.

Figure 20-10 *R3 and R2 Send Graft Messages*



Without the Graft message, host H2 would have to wait for as much as 3 minutes before it would receive the group traffic. However, with the following steps, as listed in Figure 20-10, H2 will receive the packets in just a few seconds:

1. Host H2 sends an IGMP Join message.

2. R3 looks for the RPF interface for its (S, G) state information for the group 226.1.1.1 (see earlier Example 20-5), which shows the incoming interface as s0/0 and RPF neighbor as 10.1.3.2 for the group.
3. R3 sends the Graft message out s0/0 to R2.
4. R2 now knows it needs to be receiving messages from 10.1.1.10, sent to 226.1.1.1. However, R2's (S,G) entry also shows a P flag, meaning R2 has pruned itself from the SPT. So, R2 finds its RPF interface and RPF neighbor IP address in its (S,G) entry, which references interface s0/0 and router R1.
5. R2 sends a graft to R1.

At this point, R1 immediately puts its s0/0 back into the outgoing interface list, as does R2, and now H2 receives the multicast packets. Note that R1 also sends a Graft Ack message to R2 in response to the Graft message, and R2 sends a Graft Ack in response to R3's Graft message as well.

LAN-Specific Issues with PIM-DM and PIM-SM

This section covers three small topics related to operations that only matter when PIM is used on LANs:

- Prune Override
- Assert messages
- Designated routers

Both PIM-DM and PIM-SM use these features in the same way.

Prune Override

In both PIM-DM and PIM-SM, the Prune process on multiaccess networks operates differently from how it operates on point-to-point links. The reason for this difference is that when one router sends a Prune message on a multiaccess network, other routers might not want the link pruned by the upstream router. Figure 20-11 shows an example of this problem, along with the solution through a PIM Join message that is called a *Prune Override*. In this figure, R1 is forwarding the group traffic for 239.9.9.9 on its fa0/0 interface, with R2 and R3 receiving the group traffic on their e0 interfaces. R2 does not have any connected group members, and its outgoing interface list would show null. The following list outlines the steps in logic shown in Figure 20-11, in which R3 needs to send a Prune Override:

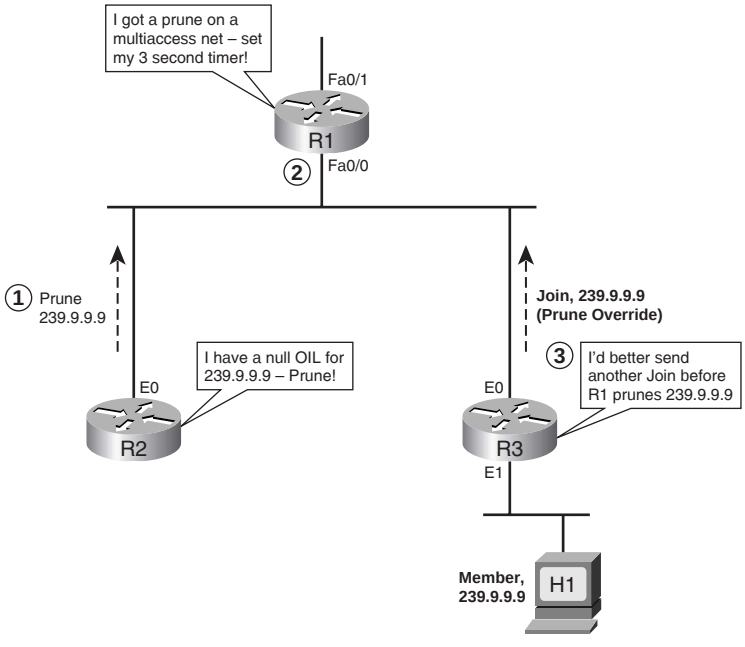
1. R2 sends a Prune for group 239.9.9.9 because R2 has a null outgoing interface list for the group.
2. R1, realizing that it received the Prune on a multiaccess network, knows that other routers might still want to get the messages. So, instead of immediately pruning the interface, R1 sets a 3-second timer that must expire before R1 will prune the interface.

3. R3 also receives the Prune message sent by R2, because Prune messages are multicast to All-PIM-Routers group address 224.0.0.13. R3 still needs to get traffic for 239.9.9.9, so R3 sends a Join message on its e0 interface.

4. (Not shown in Figure 20-11) R1 receives the Join message from R3 before removing its LAN interface from the outgoing interface list. As a result, R1 does not prune its Fa0/0 interface.

Figure 20-11 *Prune Override*

KEY POINT



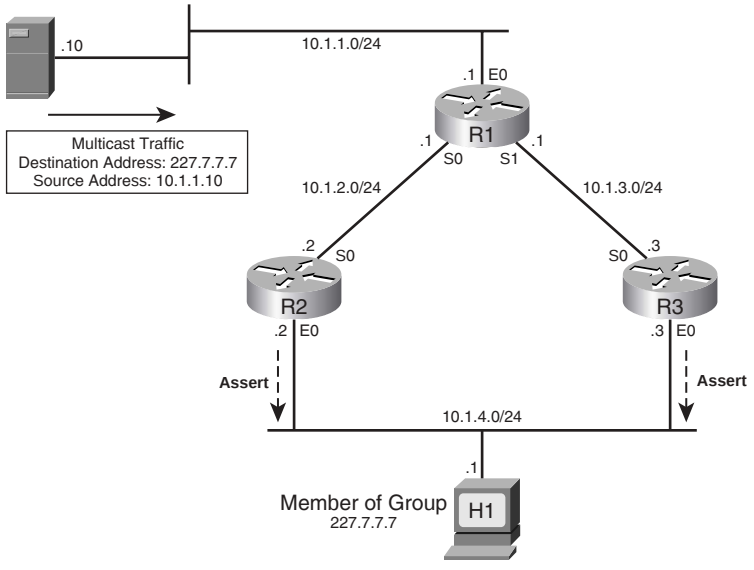
This process is called *Prune Override* because R3 overrides the Prune sent by R2. The Prune Override is actually a Join message, sent by R3 in this case. The message itself is no different from a normal Join. As long as R1 receives a Join message from R3 before its 3-second timer expires, R3 continues to receive traffic without interruption.

Assert Message

The final PIM-DM message covered in this chapter is the PIM Assert message. The Assert message is used to prevent wasted effort when more than one router attaches to the same LAN. Rather than sending multiple copies of each multicast packet onto the LAN, the PIM Assert message allows the routers to negotiate. The winner gets the right to be responsible for forwarding multicasts onto the LAN.

Figure 20-12 shows an example of the need for the Assert message. R2 and R3 both attach to the same LAN, with H1 being an active member of the group 227.7.7.7. Both R2 and R3 are receiving the group traffic for 227.7.7.7 from the source 10.1.1.10.

Figure 20-12 *R2 and R3 Sending Assert Messages*



The goal of the Assert message is to assign the responsibility of forwarding group traffic on the LAN to the router that is closest to the source. When R2 and R3 receive group traffic from the source on their s0 interfaces, they forward it on their e0 interfaces. Both of them have their s0 interfaces in the incoming interface list and e0 interfaces in the outgoing interface list. Now, R2 and R3 receive a multicast packet for the group on their e0 interfaces, which will cause them to send an Assert message to resolve who should be the forwarder.

The Assert process picks a winner based on the routing protocol and metric used to find the route to reach the unicast address of the source. In this example, that means that R2 or R3 will win based on the routes they each use to reach 10.1.1.10. R2 and R3 send and receive Assert messages that include their respective administrative distances of the routing protocols used to learn the route that matches 10.1.1.10, as well as the metric for those routes. The routers on the LAN compare their own routing protocol administrative distance and metrics to those learned in the Assert messages. The winner of the Assert process is determined as follows:

KEY POINT

1. The router advertising the lowest administrative distance of the routing protocol used to learn the route wins.
2. If a tie, the router with the lowest advertised routing protocol metric for that route wins.
3. If a tie, the router with the highest IP address on that LAN wins.

Designated Router

KEY POINT PIM Hello messages are also used to elect a designated router (DR) on a multiaccess network. A PIM-DM or PIM-SM router with the highest IP address becomes a DR.

KEY POINT

The PIM DR concept applies mainly when IGMPv1 is used. IGMPv1 does not have a mechanism to elect a Querier—that is to say that IGMPv1 has no way to decide which of the many routers on a LAN should send IGMP Queries. When IGMPv1 is used, the PIM DR is used as the IGMP Querier. IGMPv2 can directly elect a Querier (the router with the lowest IP address), so the PIM DR is not used as the IGMP Querier when IGMPv2 is used.

Note that on a LAN, one router might win the Assert process for a particular (S,G) SPT, while another might become the IGMP Querier (PIM DR for IGMPv1, IGMP Querier for IGMPv2). The winner of the Assert process is responsible for forwarding multicasts onto the LAN, whereas the IGMP Querier is responsible for managing the IGMP process by being responsible for sending IGMP Query messages on the LAN. Note also that the IGMPv2 Querier election chooses the lowest IP address, and the Assert process uses the highest IP address as a tiebreaker, making it slightly more likely that different routers are chosen for each function.

Summary of PIM-DM Messages

This section concludes the coverage of PIM-DM. Table 20-2 lists the key PIM-DM messages covered in this chapter, along with a brief definition of their use.

Table 20-2 *Summary of PIM-DM Messages*

KEY POINT

PIM Message	Definition
Hello	Used to form neighbor adjacencies with other PIM routers, and to maintain adjacencies by monitoring for received Hellos from each neighbor. Also used to elect a PIM DR on multiaccess networks.
Prune	Used to ask a neighboring router to remove the link over which the Prune flows from that neighboring router's outgoing interface list for a particular (S,G) SPT.
State Refresh	Used by a downstream router, sent to an upstream router on an RPF interface, to cause the upstream router to reset its Prune timer. This allows the downstream router to maintain the pruned state of a link, for a particular (S,G) SPT.
Assert	Used on multiaccess networks to determine which router wins the right to forward multicasts onto the LAN, for a particular (S,G) SPT.
Prune Override (Join)	On a LAN, a router may multicast a Prune message to its upstream routers. Other routers on the same LAN, wanting to prevent the upstream router from pruning the LAN, immediately send another Join message for the (S,G) SPT. (The Prune Override is not actually a Prune Override message—it is a Join. This is the only purpose of a Join message in PIM-DM, per RFC 3973.)
Graft/Graft-Ack	When a pruned link needs to be added back to an (S,G) SPT, a router sends a Graft message to its RPF neighbor. The RPF neighbor acknowledges with a Graft-Ack.

The next two short sections introduce two other dense-mode protocols, DVMRP and MOSPF.

Distance Vector Multicast Routing Protocol

RFC 1075 describes Version 1 of DVMRP. DVMRP has many versions. The operation of DVMRP is similar to PIM-DM. The major differences between PIM-DM and DVMRP are defined as follows:

- Cisco IOS does not support a full implementation of DVMRP; however, it does support connectivity to a DVMRP network.
- DVMRP uses its own distance vector routing protocol that is similar to RIPv2. It sends route updates every 60 seconds and considers 32 hops as infinity. Use of its own routing protocol adds more overhead to DVMRP operation compared to PIM-DM.
- DVMRP uses Probe messages to find neighbors using the All DVMRP Routers group address 224.0.0.4.
- DVMRP uses a truncated broadcast tree, which is similar to an SPT with some links pruned.

Multicast Open Shortest Path First

MOSPF is defined in RFC 1584, "Multicast Extensions to OSPF," which is an extension to the OSPFv2 unicast routing protocol. The basic operation of MOSPF is described here:

- MOSPF uses the group membership LSA, Type 6, which it floods throughout the originating router's area. As with unicast OSPF, all MOSPF routers in an area must have identical link-state databases so that every MOSPF router in an area can calculate the same SPT.
- The SPT is calculated "on-demand," when the first multicast packet for the group arrives.
- Through the SPF calculation, all the routers know where the attached group members are, based on the group membership LSAs.
- After the SPF calculation is completed, entries are made into each router's multicast forwarding table.
- Just like unicast OSPF, the SPT is loop free, and every router knows the upstream interface and downstream interfaces. As a result, an RPF check is not required.
- Obviously, MOSPF can only work with the OSPF unicast routing protocol. MOSPF is suitable for small networks. As more hosts begin to source multicast traffic, routers have to perform a higher number of Dijkstra algorithm computations, which demands an increasing level of router CPU resources. Cisco IOS does not support MOSPF.

Sparse-Mode Routing Protocols

There are two sparse-mode routing protocols:

- Protocol Independent Multicast Sparse Mode (PIM-SM)
- Core-Based Tree (CBT)

This section covers the operation of PIM-SM.

Operation of Protocol Independent Multicast Sparse Mode

PIM-SM works with a completely opposite strategy from that of PIM-DM, although the mechanics of the protocol are not exactly opposite. PIM-SM assumes that no hosts want to receive multicast packets until they specifically ask to receive them. As a result, until a host in a subnet asks to receive multicasts for a particular group, multicasts are never delivered to that subnet. With PIM-SM, downstream routers must request to receive multicasts using PIM Join messages. Also, once they are receiving those messages, the downstream router must continually send Join messages to the upstream router—otherwise, the upstream router stops forwarding, putting the link in a pruned state. This process is opposite to that used by PIM-DM, in which the default is to flood multicasts, with downstream routers needing to continually send Prunes or State Refresh messages to keep a link in a pruned state.

PM-SM makes the most sense with a small percentage of subnets that need to receive packets sent to any multicast group.

Similarities Between PIM-DM and PIM-SM

PIM-SM has many similarities to PIM-DM. Like PIM-DM, PIM-SM uses the unicast routing table to perform RPF checks—regardless of what unicast routing protocol populated the table. (Like PIM-DM, the “protocol independent” part of the PIM acronym comes from the fact that PIM-SM is not dependent on any particular unicast IP routing protocol.) In addition, PIM-SM also uses the following mechanisms that are used by PIM-DM:

- PIM Neighbor discovery through exchange of Hello messages.
- Recalculation of the RPF interface when the unicast routing table changes.
- Election of a DR on a multiaccess network. The DR performs all IGMP processes when IGMPv1 is in use on the network.
- The use of Prune Overrides on multiaccess networks.
- Use of Assert messages to elect a designated forwarder on a multiaccess network. The winner of the Assert process is responsible for forwarding unicasts onto that subnet.

NOTE The preceding list was derived, with permission, from *Routing TCP/IP*, Volume II, by Jeff Doyle and Jennifer DeHaven Carroll.

These mechanisms are described in the "Operation of Protocol Independent Multicast Dense Mode" section and thus are not repeated in this section.

Sources Sending Packets to the Rendezvous Point

PIM-SM uses a two-step process to initially deliver multicast packets from a particular source to the hosts wanting to receive packets. Later, the process is improved beyond these initial steps. The steps for the initial forwarding of multicasts with PIM-SM are as follows:

KEY POINT

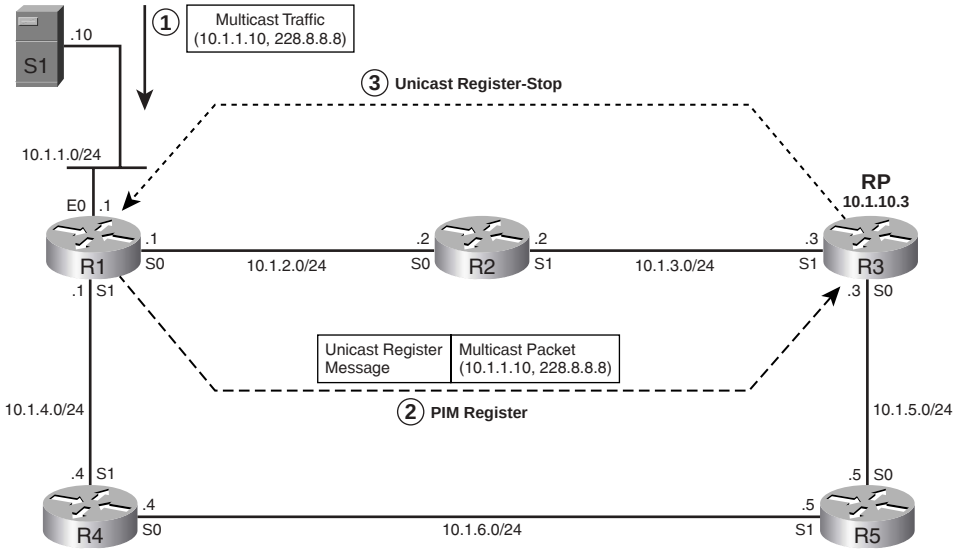
1. Sources send the packets to a router called the rendezvous point (RP).
2. The RP sends the multicast packets to all routers/hosts that have registered to receive packets for that group. This process uses a shared tree.

NOTE In addition to these two initial steps, routers with local hosts that have sent an IGMP Join for a group can go a step further, joining the source-specific tree for a particular (S,G) SPT.

This section describes the first of these two steps, in which the source sends packets to the RP. To make that happen, the router connected to the same subnet as the source host must register with the RP. The RP accepts the registration only if the RP knows of any routers or hosts that need to receive a copy of those multicasts.

Figure 20-13 shows an example of the registration process in which the RP knows that no hosts currently want the IP multicasts sent to group 228.8.8.8—no matter which source is sending them. The configuration for this example is simple, with all the routers configured with the global command **ip multicast-routing** and the interface command **ip pim sparse-mode** on all the interfaces. Also, all routers have statically configured R3 as the RP by using the global command **ip pim rp-address 10.1.10.3**. Usually, a loopback interface address is used as an RP address. The loopback network 10.1.10.3/32 of R3 is advertised in the unicast routing protocol so that all the routers know how to reach the RP.

Figure 20-13 *Source Registration Process when RP Has Not Received a Request for the Group from Any PIM-SM Router*



The following three steps, referenced in Figure 20-13, describe the sequence of events for the Source Registration process when the RP has not received a request for the group from any PIM-SM router because no host has yet joined the group.

1. Host S1 begins sending multicasts to 228.8.8.8, and R1 receives those multicasts because it connects to the same LAN.
2. R1 reacts by sending unicast PIM Register messages to the RP. The Register messages are unicasts sent to the RP IP address, 10.1.10.3 in this case.
3. R3 sends unicast Register-Stop messages back to R1 because R3 knows that it does not have any need to forward packets sent to 228.8.8.8.

In this example, the router near the source (R1) is attempting to register with the RP, but the RP tells R1 not to bother any more, because no one wants those multicast messages. R1 has not forwarded any of the native multicast messages at this point, in keeping with the PIM-SM strategy of not forwarding multicasts until a host has asked for them. However, the PIM Register message shown in Figure 20-13 encapsulates the first multicast packet. As will be seen in Figure 20-14, the encapsulated packet would be forwarded by the RP had any senders been interested in receiving the packets sent to that multicast group.

The source host may keep sending multicasts, so R1 needs to keep trying to register with the RP in case some host finally asks to receive the packets. So, when R1 receives the Register-Stop messages, it starts a 1-minute Register-Suppression timer. 5 seconds before the timer expires,

R1 sends another Register message with a flag set, called the Null-Register bit, without any encapsulated multicast packets. As a result of this additional Register message, one of two things will happen:

- If the RP still knows of no hosts that want to receive these multicast packets, it sends another Register-Stop message to R1, and R1 resets its Register-Suppression timer.
- If the RP now knows of at least one router/host that needs to receive these multicast packets, it does not reply to this briefer Register message. As a result, R1, when its timer expires, again sends its multicast packets to R3 (RP) encapsulated in PIM Register messages.

Joining the Shared Tree

So far, this section on PIM-SM has explained the beginnings of the registration process, by which a router near the source of multicast packets registers with the RP. Before completing that discussion, however, the concept of the shared tree for a multicast group, also called the *root-path tree (RPT)*, must be explained. As mentioned earlier, PIM-SM initially causes multicasts to be delivered in a two-step process: first, packets are sent from the source to the RP, and then the RP forwards the packets to the subnets that have hosts that need a copy of those multicasts. PIM-SM uses this shared tree in the second part of the process.

The RPT is a tree, with the RP at the root, that defines over which links multicasts should be forwarded to reach all required routers. One such tree exists for each multicast group that is currently active in the internetwork. So, once the multicast packets sent by each source are forwarded to the RP, the RP uses the RPT for that multicast group to determine where to forward these packets.

PIM-SM routers collectively create the RPT by sending PIM Join messages toward the RP. In PIM-SM, multicast traffic is sent only to routers that specifically request it. PIM-SM routers request the traffic by joining the RPT by sending a Join toward the RP.

PIM-SM routers choose to send a Join under two conditions:

KEY POINT

- When a PIM-SM router receives a PIM Join message on any interface other than the interface used to route packets toward the RP
- When a PIM-SM router receives an IGMP Membership Report message from a host on a directly connected subnet

Figure 20-14 shows an example of the PIM-SM join process, using the same network as Figure 20-12 but with H1 joining group 228.8.8.8. The routers react to the IGMP Join by sending a Join toward the RP, to become part of the shared SPT (*,228.8.8.8).

Figure 20-14 *Creating a Shared Tree for (*,228.8.8.8)*

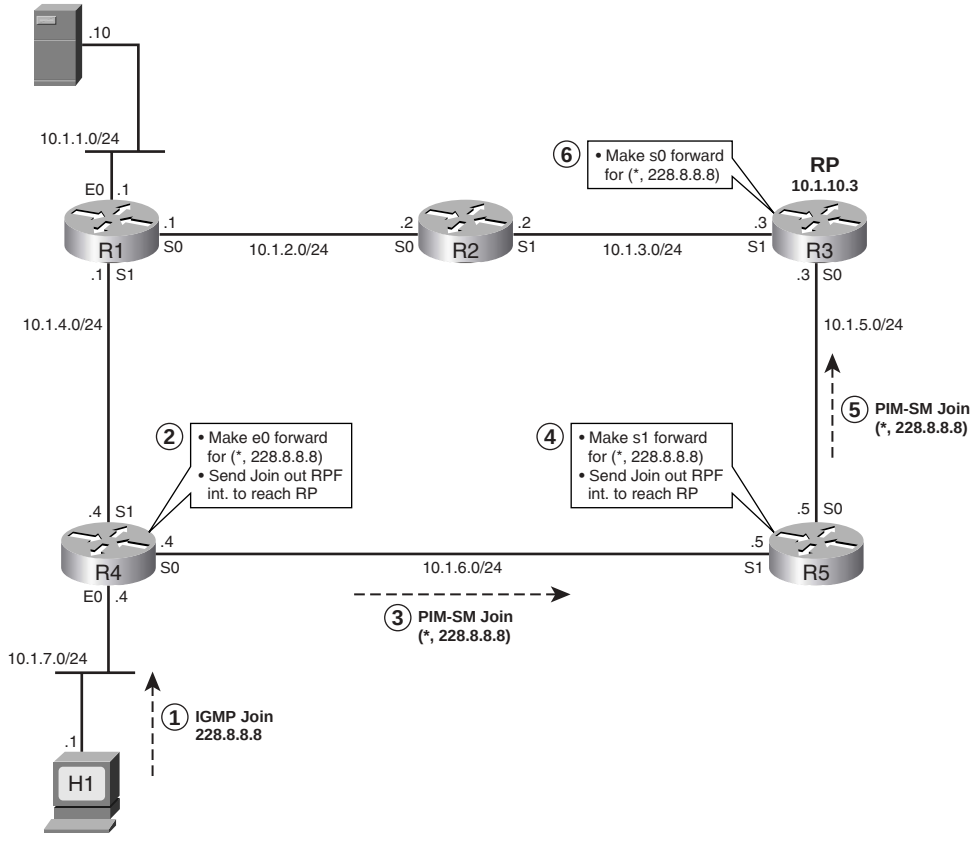


Figure 20-14 shows how H1 causes a shared tree (*,228.8.8.8) to be created, as described in the following steps:

1. H1 sends an IGMP Join message for the group 228.8.8.8.
2. R4 realizes it now needs to ask the RP to send it packets sent to 228.8.8.8, so R4 sends a PIM Join for the shared tree for group 228.8.8.8 toward the RP. R4 also puts its e0 interface into a forwarding state for the RPT for group 228.8.8.8.
3. R4 sends the Join to the RP.
4. R5 receives the Join on its s1 interface, so R5 puts its s1 interface in a forwarding state for the shared tree (represented by (*,228.8.8.8)). R5 also knows it needs to forward the Join toward the RP.
5. R5 sends the Join toward the RP.
6. R3, the RP, puts its s0 interface in a forwarding state for the (*,288.8.8.8) shared tree.

By the end of this process, the RP knows that at least one host wants packets sent to 228.8.8.8. The RPT for group 228.8.8.8 is formed with R3's s0 interface, R5's s1 interface, and R4's e0 interface.

NOTE The notation (*,G) represents a single RPT. The * represents a wildcard, meaning "any source," because the PIM-SM routers use this shared tree regardless of the source of the packets. For example, a packet sent from any source IP address, arriving at the RP, and destined to group 228.8.8.8, would cause the RP to use its (*,228.8.8.8) multicast routing table entries, because these entries are part of the RPT for group 228.8.8.8.

Completion of the Source Registration Process

So far in this description of PIM-SM, a source (10.1.1.10) sent packets to 228.8.8.8, as shown in Figure 20-13—but no one cared at the time, so the RP did not forward the packets. Next, you learned what happens when a host does want to receive packets, with the routers reacting to create the RPT for that group. This section completes the story by showing how an RP reacts to a PIM Register message when the RP knows that some hosts want to receive those multicasts.

When the RP receives a Register message for an active multicast group—in other words, the RP believes that it should forward packets sent to the group—the RP does not send a Register-Stop message, as was shown back in Figure 20-13. Instead, it reacts to the Register message by de-encapsulating the multicast packet, and forwarding it.

The behavior of the RP in reaction to the Register message points out the second major function of the Register message. Its main two functions are as follows:

KEY POINT

- To allow a router to inform the RP that it has a local source for a particular multicast group
- To allow a router to forward multicasts to the RP, encapsulated inside a unicast packet, until the registration process is completed

To show the complete process, Figure 20-15 shows an example. In the example, host H1 has already joined group 228.8.8.8, as shown in Figure 20-14. The following steps match those identified in Figure 20-15. Note that Step 3 represents the forwarding of the multicasts that were encapsulated inside Register messages at Step 2.

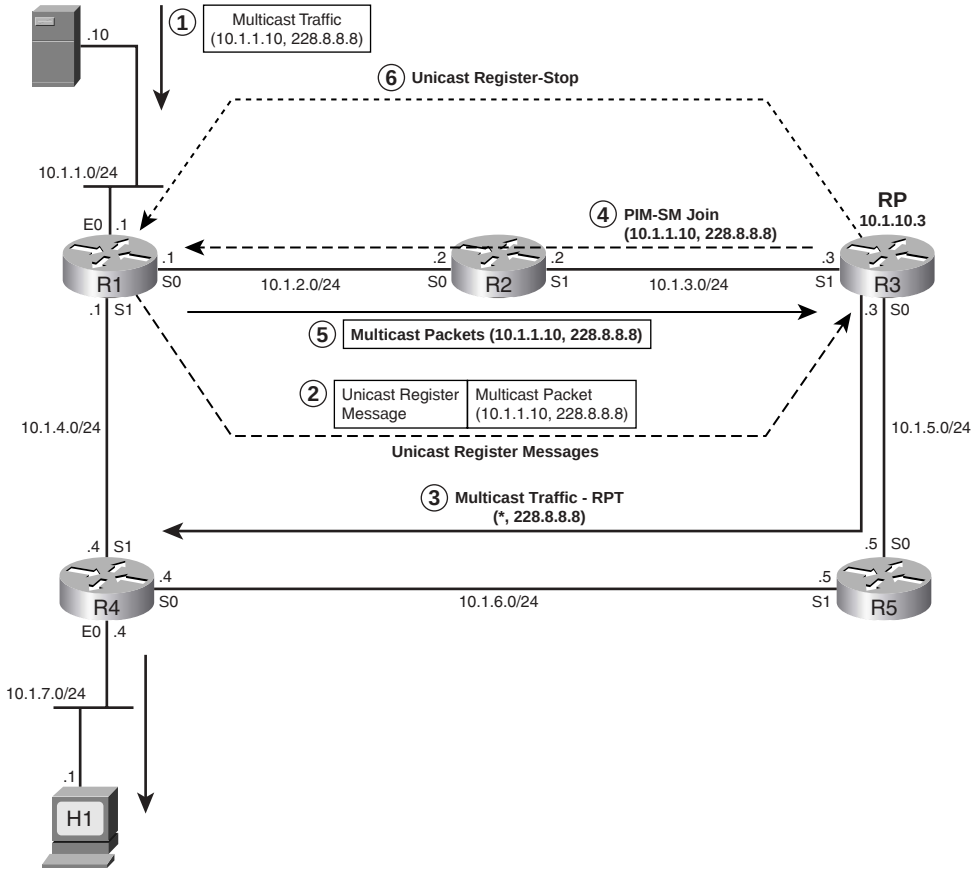
1. Host S1 sends multicasts to 228.8.8.8.
2. Router R1 encapsulates the multicasts, sending them inside Register messages to the RP, R3.
3. R3, knowing that it needs to forward the multicast packets, de-encapsulates the packets and sends them toward H1. (This action allows R1 and R3 to distribute the multicasts while the registration process completes.) R5 forwards the group traffic to R4 and R4 forwards it on its LAN.

4. R3 joins the SPT for source 10.1.1.10, group 228.8.8.8, by sending a PIM-SM Join message for group (10.1.1.10,228.8.8.8) toward the source 10.1.1.10.

5. When R1 and R2 receive the PIM-SM Join message from R2 requesting the group traffic from the source, they start forwarding group traffic toward the RP. At this point, R3 (the RP) now receives this traffic on the SPT from the source. However, R1 is also still sending the Register messages with encapsulated multicast packets to R3.

6. R3 sends unicast Register-Stop messages to R1. When R1 receives the Register-Stop messages from R3, it stops sending the encapsulated unicast Register messages to R3.

Figure 20-15 *Source Registration when the RP Needs to Receive Packets Sent to that Group*



The process may seem like a lot of trouble, but at the end of the process, multicasts are delivered to the correct locations. The process uses the efficient SPT from the source to the RP, and the shared tree (*,228.8.8.8) from the RP to the subnets that need to receive the traffic.

Note that the PIM protocols could have just let a router near the source, such as R1 in this example, continue to encapsulate multicasts inside the unicast Register messages. However, it is inefficient

to make R1 encapsulate every multicast packet, make R3 de-encapsulate every packet, and then make R3 forward the traffic. So, PIM-SM has the RP, R3 in this case, join the group-specific tree for that (S,G) combination.

Shared Distribution Tree

In Figure 20-15, the group traffic that flows over the path from the RP (R3) to R5 to R4 is called a *shared distribution tree*. It is also called a *root-path tree (RPT)* because it is rooted at the RP. If the network has multiple sources for the same group, traffic from all the sources would first travel to the RP (as shown with the traffic from host S1 in Figure 20-14), and then travel down this shared RPT to all the receivers. Because all sources in the multicast group use a common shared tree, a wildcard notation of (*,G) is used to identify an RPT, where * represents all sources and G represents the multicast group address. The RPT for the group 228.8.8.8 shown in Figure 20-14 would be written as (*,228.8.8.8).

Example 20-7 shows the multicast route table entry for R4 in Figure 20-15. On a Cisco router, the **show ip mroute** command displays the multicast route table entries.

Example 20-7 *Multicast Route Table Entry for the Group 228.8.8.8 for R4*

```
(*, 228.8.8.8), 00:00:08/00:02:58, RP 10.1.10.3, flags: SC
  Incoming interface: Serial0, RPF nbr 10.1.6.5
  Outgoing interface list:
    Ethernet0, Forward/Sparse, 00:00:08/00:02:52
```

The interpretation of the information shown in Example 20-7 is as follows:

- The first line shows that the (*,G) entry for the group 228.8.8.8 was created 8 seconds ago, and if R4 does not forward group packets using this entry in 2 minutes and 58 seconds, it will expire. Every time R4 forwards a packet, the timer is reset to 3 minutes. This entry was created because R4 received an IGMP Join message from H1.
- The RP for this group is 10.1.10.3 (R3). The S flag indicates that this group is using the sparse-mode (PIM-SM) routing protocol. The C flag indicates that R4 has a directly connected group member for 228.8.8.8.
- The incoming interface for this (*,228.8.8.8) entry is s0 and the RPF neighbor is 10.1.6.5. Note that for the SPT, the RPF interface is chosen based on the route to reach the RP, not the route used to reach a particular source.
- Group traffic is forwarded out on the Ethernet0 interface. In this example, Ethernet0 was added to the outgoing interface list because an IGMP Report message was received on this interface from H1. This interface has been in the forwarding state for 8 seconds. The Prune timer indicates that if an IGMP Join is not received again on this interface within the next 2 minutes and 52 seconds, it will be removed from the outgoing interface list.

Steady-State Operation by Continuing to Send Joins

To maintain the forwarding state of interfaces, PIM-SM routers must send PIM Join messages periodically. If a router fails to send Joins periodically, PIM-SM moves interfaces back to a pruned state.

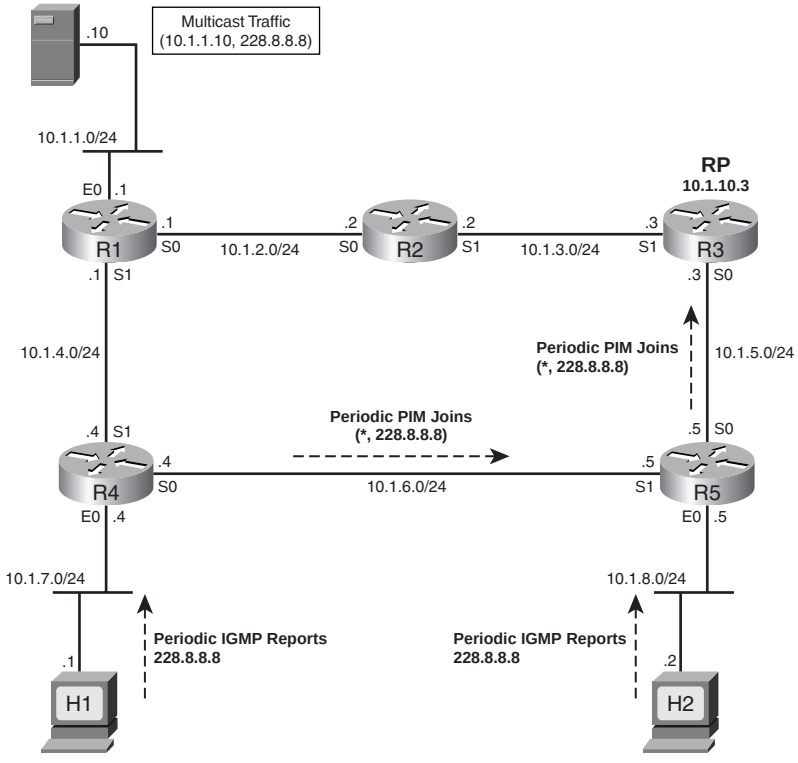
PIM-SM routers choose to maintain the forwarding state on links based on two general criteria:

KEY POINT

- A downstream router continues to send PIM joins for the group.
- A locally connected host still responds to IGMP Query messages with IGMP Report messages for the group.

Figure 20-16 shows an example in which R5 maintains the forwarding state of its link to R3 based on both of these reasons. H2 has also joined the shared tree for 228.8.8.8. H1 had joined earlier, as shown in Figures 20-14 and 20-15.

Figure 20-16 *Host H2 Sends an IGMP Join Message*



Example 20-8 shows the multicast route table entry for R5 in Figure 20-16, with these two interfaces in a forwarding state.

Example 20-8 *Multicast Route Table Entry for the Group 228.8.8.8 for R5*

```
(*,228.8.8.8), 00:00:05/00:02:59, RP 10.1.10.3, flags: SC
  Incoming interface: Serial0, RPF nbr 10.1.5.3
  Outgoing interface list:
    Serial1, Forward/Sparse, 00:01:15/00:02:20
    Ethernet0, Forward/Sparse, 00:00:05/00:02:55
```

In Example 20-8, two interfaces are listed in the outgoing interface list. The s1 interface is listed because R5 has received a PIM-SM Join message from R4. In PIM-SM, the downstream routers need to keep sending PIM-SM Join messages every 60 seconds to the upstream router. When R5 receives another PIM-SM Join from R4 on its s1 interface, it resets the Prune timer to the default value of 3 minutes. If R5 does not receive a PIM-SM Join from R4 before R5's Prune timer on that interface expires, R5 places its s1 interface in a pruned state and stops forwarding the traffic on the interface.

By contrast, R5's e0 interface is listed as forwarding in R5's outgoing interface list because R5 has received an IGMP Join message from H2. Recall from Chapter 19 that a multicast router sends an IGMP general query every 60 or 125 seconds (depending on the IGMP version) on its LAN interfaces. It must receive at least one IGMP Report/Join message as a response for a group; otherwise, it stops forwarding the group traffic on the interface. When R5 receives another IGMP Report message on its e0 interface, it resets the Prune timer for the entry to the default value of 3 minutes.

Note also that on R5, the receipt of the PIM Join from R4, or the IGMP Report on e0, triggers R5's need to send the PIM Join toward the RP.

Examining the RP's Multicast Routing Table

In the current state of the ongoing example, as last shown in Figure 20-16, the RP (R3) has joined the SPT for source 10.1.1.10, group 228.8.8.8. The RP also is the root of the shared tree for group 228.8.8.8. Example 20-9 shows both entries in R3's multicast route table.

Example 20-9 *Multicast Route Table Entry for the Group 228.8.8.8 for R3*

```
(*,228.8.8.8), 00:02:27/00:02:59, RP 10.1.10.3, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0, Forward/Sparse, 00:02:27/00:02:33
(10.1.1.10/32, 228.8.8.8), 00:02:27/00:02:33, flags: T
  Incoming interface: Serial1, RPF nbr 10.1.3.2,
  Outgoing interface list:
  Outgoing interface list: Null
```

The first entry shows the shared tree, as indicated by the S flag. Notice the incoming interface is Null because R3, as RP, is the root of the tree. Also, the RPF neighbor is listed as 0.0.0.0 for the same reason. In other words, it shows that the shared-tree traffic for the group 228.8.8.8 has originated at this router and it does not depend on any other router for the shared-tree traffic.

The second entry shows the SPT entry on R3 for multicast group 228.8.8.8, source 10.1.1.10. The T flag indicates that this entry is for an SPT, and the source is listed at the beginning of that same line (10.1.1.10). The incoming interface is s1 and the RPF neighbor for the source address 10.1.1.10 is 10.1.3.2.

As you can see, an RP uses the SPT to pull the traffic from the source to itself and uses the shared tree to push the traffic down to the PIM-SM routers that have requested it.

Shortest-Path Tree Switchover

PIM-SM routers could continue forwarding packets via the PIM-SM two-step process, whereby sources send packets to the RP, and the RP sends them to all other routers using the RPT. However, one of the most fascinating aspects of PIM-SM operations is that each PIM-SM router can build the SPT between itself and the source of a multicast group and take advantage of the most efficient path available from the source to the router. In Figure 20-16, R4 is receiving the group traffic from the source via the path R1-R2-R3-R5-R4. However, it is obvious that it would be more efficient for R4 to receive the group traffic directly from R1 on R4's s1 interface.

In the section "Completion of the Source Registration Process," earlier in this chapter, you saw that the PIM-SM design allows an RP to build an SPT between itself and the router that is directly connected with the source (also called the source DR) to pull the group traffic. Similarly, the PIM-SM design also allows any other PIM-SM router to build an SPT between the router and the source DR. This feature allows a PIM-SM router to avoid using the inefficient path, such as the one used by R4 in Figure 20-16. Also, once the router starts receiving the group traffic over the SPT, it can send a Prune message to the upstream router of the shared tree to stop forwarding the traffic for the group.

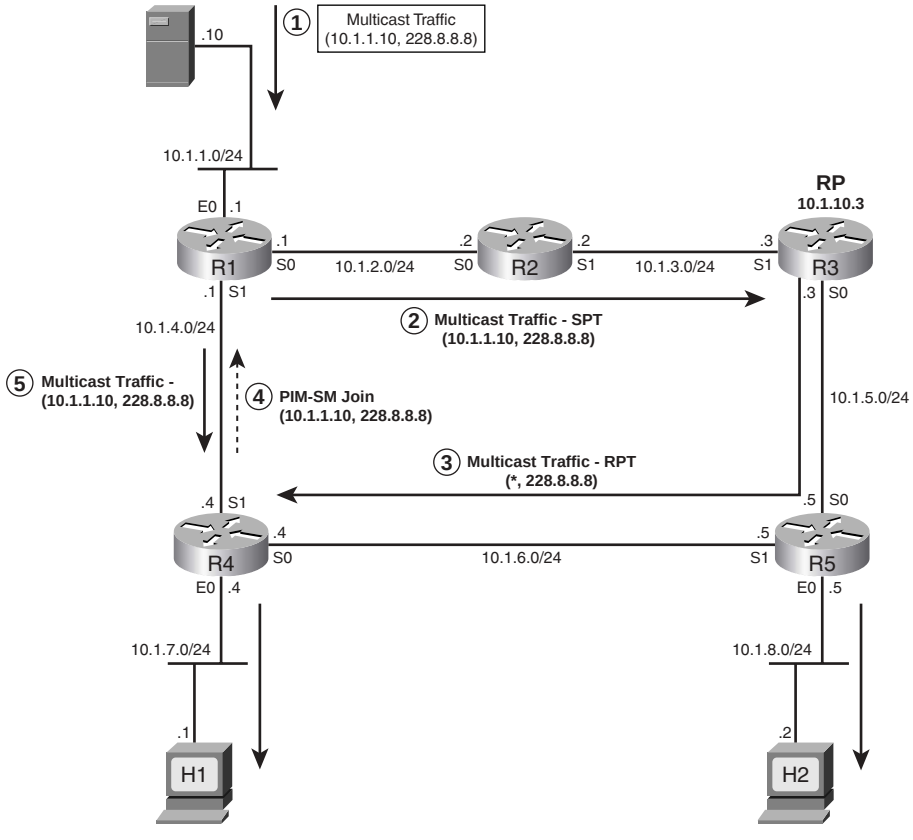
The question is, when should a router switch over from RPT to SPT? RFC 2362 for PIM-SM specifies that, "The recommended policy is to initiate the switch to the SP-tree after receiving a significant number of data packets during a specified time interval from a particular source." What number should be considered as a significant number? The RFC does not specify that. Cisco routers, by default, switch over from the RPT to the source-specific SPT after they receive the first packet from the shared tree.

NOTE You can change this behavior by configuring the global command **ip pim spt-threshold** *rate* on any router for any group. Once the traffic rate exceeds the stated rate (in kbps), the router joins the SPT. The command impacts the behavior only on the router(s) on which it is configured.

If a router is going to switch to SPT, why join the RPT first? In PIM-SM, a router does not know the IP address of a source until it receives at least one packet for the group from the source. After it receives one packet on the RPT, it can learn the IP address of a source, and initialize a switchover to the SPT for that (source,group) combination.

With the default Cisco PIM-SM operation, when multicast packets begin arriving on R4's s0 interface via the shared tree, R4 attempts to switch to the SPT for source 10.1.1.10. Figure 20-17 shows the general steps.

Figure 20-17 *R4 Initializing Switchover from RPT to SPT by Sending a PIM-SM Join to R1*



The first three steps Figure 20-17 are as follows:

1. The source (S1,10.1.1.10) sends a multicast packet to the first-hop router R1.
2. R1 forwards the packet to the RP (R3).
3. The RP forwards the packet to R4 via the shared tree.

At Step 3, R4 learned that the source address of the multicast group 228.8.8.8 is 10.1.1.10. So, besides forwarding the packet at Step 3, R4 can use that information to join the SPT for group 228.8.8.8, from source 10.1.1.10, using the following steps from Figure 20-17.

4. R4 consults its unicast routing table, finds the next-hop address and outgoing interface it would use to reach source 10.1.1.10, and sends the PIM-SM Join message out that interface (s1) to R1. This PIM-SM Join message is specifically for the SPT of (10.1.1.10,228.8.8.8). The Join travels hop by hop until it reaches the source DR.

5. As a result of the Join, R1 places its s1 interface in a forwarding state for SPT (10.1.1.10,228.8.8.8). So, R1 starts forwarding multicasts from 10.1.1.10 to 228.8.8.8 out its s1 interface as well.

R4 now has a multicast routing table entry for the SPT, as shown in Example 20-10.

Example 20-10 *Multicast Route Table Entry for the Group 228.8.8.8 for R4*

```
(*,228.8.8.8), 00:02:36/00:02:57, RP 10.1.10.3, flags: SCJ
  Incoming interface: Serial0, RPF nbr 10.1.6.5
  Outgoing interface list:
    Ethernet0, Forward/Sparse, 00:02:36/00:02:13
(10.1.1.10/32, 228.8.8.8), 00:00:23/00:02:33, flags: CJT
  Incoming interface: Serial1, RPF nbr 10.1.4.1,
  Outgoing interface list:
  Ethernet0, Forward/Sparse, 00:00:23/00:02:37
```

In Example 20-10, you see two entries for the group. The J flag (for join) on both the entries indicates that the traffic was switched from RPT to SPT, and now the (S,G) entry will be used for forwarding multicast packets for the group. Notice that the incoming interfaces for the (*,G) entry and (S,G) entry are different.

Pruning from the Shared Tree

Once a PIM-SM router has joined a more efficient SPT, it may not need to receive multicast packets over the RPT any more. For example, when R4 in Figure 20-17 notices that it is receiving the group traffic over RPT and SPT, it can and should ask the RP to stop sending the traffic.

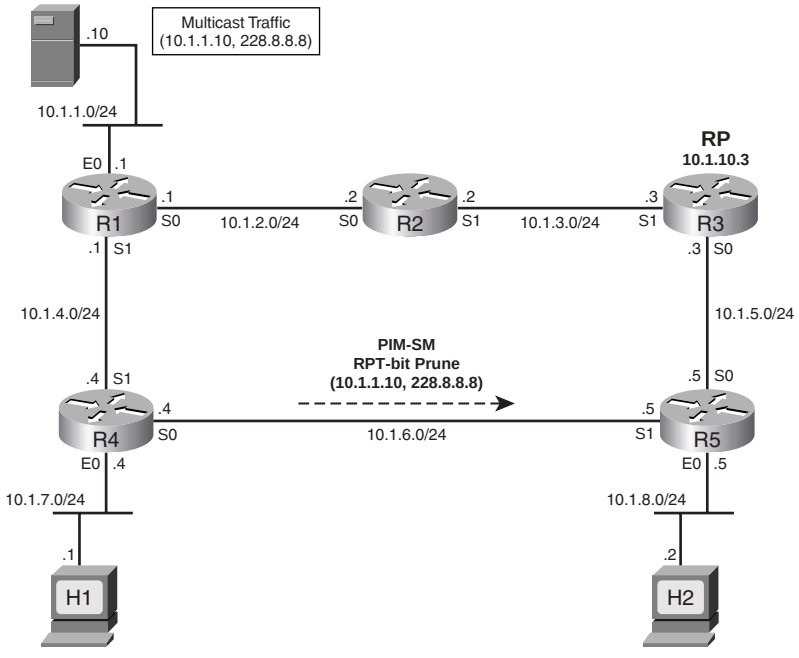
To stop the RP from forwarding traffic to a downstream router on the shared tree, the downstream router sends a PIM-SM Prune message to the RP. The Prune message references the (S,G) SPT, which identifies the IP address of the source. Essentially, this prune means the following to the RP:

> Stop forwarding packets from the listed source IP address, to the listed group address, down the RPT.

For example, in Figure 20-18, which continues the example shown in Figure 20-17, R4 sends a Prune out its s0 interface toward R5. The Prune lists (S,G) entry (10.1.1.10,228.8.8.8), and it sets a bit called the RP-tree bit (RPT-bit). By setting the RPT-bit in the Prune message, R4 informs

R5 (the upstream router) that it has switched to SPT and the Prune message is for the redundant traffic for the group 228.8.8.8, from 10.1.1.10, that R4 is receiving on the shared tree.

Figure 20-18 *R4 Sends PIM-SM Prune with RP Bit Set to R5*



To stop the packets from being sent over the RPT to R4, R5 must prune its interface s1 in the RPT (*, 228.8.8.8). R5 may go on to join the SPT for (10.1.1.10,228.8.8.8.8) as well.

This concludes the coverage of the operations of PIM-SM. The next section covers some details about how routers can learn the IP address of the PIM RP.

Dynamically Finding RPs and Using Redundant RPs

In a PIM-SM network, every router must somehow learn the IP address of an RP. A PIM-SM router can use one of the following three methods to learn the IP address of an RP:

KEY POINT

- The RP address can be statically configured on all the PIM-SM routers with the Cisco IOS global command **ip pim rp-address** *address*. This is the method used for the five-router topology shown in Figure 20-19.
- The Cisco-proprietary Auto-RP protocol can be used to designate the RP and advertise its IP address so that all PIM-SM routers can learn its IP address automatically.
- A standard BootStrap Router (BSR) protocol can be used to designate the RP and advertise its IP address so that all the PIM-SM routers can learn its IP address automatically.

Additionally, because PIM-SM relies so heavily on the RP, it makes sense to have redundant RPs. Cisco IOS offers two methods of providing redundant RPs, which are also covered in this section:

KEY POINT

- Anycast RP using the Multicast Source Discovery Protocol (MSDP)
- BootStrap Router (BSR)

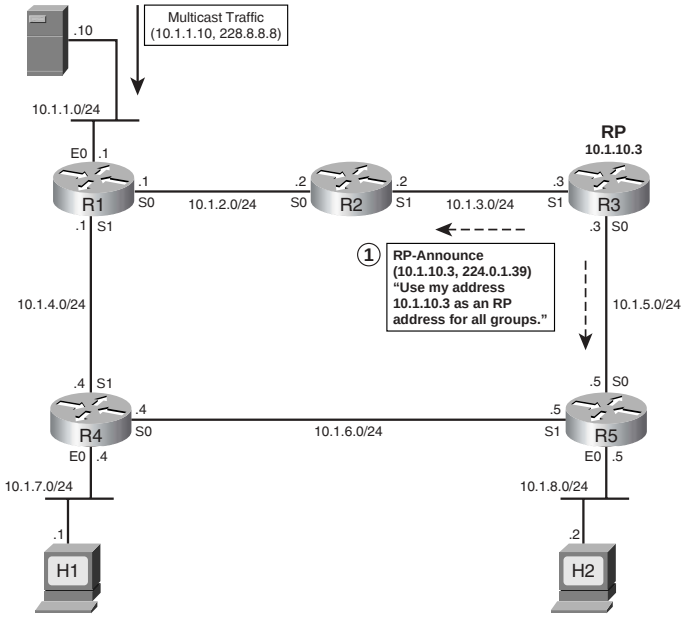
Dynamically Finding the RP Using Auto-RP

Static RP configuration is suboptimal under the following conditions:

- When an enterprise has a large number of PIM-SM routers and the enterprise wants to use many different RPs for different groups, it becomes time consuming and cumbersome to statically configure the IP addresses of many RPs for different groups on all the routers.
- When an RP fails or needs to be changed because a new RP is being installed, it becomes extremely difficult in a statically configured PIM-SM domain to switch over to an alternative RP without considerable downtime.

Auto-RP provides an alternative in which routers dynamically learn the unicast IP address used by each RP. Auto-RP uses a two-step process, which is shown in Figure 20-19 and Figure 20-20. In the first step, the RP sends RP-Announce messages to the reserved multicast address 224.0.1.39, stating that the router is an RP. The RP-Announce message also allows the router to advertise the multicast groups for which it is the RP, thereby allowing some load-balancing of the RP workload among different routers. The RP continues to send these RP-Announce messages every minute.

Figure 20-19 *R3 Sends RP-Announce Messages*

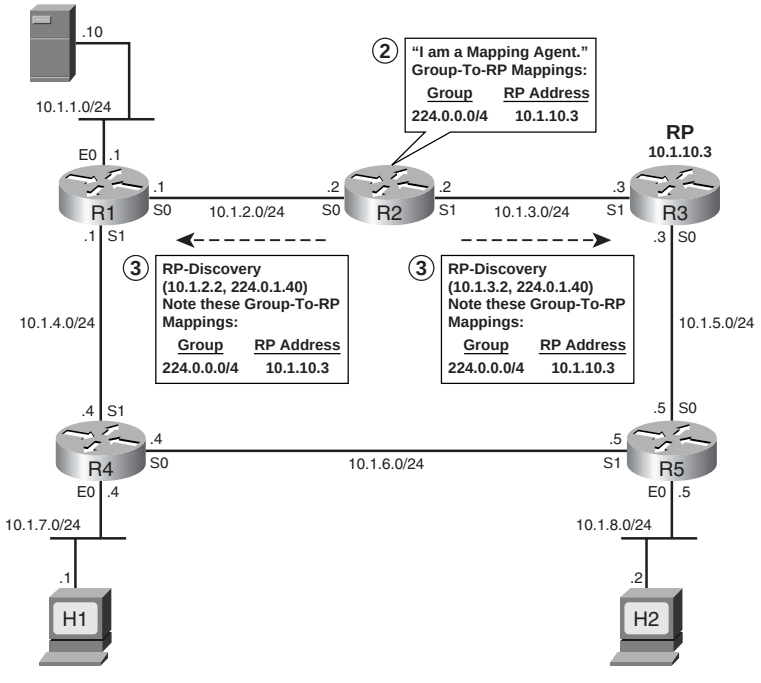


For example, Figure 20-19 shows R3 as an RP that uses Auto-RP. R3 supports all multicast groups in this case. The RP-Announce message is shown as Step 1, to link it with Step 2 in Figure 20-20.

KEY POINT The second step for Auto-RP requires that one router be configured as a mapping agent. The mapping agent is usually the same router that was selected as an RP, but can be a different PIM-SM router. The mapping agent learns all the RPs and the multicast groups they each support. Then, the mapping agent multicasts another message, called RP-Discovery, that identifies the RP for each range of multicast group addresses. This message goes to reserved multicast address 224.0.1.40. It is this RP-Discovery message that actually informs the general router population as to which routers they should use as RPs.

For example, in Figure 20-20, R2 is configured as a mapping agent. To receive all RP-Announce messages, R2 locally joins the well-known Cisco-RP-Announce multicast group 224.0.1.39. In other words, the mapping agent has become a group member for 224.0.1.39 and is listening for the group traffic. When R2 receives the RP-Announce packets shown in Figure 20-19, it examines the packet, creates group-to-RP mappings, and maintains this information in its cache, as shown in Figure 20-20.

Figure 20-20 *R2 Creates Group-to-RP Mappings and Sends Them in RP-Discovery Messages*



At first glance, the need for the mapping agent may not be obvious. Why not just let the RPs announce themselves to all the other routers? Well, if Auto-RP supported only one RP, or even only one RP to support each multicast group, the mapping agent would be a waste of effort. However, to support RP redundancy—in other words, to support multiple RPs that can act as RP for the same multicast group—the Auto-RP mapping agent decides which RP should be used to support each group at the moment. To do so, the mapping agent selects the router with the highest IP address as an RP for the group. (Note that you can also configure multiple mapping agents, for redundancy.)

As soon as Cisco routers are configured with PIM-SM and Auto-RP, they automatically join the well-known Cisco-RP-Discovery multicast group 224.0.1.40. That means they are listening to the group address 224.0.1.40, and when they receive a 224.0.1.40 packet, they learn group-to-RP mapping information and maintain it in their cache. When a PIM-SM router receives an IGMP Join message for a group or PIM-SM Join message from a downstream router, it checks the group-to-RP mapping information in its cache. Then it can proceed as described throughout the PIM-SM explanations in this chapter, using that RP as the RP for that multicast group.

The following list summarizes the steps used by Auto-RP:

KEY POINT

1. Each RP is configured to use Auto-RP and to announce itself and its supported multicast groups via RP-Announce messages (224.0.1.39).
2. The Auto-RP mapping agent, which may or may not also be an RP router, gathers information about all RPs by listening to the RP-Announce messages.
3. The mapping agent builds a mapping table that lists the currently best RP for each range of multicast groups, with the mapping agent picking the RP with the highest IP address if multiple RPs support the same multicast groups.
4. The mapping agent sends RP-Discover messages to 224.0.1.40 advertising the mappings.
5. All routers listen for packets sent to 224.0.1.40 to learn the mapping information and find the correct RP to use for each multicast group.

KEY POINT

Finally, one last small but important point deserves some attention before moving on to BSR. Auto-RP creates a small chicken-and-egg problem in that the purpose of Auto-RP is to find the RPs, but to get the RP-Announce and RP-Discovery messages, PIM-SM routers would need to send a Join toward the RP, which they do not know yet. To overcome this problem, Cisco added a variation of PIM called *sparse-dense mode*. In PIM sparse-dense mode, a router uses PIM-DM rules when it does not know the location of the RP, and PIM-SM rules when it does know the location of the RP. So, under normal conditions with Auto-RP, the routers would use dense mode long enough to learn the group-to-RP mappings from the mapping agent, and then switch over to sparse mode. Also, if any other multicast traffic occurred before the routers learned of the RPs using Auto-RP, the multicast packets would still be forwarded using dense-mode rules. (PIM

sparse-dense mode is configured per interface using the **ip pim sparse-dense-mode** interface subcommand.)

Dynamically Finding the RP Using BSR

Cisco provided the proprietary Auto-RP feature to solve a couple of specific problems. PIM Version 2, which came later, provided a different solution to the same problem, namely the BootStrap Router (BSR) feature. From a very general perspective, BSR works similarly to Auto-RP. Each RP sends a message to another router, which collects the group-to-RP mapping information. That router then distributes the mapping information to the PIM routers. However, any examination of BSR beyond that level of detail shows that these two tools do differ in many ways.

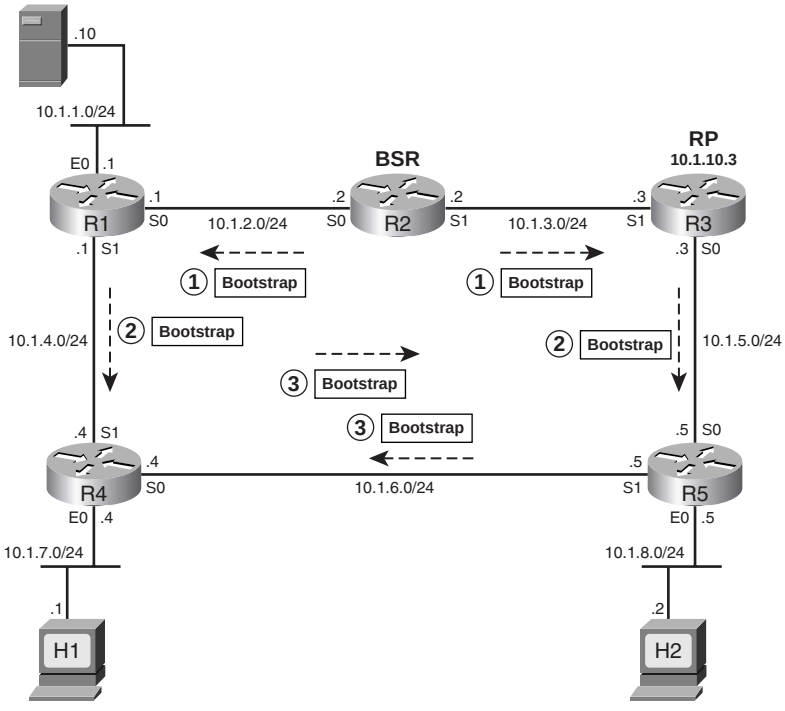
It is helpful to first understand the concept of the bootstrap router, or BSR router, before thinking about the RPs. One router acts as BSR, which is similar to the mapping agent in Auto-RP. The BSR receives mapping information from the RPs, and then it advertises the information to other routers. However, there are some specific differences between the actions of the BSR, and their implications, and the actions of the Auto-RP mapping agent:

- The BSR router does not pick the best RP for each multicast group; instead, the BSR router sends all group-to-RP mapping information to the other PIM routers inside bootstrap messages.
- PIM routers each independently pick the currently best RP for each multicast group by running the same hash algorithm on the information in the bootstrap message.
- The BSR floods the mapping information in a bootstrap message sent to the all-PIM-routers multicast address (224.0.0.13).
- The flooding of the bootstrap message does not require the routers to have a known RP or to support dense mode. (This will be described in more detail in the next few pages.)

Figure 20-21 shows an example, described next, of how the BSR floods the bootstrap message. PIMv2 creates specific rules for BSR bootstrap messages, stating that PIM routers should flood these messages. PIM-SM routers flood bootstrap messages out all non-RPF interfaces, which in effect guarantees that at least one copy of the message makes it to every router. Note that this logic is not dependent on a working dense- or spare-mode implementation. As a result, BSR overcomes the chicken-and-egg problem of Auto-RP.

For example, in Figure 20-21, imagine that R4's s1 interface is its RPF interface to reach R2, and R5's RPF interface to reach R2 is its s0 interface. So, they each forward the bootstrap messages at Step 3 of Figure 20-21. However, because R4 receives the bootstrap message from R5 on one of R4's non-RPF interfaces, R4 discards the packet, thereby preventing loops. R5 also does not forward the bootstrap message any further for the same basic reasons.

Figure 20-21 *BSR Flooding Bootstrap Messages*



The other important part of BSR operation is for each candidate RP (c-RP) to inform the BSR router that it is an RP and to identify the multicast groups it supports. This part of the process with BSR is simple if you keep in mind the following point:

> All PIM routers already know the unicast IP address of the BSR based on the earlier receipt of bootstrap messages.

So, the c-RPs simply send unicast messages, called c-RP Advertisements, to the BSR. These c-RP advertisements include the IP address used by the c-RP, and the groups it supports.

The BSR feature supports redundant RPs and redundant BSRs. As mentioned earlier, the bootstrap message sent by the BSR router includes all candidate RPs, with each router using the same hash algorithm to pick the currently best RP for each multicast group. The mapping information can list multiple RPs that support the same group addresses.

Additionally, multiple BSR routers can be configured. In that case, each candidate BSR (c-BSR) router sends bootstrap messages that include the priority of the BSR router and its IP address. The highest-priority BSR wins, or if a tie occurs, the highest BSR IP address wins. Then, the winning BSR, called the preferred BSR, continues to send bootstrap messages, while the other BSRs monitor those messages. If the preferred BSR's bootstrap messages cease, the redundant BSRs can attempt to take over.

Anycast RP with MSDP

The final tool covered here for finding a router's RP is called Anycast RP with Multicast Source Discovery Protocol (MSDP). Anycast RP is actually an implementation feature more than a new feature with new configuration commands. As will be explained in the upcoming pages, Anycast RP can actually use static RP configuration, Auto-RP, and BSR.

The key differences between using Anycast RP and using either Auto-RP or BSR relate to how the redundant RPs are used. The differences are as follows:

KEY POINT

- **Without Anycast RP**—RP redundancy allows only one router to be the active RP for each multicast group. Load sharing of the collective work of the RPs is accomplished by using one RP for some groups and another RP for other groups.
- **With Anycast RP**—RP redundancy and load sharing can be achieved with multiple RPs concurrently acting as the RP for the same group

The way Anycast RP works is to have each RP use the same IP address. The RPs must advertise this address, typically as a /32 prefix, with its IGP. Then, the other methods of learning an RP—static configuration, Auto-RP, and BSR—all view the multiple RPs as a single RP. At the end of the process, any packets sent to "the" RP are routed per IGP routes to the closest RP. Figure 20-22 shows an example of the process.

Figure 20-22 *Learning the RP Address with Anycast RP*

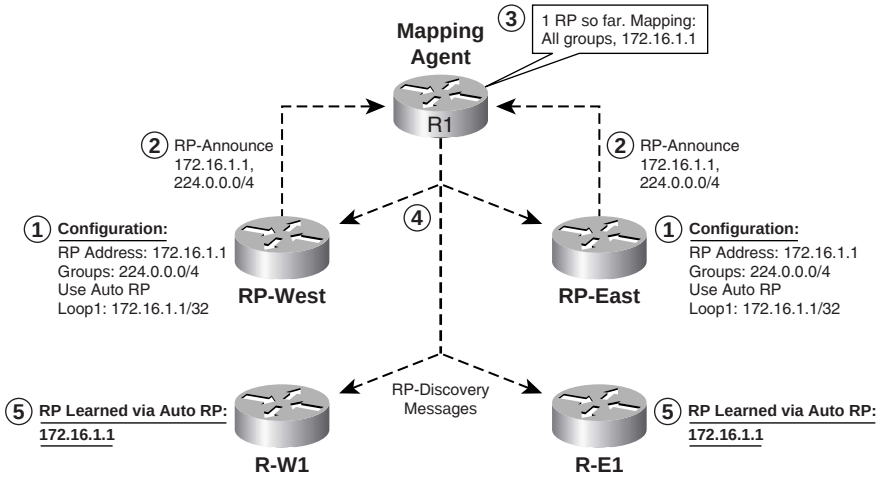


Figure 20-22 shows a design using two RPs (RP-East and RP-West) along with Auto-RP. The steps shown in the figure are as follows:

1. Both RPs are configured with 172.16.1.1/32, and configured to use that IP address for RP functions. In this case, both are configured to be the RP for all multicast groups.

2. Both RPs act as normal for Auto-RP by sending RP-Announce messages to 224.0.1.39.
3. The Auto-RP mapping agent builds its mapping table with a single entry, because it cannot tell the difference between the two RPs, because both use IP address 172.16.1.1.
4. The Auto-RP mapping agent acts as normal, sending an RP-Discovery message to 224.0.1.40. It includes (in this case) a single mapping entry: all groups map to 172.16.1.1.
5. All the routers, including routers R-W1 and R-E1, learn via Auto-RP that the single RP for all groups is 172.16.1.1.

The last step described in the list brings the discussion to the main benefit of Anycast RP. At this point, the core Auto-RP function of advertising the IP address of the RP is complete. Of course, the IP address exists on two routers in Figure 20-22, but it could be more than that in other designs. Because of the IGP routes, when routers in the western part of the network (like R-W1) send packets to the RP at 172.16.1.1, they are actually sending the packets to RP-West. Likewise, when routers in the eastern part of the network (like R-E1) send packets to the RP (172.16.1.1), they are actually sending the packets to RP-East. This behavior is only achieved by using the Anycast RP implementation option beyond simply using Auto-RP.

The two biggest benefits of this design with Anycast RP are as follows:

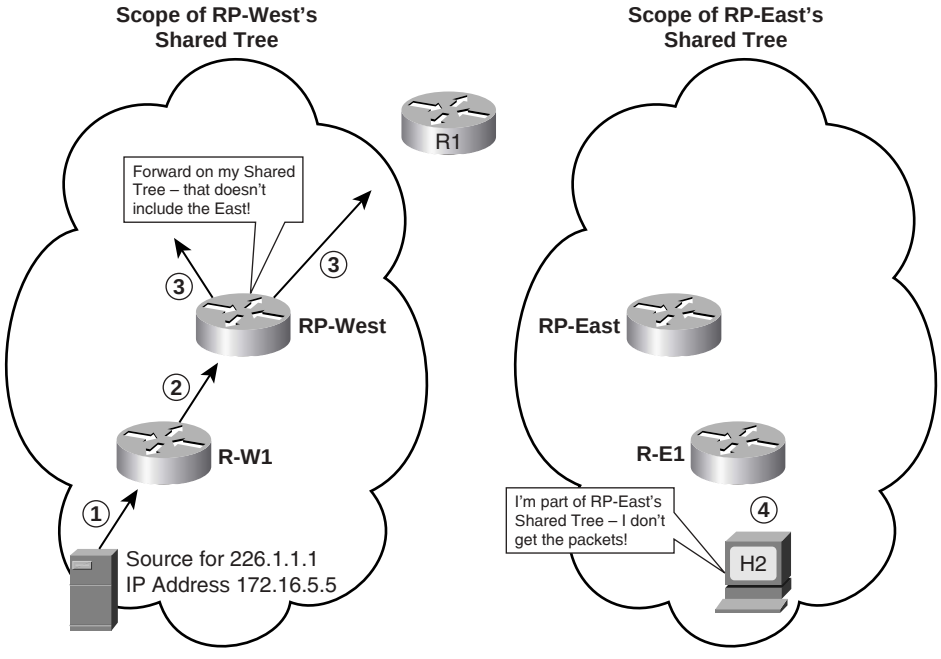
KEY POINT

- Multiple RPs share the load for a single multicast group.
- Recovery after a failed RP happens quickly. If an RP fails, multicast traffic is only interrupted for the amount of time it takes the IGP to converge to point to the other RP sharing the same IP address.

The design of Anycast RP creates a problem that must be overcome using MSDP. The problem relates to the fact that each individual RP builds its own shared tree, but any multicast source sends packets to one of the RPs. For example, Figure 20-23 shows the same network as Figure 20-22, but now with a multicast source in the western part of the network. The routers in the west side of the figure receive the packets as distributed by RP-West via its shared tree. However, the routers in RP-East's shared tree do not get the packets because RP-East never gets the packet sent by the server in the west side.

The solution to this problem is for the RPs to tell each other about all known sources by using MSDP. MSDP allows RPs to send messages to each other, revealing the IP addresses of each source for each multicast group. In Figure 20-23, RP-West could tell RP-East about the multicast source for 226.1.1.1 at unicast IP address 172.16.5.5. Then, RP-East can join the SPT of source 172.16.5.5, group 226.1.1.1, just as it would have done if it had received the multicast traffic directly from 172.16.5.5.

Figure 20-23 *The Anycast RP Problem (Later Solved with MSDP)*



Summary: Finding the RP

This section covers the concepts behind four separate methods for finding the RP. Three are specific configuration features, namely static configuration, Auto-RP, and BSR. The fourth, Anycast RP, actually uses any of the first three methods, but with the design that includes having the RPs use the same unicast IP address to achieve better redundancy features. Table 20-3 summarizes the methods of finding the RP with PIM-SM.

Table 20-3 *Comparison of Methods of Finding the RP*

KEY POINT

Method	RP Details	Mapping Info	Redundant RP Support?	Load Sharing of One Group?
Static	Simple reference to unicast IP address.	—	No	No
Auto-RP	Sends RP-Announce to 224.0.1.39; relies on sparse-dense mode.	Mapping agent sends via RP-Discovery to 224.0.1.40	Yes	No

Table 20-3 *Comparison of Methods of Finding the RP (Continued)*

Method	RP Details	Mapping Info	Redundant RP Support?	Load Sharing of One Group?
BSR	Sends c-RP advertisements as unicasts to BSR IP address; does not need sparse-dense mode.	Sends bootstrap messages flooded over non-RPF path	Yes	No
Anycast RP	Each RP uses identical IP addresses.	Can use Auto-RP or BSR normal processes	Yes	Yes

Bidirectional PIM

PIM-SM works efficiently with a relatively small number of multicast senders. However, in cases with a large number of senders and receivers, PIM-SM becomes less efficient. Bidirectional PIM addresses this relative inefficiency by slightly changing the rules used by PIM-SM.

To appreciate bidirectional PIM, a brief review of PIM-SM's normal operations is useful. While many variations can occur, the following general steps can be used by PIM-SM:

1. The RP builds a shared tree, with itself as the root, for forwarding multicast packets.
2. When a source first sends multicasts, the router nearest the source forwards the multicasts to the RP, encapsulated inside a PIM Register message.
3. The RP joins the source-specific tree for that source by sending a PIM Join toward that source.
4. Later, the routers attached to the same LANs as the receivers can send a PIM Join toward the source to join the SPT for that source.

With bidirectional PIM, the last three steps in this list are not performed. Bidirectional PIM instead follows these steps:

KEY POINT

1. As with normal PIM-SM, the RP builds a shared tree, with itself as the root, for forwarding multicast packets.
2. When a source sends multicasts, the router receiving those multicasts does not use a PIM Register message. Instead, it forwards the packets in the opposite direction of the shared tree, back up the tree toward the RP. This process continues for all multicast packets from the source.
3. The RP forwards the multicasts via the shared tree.
4. All packets are forwarded per Steps 2 and 3. The RP does not join the source tree for the source, and the leaf routers do not join the SPT, either.

The name "bidirectional" comes from Step 2, in which the router near the source forwards packets back up the tree toward the RP. The other direction in the tree is used at Step 3, with the RP forwarding multicasts using the shared tree.

Comparison of PIM-DM and PIM-SM

One of the most confusing parts of the PIM-DM and PIM-SM designs is that it appears that if sources keep sending, and receivers keep listening, there is no difference between the end results of the end-user multicast packet flow using these two options. Once PIM-SM completes its more complicated processes, the routers near the receivers have all joined the SPT to the source, and the most efficient forwarding paths are used for each (S,G) tree.

Although its underlying operation is a bit more complicated, PIM-SM tends to be the more popular option today. PIM-SM's inherent strategy of not forwarding multicasts until hosts request them makes it more efficient during times of low usage. When the numbers of senders and receivers increases, PIM-SM quickly moves to use the SPT—the same SPT that would have been derived using PIM-DM. As such, PIM-SM has become a more popular option for most enterprise implementations today. It has also become a popular option for interdomain multicast as well.

Table 20-4 summarizes the important features of PIM-DM and PIM-SM.

Table 20-4 *Comparison of PIM-DM and PIM-SM*

KEY POINT

Feature	PIM-DM	PIM-SM
Destination address for Version 1 Query messages, and IP protocol number	224.0.0.2 and 2	224.0.0.2 and 2
Destination address for Version 2 Hello messages, and IP protocol number	224.0.0.13 and 103	224.0.0.13 and 103
Default interval for Query and Hello messages	30 seconds	30 seconds
Default Holdtime for Versions 1 & 2	90 seconds	90 seconds
Rule for electing a designated router on a multiaccess network	Router with the highest IP address on the subnet	Router with the highest IP address on the subnet
Main design principle	A router automatically receives the traffic. If it does not want the traffic, it has to say no (send a Prune message) to its sender.	Unless a router specifically makes a request to an RP, it does not receive multicast traffic.

Table 20-4 *Comparison of PIM-DM and PIM-SM (Continued)*

Feature	PIM-DM	PIM-SM
SPT or RPT?	Uses only SPT	First uses RPT and then switches to SPT
Uses Join/Prune messages?	Yes	Yes
Uses Graft and Graft-Ack messages?	Yes	No
Uses Prune Override mechanism?	Yes	Yes
Uses Assert message?	Yes	Yes
Uses RP?	No	Yes
Uses source registration process?	No	Yes

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most Cisco Press *Exam Certification Guides*, this book does not repeat information listed in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 20-5 lists the protocol standards referenced in this chapter.

Table 20-5 *RFC Reference for Chapter 20*

RFC	What It Defines
3973	PIM-DM
3618	MSDP
3446	Anycast RP
2362	PIM-SM
1584	Multicast Extensions to OSPF

Table 20-6 lists some of the most common Cisco IOS commands related to the topics in this chapter and Chapter 19.

Table 20-6 *Command Reference for Chapters 19 and 20*

Command	Command Mode and Description
ip multicast-routing	Global mode; required first command on Cisco routers to use multicasting.
ip pim dense-mode[1]	Interface config mode; configures the interface to use PIM-DM routing protocol.
ip pim sparse-mode[1]	Interface config mode; configures the interface to use PIM-SM routing protocol.
ip pim sparse-dense-mode	Interface config mode; configures the interface to use PIM-SM routing protocol for a group if the RP address is known; otherwise, uses PIM-DM routing protocol.

Table 20-6 *Command Reference for Chapters 19 and 20 (Continued)*

Command	Command Mode and Description
ip igmp version {**1** | **2**}	Interface config mode; sets the IGMP version on an interface. The default is 2.
ip igmp query-interval *seconds*	Interface config mode; changes the interval for IGMP queries sent by the router from the default 60 seconds.
ip igmp query-max-response-time *seconds*	Interface config mode; changes the Max Response Time advertised in IGMP Queries from the default of 10 seconds for IGMPv2 and IGMPv3.
ip igmp join-group *group-address*	Interface config mode; configures a router to join a multicast group. The group-address is a multicast IP address in four-part dotted-decimal notation.
ip multicast boundary *access-list* [**filter-autorp**]	Interface config mode; configures an interface as a multicast boundary for administrative scoping. A numbered or named access list controls the range of group addresses affected by the boundary. (Optional) **filter-autorp** filters Auto-RP messages denied by the boundary ACL.
ip multicast ttl-threshold *ttl-value*	Interface config mode; configures an interface as a multicast boundary for TTL scoping. Time-to-Live value represents number of hops, ranging from 0 to 255. The default value is 0, which means that all multicast packets are forwarded out the interface.
ip cgmp	Interface config mode; enables support for CGMP on an interface.
ip pim version {**1** | **2**}	Interface config mode; sets the PIM version on an interface. The default is 2.
ip pim query-interval *seconds*	Interface config mode; changes the interval for PIMv2 Hello or PIMv1 Router Query messages from the default 60 seconds.
ip pim message-interval *seconds*	Interface config mode; changes the interval for sparse-mode Join/Prune messages from the default 60 seconds.
ip pim spt-threshold {**kbps** | **infinity**} [**group-list** *access-list-number*]	Global mode; specifies the incoming rate for the multicast traffic for a PIM-SM router to switch from RPT to SPT. The default is to switch after the first multicast packet is received. If the **group-list** option is used, the command parameters are applied only to the groups permitted by the access list; otherwise, they are applied to all groups.

continues

Table 20-6 *Command Reference for Chapters 19 and 20 (Continued)*

Command	Command Mode and Description
ip pim rp-address *rp-address* [*access-list*] [**override**]	Global mode; statically configures the IP address of an RP where *rp-address* is a unicast IP address in four-part, dotted notation. (Optional) *access-list* represents a number or name of an access list that defines for which multicast groups the RP should be used. (Optional) **override** indicates that if there is a conflict, the RP configured with this command prevails over the RP learned dynamically by Auto-RP or any other method.
ip pim send-rp-announce *interface-type interface-number* **scope** *ttl-value* [**group-list** *access-list*] [**interval** *seconds*]	Global mode; configures the router to be an RP, and the router sends RP-Announce messages using the Auto-RP method for the interface address selected. Scope represents the TTL. (Optional) **group-list** defines the multicast groups for which this router is RP. (Optional) **interval** changes the announcement frequency from the default 60 seconds.
ip pim send-rp-discovery [*interface-type interface-number*] **scope** *ttl-value*	Global mode; configures the router to be a mapping agent, and the router sends RP-Discovery messages using the Auto-RP method. **scope** represents the TTL. (Optional) The IP address of the interface specified is used as the source address for the messages. The default is to use the IP address of the interface on which the message is sent as the source address.
ip pim rp-announce-filter rp-list *access-list* **group-list** *access-list*	Global mode; configures a mapping agent to filter RP-Announce messages coming from specific RPs. **rp-list** *access-list* specifies a number or name of a standard access list that specifies that this filter is only for the RP addresses permitted in this ACL. **group-list** *access-list* specifies a number or name of a standard access list that describes permitted group addresses. The filter defines that only the group range permitted in the **group-list** *access-list* should be accepted from the RP-Announcements received from the RP addresses permitted by the **rp-list** *access-list*.
show ip igmp groups [*group-name* | *group-address* | *interface-type interface-number*] [**detail**]	User mode; displays the list of multicast groups for which the router has directly connected group members, learned via IGMP.
show ip mroute [*group-address* | *group-name*] [*source-address* | *source-name*] [*interface-type interface-number*] [**summary**] [**count**] [**active** *kbps*]	User mode; displays the contents of the IP multicast routing table.

Table 20-6 *Command Reference for Chapters 19 and 20 (Continued)*

Command	Command Mode and Description
show ip pim neighbor [*interface-type interface-number*]	User mode; displays the list of neighbors discovered by PIM.
show ip pim rp [**mapping** [**elected** | **in-use**] | **metric**] [*rp-address*]	User mode; displays the active RPs associated with multicast groups.
show ip rpf {*source-address* | *source-name*} [**metric**]	User mode; displays the information IP multicasting routing uses to perform the RPF check.
clear ip cgmp [*interface-type interface-number*]	Enable mode; the router sends a CGMP Leave message and instructs the switches to clear all group entries they have cached.
debug ip igmp	Enable mode; displays IGMP messages received and sent, and IGMP-host-related events.
debug ip pim	Enable mode; displays PIM messages received and sent, and PIM-related events.

[1]When you configure any one of these commands on a LAN interface, IGMPv2 is automatically enabled on the interface.

Table 20-7 summarizes important flags displayed in an mroute entry when you use the command **show ip mroute**.

Table 20-7 *mroute Flags*

KEY POINT

Flag	Description
D (dense)	Entry is operating in dense mode.
S (sparse)	Entry is operating in sparse mode.
C (connected)	A member of the multicast group is present on the directly connected interface.
L (local)	The router itself is a member of the multicast group.
P (pruned)	Route has been pruned.
R (RP-bit set)	Indicates that the (S,G) entry is pointing toward the RP. The RP is typically in a pruned state along the shared tree after a downstream router has switched to SPT for a particular source.
F (register flag)	Indicates that the software is registering for a multicast source.

continues

Table 20-7 *mroute Flags*

Flag	Description
T (SPT-bit set)	Indicates that packets have been received on the shortest-path source tree.
J (join SPT)	This flag has meaning only for sparse-mode groups. For (*,G) entries, the J flag indicates that the rate of traffic flowing down the shared tree has exceeded the SPT-Threshold set for the group. This calculation is done once a second. On Cisco routers, the default SPT-Threshold value is 0 kbps. When the J flag is set on the (*,G) entry and the router has a directly connected group member denoted by the C flag, the next (S,G) packet received down the shared tree will trigger a switch over from RPT to SPT for source S and group G. For (S,G) entries, the J flag indicates that the entry was created because the router has switched over from RPT to SPT for the group. When the J flag is set for the (S,G) entries, the router monitors the traffic rate on SPT and switches back to RPT for this source if the traffic rate on the source tree falls below the group's SPT-Threshold for more than 1 minute.

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

dense-mode protocol, RPF check, sparse-mode protocol, RP, multicast scoping, TTL scoping, administrative scoping, PIM-DM, PIM Hello message, designated router, source-based

distribution tree, multicast state information, Join/Prune message, upstream router, downstream router, Graft message, Graft Ack message, Prune Override, Assert message, DVMRP, MOSPF, PIM-SM, source DR, source registration, shared distribution tree, shortest-path tree switchover, PIM-SM (S, G) RP-bit Prune, Auto-RP

Further Reading

Developing IP Multicast Networks, Volume I, by Beau Williamson (Cisco Press, 2000).

Part VII: Security

Chapter 21 Security

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- Security
  - — Access Lists
  - — LAN Security
  - — Device Security/Access
  - — Spoofing

CHAPTER 21

Security

Over the years, the CCIE program has expanded to add several CCIE certifications besides the Routing and Switching track. As a result, some topics previously covered in the Routing and Switching exam have been removed, or shortened, because they are more appropriate for another CCIE track. For example, the CCIE Routing and Switching track formerly covered voice to some degree, but the CCIE Voice track now covers voice to a much deeper level.

The topics in this chapter are certainly covered in more detail in the CCIE Security written and lab exams. However, because security has such an important role in networks, and because many security features relate specifically to router and switch operations, some security details remain within the CCIE Routing and Switching track. This chapter covers many of the core security features related to routers and switches.

"Do I Know This Already?" Quiz

Table 21-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 21-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Router and Switch Device Security	1–3	
Layer 2 Security	4–6	
Layer 3 Security	7–9	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. Consider the following configuration commands, which will be pasted into a router's configuration. Assuming no other AAA configuration or other security-related configuration exists before pasting in this configuration, which of the following is true regarding the process and sequences for authentication of a user attempting to enter privileged mode?

```
enable secret fred
enable authentication wilma
username barney password betty
aaa new-model
aaa authentication enable default group radius local
aaa authentication enable wilma group fred local
aaa authentication login default group radius local
aaa authentication login fred line group radius none
radius-server host 10.1.1.1 auth-port 1812 acct-port 1646
radius-server host 10.1.1.2 auth-port 1645 acct-port 1646
radius-server key cisco
radius-server host 10.1.1.3 auth-port 1812 acct-port 1646
radius-server host 10.1.1.4 auth-port 1645 acct-port 1646
radius-server key cisco
aaa group server radius fred
server 10.1.1.3 auth-port 1645 acct-port 1646
server 10.1.1.4 auth-port 1645 acct-port 1646
line con 0
 password cisco
 login authentication fred
line vty 0 4
 password cisco
```

   a. The user will only need to supply a password of "fred" without a username.

   b. The RADIUS server at either 10.1.1.1 or 10.1.1.2 must approve the username/password supplied by the user.

   c. The RADIUS server at 10.1.1.3 is checked first; if no response, then the server at 10.1.1.4 is checked.

   d. None of these answers is correct.

2. Using the same exhibit and conditions as question 1, which of the following is true regarding the process and sequences for authentication of a user attempting to log in through the console?

   a. A simple password of "cisco" will be required.

   b. The user will supply a username/password, which will be authenticated if either server 10.1.1.1 or 10.1.1.2 returns a RADIUS message approving the user.

   c. The username/password is presented to the RADIUS server at 10.1.1.3 first; if no response, then the server at 10.1.1.4 is checked next.

   d. None of these answers is correct.

3. Using the same exhibit and conditions as question 1, which of the following is true regarding the process and sequences for authentication of a user attempting to log in via Telnet?

   a. A simple password of cisco will be required.

   b. The router will attempt authentication with RADIUS server 10.1.1.1 first; if no response, then 10.1.1.2; if no response, then it will require password cisco.

   c. The router will attempt authentication with RADIUS server 10.1.1.1 first; if no response, then 10.1.1.2; if no response, then it will require a username/password of betty/barney.

   d. The username/password is presented to the RADIUS server at 10.1.1.3 first; if no response, then the server at 10.1.1.4 is checked next.

   e. If neither 10.1.1.1 nor 10.1.1.2 respond, the user cannot be authenticated, and is rejected.

   f. None of the other answers is correct.

4. Which of the following are considered best practices for Layer 2 security?

   a. Inspect ARP messages to prevent hackers from causing hosts to create incorrect ARP table entries.

   b. Enable port security.

   c. Put all management traffic in VLAN 1, but no user traffic.

   d. Configure DTP to use the auto setting.

   e. Shut down unused ports.

5. Assuming a Cisco 3550 switch, which of the following is true regarding the port security feature?

   a. The default maximum number of MACs allowed to be reached on an interface is three.

   b. Sticky-learned MAC addresses are automatically added to the startup configuration once they are learned the first time.

   c. Dynamic (non-sticky) learned MAC addresses are added to the running configuration, but they can be saved using the **copy run start** command.

   d. A port must be set to be a static access or trunking port for port security to be allowed on the interface.

   e. None of the other answers is correct.

6. Which of the following is true regarding the use of IEEE 802.1X for LAN user authentication?
   a. The EAPoL protocol is used between the authenticator and authentication server.
   b. The supplicant is client software on the user's device.
   c. A switch acts in the role of 802.1X authentication server.
   d. The only traffic allowed to exit a currently unauthenticated 802.1X port are 802.1X-related messages.

7. The following ACE is typed into configuration mode on a router: **access-list 1 permit 10.44.38.0 0.0.3.255**. If this statement had instead used a different mask, with nothing else changed, which of the following choices for mask would result in a match for source IP address 10.44.40.18?
   a. 0.0.1.255
   b. 0.0.5.255
   c. 0.0.7.255
   d. 0.0.15.255

8. An enterprise uses a registered class A network. A smurf attack occurs from the Internet, with the enterprise receiving lots of ICMP Echoes, destined to subnet broadcast address 9.1.1.255, which is the broadcast address of an actual deployed subnet (9.1.1.0/24) in the enterprise. The packets all have a source address of 9.1.1.1. Which of the following tools might help mitigate the effects of the attack?
   a. Ensure that the **no ip directed-broadcast** command is configured on the router interfaces connected to the 9.1.1.0/24 subnet.
   b. Configure an RPF check so that the packets would be rejected based on the invalid source IP address.
   c. Routers will not forward packets to subnet broadcast addresses, so there is no need for concern in this case.
   d. Filter all packets sent to addresses in subnet 9.1.1.0/24.

9. Which of the following statements is true regarding the router Cisco IOS Software TCP intercept feature?
   a. Always acts as a proxy for incoming TCP connections, completing the client-side connection, and only then creating a server-side TCP connection.
   b. Can monitor TCP connections for volume and for incomplete connections, as well as serve as a TCP proxy.
   c. If enabled, must operate on all TCP connection requests entering a particular interface.
   d. None of the other answers is correct.

Foundation Topics

Router and Switch Device Security

Securing access to a router or switch CLI is one of the first steps in securing a routed/switched network. Cisco includes several basic mechanisms appropriate for protecting devices in a lab, as well as more robust security features appropriate for devices deployed in production environments. Additionally, these same base authentication features can be used to authenticate dial PPP users. The first section of this chapter examines each of these topics.

Simple Password Protection for the CLI

Figure 21-1 provides a visual reminder of some hopefully familiar details about how users can reach a router's CLI user mode, and move into enable (privileged) mode using the **enable** command.

Figure 21-1 *Router User and Enable Modes*

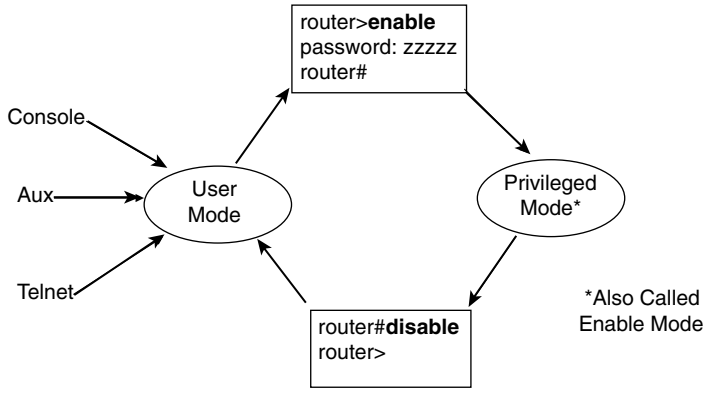


Figure 21-1 shows three methods to reach user mode on a router. The figure also applies to Cisco IOS–based switches, except that Cisco switches do not have auxiliary ports.

Cisco IOS can be configured to require simple password protection for each of the three methods to access user mode. To do so, the **login** line subcommand is used to tell Cisco IOS to prompt the user for a password, and the **password** command defines the password. The configuration mode implies for which of the three access methods the password should be required. Example 21-1 shows a simple example.

Example 21-1 *Simple User Mode CLI Password Protection*

```
! The login and password commands under line con 0 tell the router to supply a password
! prompt, and define the password required at the console port, respectively.
line con 0
 login
 password fred
!
line vty 0 15
 login
 password barney
```

These passwords are stored as clear text in the configuration, but they can be encrypted by including the **service password-encryption** global command. Example 21-2 shows the results of adding this command.

Example 21-2 *Using the* **service password-encryption** *Command*

```
! The service password-encryption global command causes all existing clear-text
! passwords in the running config to be encrypted.
service password-encryption
! The "7" in the password commands means that the following value is the
! encrypted password per the service password-encryption command.
line con 0
 password 7 05080F1C2243
 login
line vty 0 4
 password 7 00071A150754
 login
```

Note that when the **service password-encryption** command is added to the configuration, all clear-text passwords in the running configuration are changed to an encrypted value. The passwords in the startup configuration are not changed until the **copy running-config startup-config** (or **write memory** for all you fellow old-timers out there) command has been used to save the configuration. Also, after disabling password encryption (**no service password-encryption**), passwords are not automatically decrypted—instead, Cisco IOS waits for a password to be changed before listing the password in its unencrypted form.

Note that the encryption used by the **service password-encryption** command is weak. Publicly available tools can decrypt the password. The encryption is useful to prevent the curious from logging into a router or switch, but it provides no real protection against even a hacker with modest ability.

Better Protection of Enable and Username Passwords

The password required by the **enable** command can be defined by either the **enable password** *pw* command or the **enable secret** *pw* command. If both are configured, the **enable exec** command only accepts the password defined in the **enable secret** command.

The password in the **enable password** command follows the same encryption rules as login passwords, only being encrypted if the **service password-encryption** command is configured. However, the **enable secret** password is not affected by **service password-encryption**. Instead, it is always stored as an MD5-hashed value, instead of being encrypted, resulting in a much harder to break password. Example 21-3 shows how Cisco IOS represents this subtle difference in how the password values are stored.

Example 21-3 *Differences in Hashed/Encrypted Enable Passwords*

KEY POINT

```
! The enable password lists a 7 in the output to signify an encrypted value
! per the service password-encryption command; the
! enable secret command lists a 5, signifying an MD5-hashed value.
service password-encryption
!
enable secret 5 $1$GvDM$ux/PhTwSscDNOyNIyr5Be/
enable password 7 070C285F4D064B
```

The **username** *name* **password** *password* command has a feature similar to the **enable secret** command. The **service password-encryption** command encrypts the password listed in the **username** *name* **password** *password* command; however, the **username** *name* **secret** *password* command uses the same MD5 hash as the **enable secret** command to better protect the password. And, as with **enable secret**, a 5 is listed in the command as stored in the configuration—for example, **username barney secret 5 1oMnb$EGf1zE5QPip4UW7TTqQTR**.

User Mode and Privileged Mode AAA Authentication

The term *authentication, authorization, and accounting (AAA)* refers to a variety of common security features. This section focuses on the first "A" in AAA—authentication—and how it is used to manage access to a router or IOS switch's user mode and privileged mode.

The strongest authentication method to protect the CLI is to use a TACACS+ or RADIUS server. The *Cisco Secure Access Control Server (ACS)* is a Cisco Systems software product that can be installed on Unix, Linux, and several Windows platforms, holding the set of usernames and passwords used for authentication. The routers and switches then need to receive the username and password from the user, send it as encrypted traffic to the server, and receive a reply—either accepting or rejecting the user. Table 21-2 summarizes some of the key facts about RADIUS and TACACS+.

Table 21-2 *Comparing RADIUS and TACACS+ for Authentication*

KEY POINT

	RADIUS	TACACS+
Scope of Encryption: packet payload or just the password	Password only	Entire payload
Layer 4 Protocol	UDP	TCP
Well-Known Port/IOS Default Port Used for authentication	1812/1645[1]	49/49
Standard or Cisco-Proprietary	RFC 2865	Proprietary

[1]Radius originally defined port 1645 as the well-known port, which was later changed to port 1812.

Using a Default Set of Authentication Methods

AAA authentication configuration includes commands by which a set of authentication methods is defined. A single *authentication method* is exactly what it sounds like—a way to authenticate a user. For example, one method is to ask a RADIUS server to authenticate a login user; another is to let a router look at a set of locally defined **username** commands. A set of configuration methods represents an ordered list of authentication methods, each of which is tried in order until one of the methods returns an authentication response, either accepting or rejecting the user.

The simplest AAA configuration defines a default set of authentication methods used for all router or switch logins, plus a second set of default authentication methods used by the **enable** command. The defined default login authentication methods apply to all login access—console, Telnet, and aux (routers only). The default authentication methods used by the **enable** command simply dictate what Cisco IOS does when a user types the **enable** command. The overall configuration uses the following general steps:

KEY POINT

Step 1 Enable AAA authentication with the **aaa new-model** global command.

Step 2 If using RADIUS or TACACS+, define the IP address(es) and encryption keys used by the server(s) by using the **radius-server host**, **radius-server key**, **tacacs-server host**, and **tacacs-server key** commands.

Step 3 Define the default set of authentication methods used for all CLI access by using the **aaa authentication login default** command.

Step 4 Define the default set of authentication methods used for enable-mode access by using the **aaa authentication enable default** command.

Example 21-4 shows a sample router configuration using these commands. In this case, two RADIUS servers are configured. One of the servers uses the Cisco IOS default port of 1645, and the other uses the reserved well-known port 1812. Per the following configuration, this router attempts the following authentication:

- When a login attempt is made, Cisco IOS attempts authentication using the first RADIUS server; if there's no response, IOS tries the second RADIUS server; if there's no response, the user is allowed in (authentication mode **none**).
- When any user issues the **enable** command, the router tries the RADIUS servers, in order; if none of the RADIUS servers replies, the router will accept the single username/password configured on the router of **cisco**/**cisco**.

Example 21-4 *Differences in Hashed/Encrypted Enable Passwords*

```
! The next command shows that the enable secret password is still configured,
! but it will not be used. The username command defines a user/password that
! will be used for enable authentication if the RADIUS servers are not reachable.
! Note that the 0 in the username command means the password is not encrypted.
```

Example 21-4 *Differences in Hashed/Encrypted Enable Passwords (Continued)*

```
R1# show running-config
! lines omitted for brevity
enable secret 5 $1$GvDM$ux/PhTwSscDNOyNIyr5Be/
username cisco password 0 cisco
! Next, AAA is enabled, and the default enable and login authentication is
! defined.
aaa new-model
aaa authentication enable default group radius local
aaa authentication login default group radius none
! Next, the two RADIUS servers are configured. The port numbers were omitted when
! the radius-server host 10.1.1.2 command was issued, and IOS filled in its
! default. Similarly, radius-server host 10.1.1.1 auth-port 1812 was issued,
! with IOS adding the accounting port number default into the command.
radius-server host 10.1.1.1 auth-port 1812 acct-port 1646
radius-server host 10.1.1.2 auth-port 1645 acct-port 1646
radius-server key cisco
! Before adding AAA configuration, both the console and vtys had both the login
! and password commands as listed in Example 21-1. The act of enabling AAA
! deleted the login command, which now by default uses the settings on global
! command aaa authentication login default. The passwords remaining below would
! be used only if the aaa authentication login command listed a method of "line."
line con 0
 password cisco
line vty 0 4
 password cisco
```

Using Multiple Authentication Methods

AAA authentication allows reference to multiple servers and to multiple authentication methods so that a user can be authenticated even if one authentication method is not working. The **aaa authentication** command supports up to four methods on a single command. Additionally, there is no practical limit to the number of RADIUS or TACACS+ servers that can be referenced in a RADIUS or TACACS+ server group. The logic used by Cisco IOS when using these methods is as follows:

KEY POINT

- Use the first listed method first; if that method does not respond, move on to the next, and then the next, and so on until a method responds. Use the first-responding-method's decision (allow or reject).
- If a method refers to a set of more than one server, try the first server, with "first" being based on the order of the commands in the configuration file. If no response, move on to the next sequential server, and so on, until a server responds. Use the first-responding-server's decision (allow or reject).
- If no response occurs for any method, reject the request.

For example, Example 21-4 listed RADIUS servers 10.1.1.1 and 10.1.1.2, in that order, so those servers would be checked in that same order. If neither replies, then the next method would be used—**none** for login sessions (meaning automatically allow the user in), and **local** (meaning authenticate based on configured **username** commands).

Table 21-3 lists the authentication methods allowed for login and enable (privileged exec) mode, along with a brief description.

Table 21-3 *Authentication Methods for Login and Enable*

KEY POINT

Method	Meaning
group radius	Use the configured RADIUS servers
group tacacs+	Use the configured TACACS+ servers
group *name*	Use a defined group of either RADIUS or TACACS+ servers
enable	Use the enable password, based on **enable secret** or **enable password** commands
line[1]	Use the password defined by the **password** command in **line** configuration mode
local	Use **username** commands in the local configuration; treats the username as case insensitive, but the password as case sensitive
local-case	Use **username** commands in the local configuration; treats both the username and password as case sensitive
none	No authentication required; user is automatically authenticated

[1]Cannot be used for enable authentication.

Groups of AAA Servers

By default, Cisco IOS automatically groups RADIUS and TACACS+ servers configured with the **radius-server host** and **tacacs-server host** commands into groups, aptly named *radius* and *tacacs+*. The **aaa authentication** command includes the keywords **group radius** or **group tacacs+** to refer to these default groups. By default, all defined RADIUS servers end up in the radius group, and all defined TACACS+ servers end up in the tacacs+ group.

In some cases, particularly with larger-scale dial implementations, a design may call for the separation of different sets of RADIUS or TACACS+ servers. To do so, servers can be grouped by name. Example 21-5 shows an example configuration with two servers in a RADIUS group named fred, and shows how the **aaa authentication** command can refer to the group.

Example 21-5 *Configuring a RADIUS Server Group*

```
! The next three commands create RADIUS group fred. Note that the servers are
! configured inside AAA group config mode, using the server subcommand. Note that
! IOS added the auth-port and acct-port parameters automatically.
```

Example 21-5 *Configuring a RADIUS Server Group (Continued)*

```
R1(config)# aaa group server radius fred
R1(config-group)# server 10.1.1.3 auth-port 1645 acct-port 1646
R1(config-group)# server 10.1.1.4 auth-port 1645 acct-port 1646
! To use group fred instead of the default group, the aaa authentication
! commands need to refer to group fred, as shown next.
aaa new-model
aaa authentication enable default group fred local
aaa authentication login default group fred none
```

Overriding the Defaults for Login Security

The console, vty, and aux (routers only) lines can override the use of the default login authentication methods. To do so, in line configuration mode, the **login authentication** *name* command is used to point to a named set of configuration methods. Example 21-6 shows a named group of configuration methods called **for-console**, **for-vty**, and **for-aux**, with each applied to the related login method. Each of the named groups defines a different set of authentication methods. Example 21-6 shows an example that implements the following requirements:

- **console**—Try the RADIUS servers, and use the line password if no response
- **vty**—Try the RADIUS servers, and use local usernames/passwords if no response
- **aux**—Try the RADIUS servers, and do not authenticate if no response

Example 21-6 *Overriding the Default Login Authentication Method*

```
! The configuration shown here has been added to the configuration from earlier
! examples.
aaa authentication login for-console group radius line
aaa authentication login for-vty group radius local
aaa authentication login for-aux group radius
! The methods are enabled below with the login authentication commands. Note that
! the local passwords still exist on the console and vtys; for the console,
! that password would be used (based on the line keyword in the aaa
! authentication command above) if the RADIUS servers are all nonresponsive.
! However, the vty password command would not be used by this configuration.
line con 0
 password 7 14141B180F0B
 login authentication for-console
line aux 0
 login authentication for-aux
line vty 0 4
 password 7 104D000A0618
 login authentication for-vty
```

PPP Security

As described in Chapter 17, "Synchronous Serial Links and Protocols," PPP provides the capability to use PAP and CHAP for authentication, which is particularly useful for dial applications. Chapter 17 used only the default authentication method for CHAP/PAP—namely, the reliance on a locally configured set of **username** *name* **password** *password* commands.

Cisco IOS supports the use of AAA authentication for PPP using the same general set of commands as used for login authentication. The configuration steps are as follows:

KEY POINT

Step 1 Just as with login authentication, enable AAA authentication with the **aaa new-model** global command.

Step 2 Just as with login authentication, if used, configure RADIUS and/or TACACS+ servers, using the same commands and syntax as used for login and enable authentication.

Step 3 Similar to login authentication, define PPP to use a default set of authentication methods with the **aaa authentication ppp default** command. (The only difference is that the **ppp** keyword is used instead of **login**.)

Step 4 Similar to login authentication, use the **aaa authentication ppp** *list-name method1* [*method2...*] command to create a named group of methods that can be used instead of the default set.

Step 5 To use a named group of authentication methods instead of the default set, use the **ppp authentication** {*protocol1* [*protocol2...*]} *list-name* command. For example, the command **ppp authentication chap fred** references the authentication methods defined by the **aaa authentication ppp fred** command.

Layer 2 Security

The Cisco SAFE Blueprint document (available at http://www.cisco.com/go/safe) suggests a wide variety of best practices for switch security. In most cases, the recommendations depend on one of three general characterizations of the switch ports, as follows:

- **Unused ports**—Switch ports that are not yet connected to any device—for example, switch ports that are pre-cabled to a faceplate in an empty cubicle
- **User ports**—Ports cabled to end-user devices, or any cabling drop that sits in some physically unprotected area
- **Trusted ports or trunk ports**—Ports connected to fully trusted devices, like other switches known to be located in an area with good physical security

The following list summarizes the best practices that apply to both unused and user ports. The common element between these types of ports is that a malicious person can gain access once they get inside the building, without having to gain further access behind the locked door to a wiring closet or data center.

KEY POINT

- Disable unneeded dynamic protocols like CDP and DTP.
- Disable trunking by configuring these ports as access ports.
- Enable BPDU Guard and Root Guard to prevent STP attacks and keep a stable STP topology.
- Use either Dynamic ARP Inspection (DAI) or private VLANs to prevent frame sniffing.
- Enable port security to at least limit the number of allowed MAC addresses, and possibly restrict the port to use only specific MAC addresses.
- Use 802.1X user authentication.
- Use DHCP snooping and IP Source Guard to prevent DHCP DoS and man-in-the-middle attacks.

Besides the preceding recommendations specifically for unused ports and user ports, the Cisco SAFE Blueprint makes the following additional recommendations:

KEY POINT

- For any port (including trusted ports), consider the general use of private VLANs to further protect the network from sniffing, including preventing routers or L3 switches from routing packets between devices in the private VLAN.
- Configure VTP authentication globally on each switch to prevent DoS attacks.
- Disable unused switch ports and place them in an unused VLAN.
- Avoid using VLAN 1.
- For trunks, do not use the native VLAN.

The rest of this section's coverage of switch security addresses the points in these two lists of best practices, with the next subsection focusing on best practices for unused and user ports (based on the first list), and the following subsection focusing on the general best practices (based on the second list).

Switch Security Best Practices for Unused and User Ports

The first three items in the list of best practices for unused and user ports are mostly covered in earlier chapters. For a brief review, Example 21-7 shows an example configuration on a Cisco 3550 switch, with each of these items configured and noted. In this example, fa0/1 is a currently unused port. CDP has been disabled on the interface, but it remains enabled globally, on the

presumption that some ports still need CDP enabled. DTP has been disabled as well, and STP Root Guard and BPDU Guard are enabled.

Example 21-7 *Disabling CDP and DTP and Enabling Root Guard and BPDU Guard*

```
! The cdp run command keeps CDP enabled globally, but it has been disabled on
! fa0/1, the unused port.
cdp run
int fa0/0
 no cdp enable
! The switchport mode access interface subcommand prevents the port from trunking,
! and the switchport nonegotiate command prevents any DTP messages
! from being sent or processed.
 switchport mode access
 switchport nonegotiate
! The last two interface commands enable Root Guard and BPDU Guard, per interface,
! respectively. BPDU Guard can also be enabled for all ports with PortFast
! enabled by configuring the spanning-tree portfast bpduguard enable global
! command.
 spanning-tree guard root
 spanning-tree bpduguard enable
```

Port Security

Switch port security monitors a port to restrict the number of MAC addresses associated with that port in the Layer 2 switching table. It can also enforce a restriction for only certain MAC addresses to be reachable out the port.

To implement port security, the switch adds more logic to its normal process of examining incoming frames. Instead of automatically adding a Layer 2 switching table entry for the source MAC and port number, the switch considers the port security configuration and whether it allows that entry. By preventing MACs from being added to the switch table, port security can prevent the switch from forwarding frames to those MACs on a port.

Port security supports the following key features:

KEY POINT

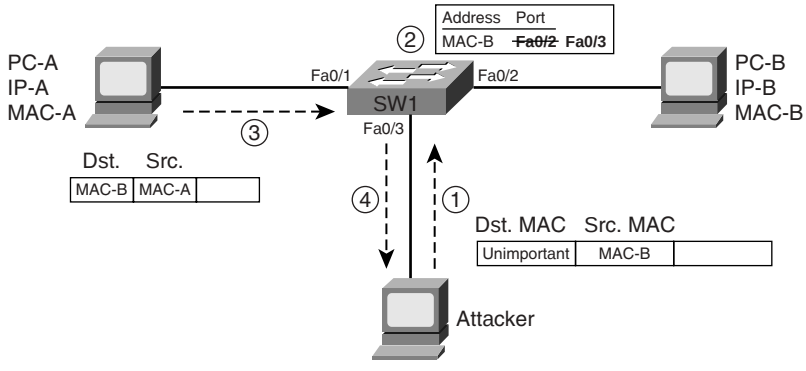
- Limiting the number of MACs that can be associated with the port
- Limiting the actual MAC addresses associated with the port, based on three methods:
  - Static configuration of the allowed MAC addresses
  - Dynamic learning of MAC addresses, up to the defined maximum, where dynamic entries are lost upon reload
  - Dynamically learning but with the switch saving those entries in the configuration (called *sticky learning*)

Port security protects against a couple of types of attacks. Once a switch's forwarding table fills, the switch times out older entries. When the switch receives frames destined for those MACs that are no longer in the table, the switch floods the frames out all ports. An attacker could cause the switch to fill its switching table by sending lots of frames, each with a different source MAC, forcing the switch to time out the entries for most or all of the legitimate hosts. As a result, the switch floods legitimate frames because the destination MACs are no longer in the CAM, allowing the attacker to see all the frames.

An attacker could also claim to be the same MAC address as a legitimate user by simply sending a frame with that same MAC address. As a result, the switch would update its switching table, and send frames to the attacker, as shown in Figure 21-2.

Figure 21-2 *Claiming to Use Another Host's MAC Address*

KEY POINT



1. Attacker sources frame using PC-B's actual MAC.
2. SW1 updates its MAC address table.
3. Another frame is sent to destination MAC-B.
4. SW1 forwards frame to attacker.

Port security prevents both styles of these attacks by limiting the number of MAC addresses and by limiting MACs to particular ports. Port security configuration requires just a few configuration steps, all in interface mode. The commands are summarized in Table 21-4.

Table 21-4 *Port Security Configuration Commands*

KEY POINT

Command	Purpose
switchport mode {**access** | **trunk**}	Port security requires that the port be statically set as either access or trunking
switchport port-security [**maximum** *value*]	Enables port security on an interface, and optionally defines the number of allowed MAC addresses on the port (default 1)
switchport port-security mac-address *mac-address* [**vlan** {*vlan-id* | {**access** | **voice**}}]	Statically defines an allowed MAC address, for a particular VLAN (if trunking), and for either the access or voice VLAN

continues

Table 21-4 *Port Security Configuration Commands (Continued)*

Command	Purpose
switchport port-security mac-address sticky	Tells the switch to remember the dynamically learned MAC addresses
switchport port-security [**aging**] [**violation** {**protect** | **restrict** | **shutdown**}]	Defines the Aging timer and actions taken when a violation occurs

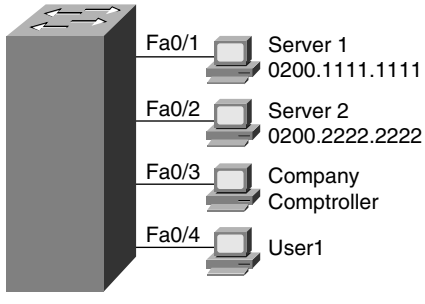
Of the commands in Table 21-4, only the first two are required for port security. With just those two commands, a port allows the first-learned MAC address to be used, but no others. If that MAC address times out of the CAM, another MAC address may be learned on that port, but only one is allowed at a time.

The next two commands in the table allow for the definition of MAC addresses. The third command statically defines the permitted MAC addresses, and the fourth command allows for sticky learning. Sticky learning tells the switch to learn the MACs dynamically, but then add the MACs to the running configuration. This allows port security to be enabled and existing MAC addresses to be learned, but then have them locked into the configuration as static entries simply by saving the running configuration. (Note that the **switchport port-security maximum** *x* command would be required to allow more than one MAC address, with x being the maximum number.)

The last command in the table tells the switch what to do when violations occur. The **protect** option simply tells the switch to perform port security. The **restrict** option tells it to also send SNMP traps and issue log messages regarding the violation. Finally, the **shutdown** option puts the port in a err-disabled state, and requires a **shutdown**/**no shutdown** combination on the port to recover the port's forwarding state.

Example 21-8 shows a sample configuration, based on Figure 21-3. In the figure, Server 1 and Server 2 are the only devices that should ever be connected to interfaces Fast Ethernet 0/1 and 0/2, respectively. In this case, a rogue device has attempted to connect to fa0/1.

Figure 21-3 *Port Security Configuration Example*



Example 21-8 *Using Port Security to Define Correct MAC Addresses Connected to Particular Interfaces*

```
! FA0/1 has been configured to use a static MAC address, defaulting to allow
! only one MAC address.
interface FastEthernet0/1
 switchport mode access
 switchport port-security
 switchport port-security mac-address 0200.1111.1111
! FA0/2 has been configured to use a sticky-learned MAC address, defaulting to
! allow only one MAC address.
interface FastEthernet0/2
 switchport mode access
 switchport port-security
 switchport port-security mac-address sticky
! FA0/1 shows as err-disabled, as a device that was not 0200.1111.1111 tried to
! connect. The default violation mode is shutdown, as shown. It also lists the
! fact that a single MAC address is configured, that the maximum number of MAC
! addresses is 1, and that there are 0 sticky-learned MACs.
fred# show port-security interface fastEthernet 0/1
Port Security : Enabled
Port status : Err-Disabled
Violation mode : Shutdown
Maximum MAC Addresses : 1
Total MAC Addresses : 1
Configured MAC Addresses : 1
Sticky MAC Addresses : 0
Aging time : 0 mins
Aging type : Absolute
SecureStatic address aging : Disabled
Security Violation count : 1
! FA0/2 shows as SecureUp, meaning that port security has not seen any violations
! on this port. Note also at the end of the stanza that the security violations
! count is 0. It lists the fact that one sticky MAC address has been learned.
fred# show port-security interface fastEthernet 0/2
Port Security : Enabled
Port status : SecureUp
Violation mode : Shutdown
Maximum MAC Addresses : 1
Total MAC Addresses : 1
Configured MAC Addresses : 0
Sticky MAC Addresses : 1
Aging time : 0 mins
Aging type : Absolute
SecureStatic address aging : Disabled
Security Violation count : 0
! Note the updated configuration in the switch. Due to the sticky option, the
! switch added the last shown configuration command.
```

KEY POINT

continues

Example 21-8 *Using Port Security to Define Correct MAC Addresses Connected to Particular Interfaces (Continued)*

```
Fred# show running-config
(Lines omitted for brevity)
interface FastEthernet0/2
 switchport mode access
 switchport port-security
 switchport port-security mac-address sticky
 switchport port-security mac-address sticky 0200.2222.2222
```

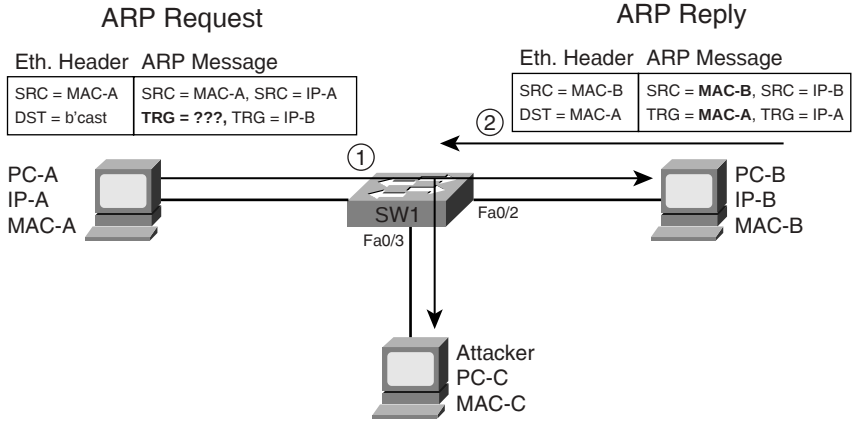
The final part of the example shows that sticky learning updated the running configuration. The MAC address is stored in the running configuration, but it is stored in a command that also uses the **sticky** keyword, differentiating it from a truly statically configured MAC. Note that the switch does not automatically save the configuration in the startup-config file.

Dynamic ARP Inspection

A switch can use DAI to prevent certain types of attacks that leverage the use of IP ARP messages. To appreciate just how those attacks work, you need to keep in mind several detailed points about the contents of ARP messages. Figure 21-4 shows a simple example with the appropriate usage of ARP messages, with PC-A finding PC-B's MAC address.

Figure 21-4 *Normal Use of ARP, Including Ethernet Addresses and ARP Fields*

KEY POINT



1. PC-A Sends ARP Broadcast Looking for IP-B's MAC Address (Target MAC)
2. PC-B Sends LAN Unicast ARP Reply

The ARP message itself does not include an IP header. However, it does include four important addressing fields: the source MAC and IP address of the sender of the message, and the target MAC and IP address. For an ARP request, the target IP lists the IP address whose MAC needs to

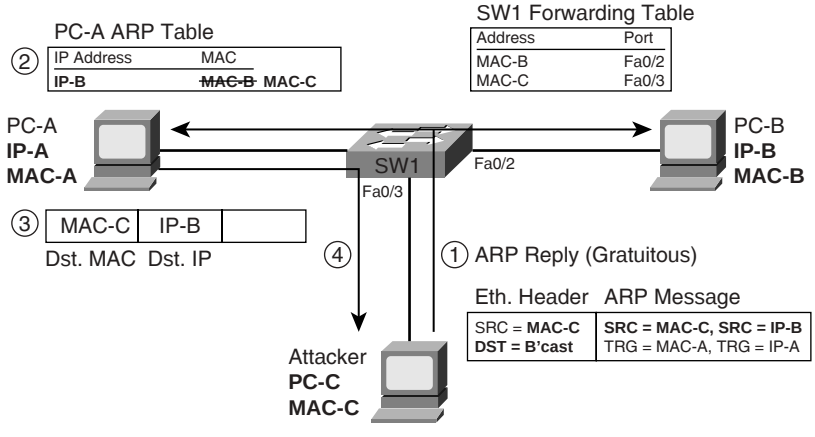
be found, and the target MAC Address field is empty, as that is the missing information. Note that the ARP reply (a LAN unicast) uses the source MAC field to imply the MAC address value—for example, PC-B sets the source MAC inside the ARP message to its own MAC address, and the source IP to its own IP address.

An attacker can form a man-in-the-middle attack in a LAN by creative use of *gratuitous ARPs*. A gratuitous ARP occurs when a host sends an ARP reply, without even seeing an ARP request, and with a broadcast destination Ethernet address. The more typical ARP reply in Figure 21-4 shows the ARP reply as a unicast, meaning that only the host that sent the request will learn an ARP entry; by broadcasting the gratuitous ARP, all hosts on the LAN will learn an ARP entry.

While gratuitous ARPs can be used to good effect, they can also be used by an attacker. The attacker can send a gratuitous ARP, claiming to be an IP address of a legitimate host. All the hosts in the subnet (including routers and switches) update their ARP tables, pointing to the attacker's MAC address—and then later sending frames to the attacker instead of to the true host. Figure 21-5 depicts the process.

Figure 21-5 *Man-in-the-Middle Attack Using Gratuitous ARPs*

KEY POINT



The steps shown in Figure 21-5 can be explained as follows:

1. The attacker broadcasts gratuitous ARP listing IP-B, but with MAC-C as the source IP and MAC.
2. PC-A updates its ARP table to list IP-B's associated address as MAC-C.
3. PC-A sends a frame to IP-B, but with destination MAC MAC-C.
4. SW1 forwards the frame to MAC-C, which is the attacker.

The attack results in other hosts, like PC-A, sending frames meant for IP-B to MAC address MAC-C—the attacker's PC. The attacker then simply forwards another copy of each frame to

PC-B, becoming a man in the middle. As a result, the user can continue to work, and the attacker can gain a much larger amount of data.

Switches use DAI to defeat ARP attacks by examining the ARP messages and then filtering inappropriate messages. DAI considers each switch port to be either untrusted (the default) or trusted, performing DAI messages only on untrusted ports. DAI examines each ARP request or reply (on untrusted ports) to decide if it is inappropriate; if inappropriate, the switch filters the ARP message. DAI determines if an ARP message is inappropriate by using the following logic:

KEY POINT

1. If an ARP reply lists a source IP address that was not DHCP-assigned to a device off that port, DAI filters the ARP reply.
2. DAI uses additional logic like Step 1, but uses a list of statically defined IP/MAC address combinations for comparison.
3. For a received ARP reply, DAI compares the source MAC address in the Ethernet header to the source MAC address in the ARP message. These MACs should be equal in normal ARP replies; if they are not, DAI filters the ARP message.
4. Like Step 3, but DAI compares the destination Ethernet MAC and the target MAC listed in the ARP body.
5. DAI checks for unexpected IP addresses listed in the ARP message, such as 0.0.0.0, 255.255.255.255, multicasts, and so on.

Table 21-5 lists the key Cisco 3550 switch commands used to enable DAI. DAI must first be enabled globally. At that point, all ports are considered to be untrusted by DAI. Some ports, particularly ports connected to devices in secure areas (ports connecting servers, other switches, and so on), need to be explicitly configured as trusted. Then, additional configuration is required to enable the different logic options. For example, DHCP snooping needs to be enabled before DAI can use the DHCP snooping binding database to perform the logic in Step 1 in the preceding list. Optionally, you can configure static IP addresses, or perform additional validation (per the last three points in the preceding list) using the **ip arp inspection validate** command.

Table 21-5 *Cisco IOS Switch Dynamic ARP Inspection Commands*

Command	Purpose
ip arp inspection vlan *vlan-range*	Global command to enable DAI on this switch for the specified VLANs.
[**no**] **ip arp inspection trust**	Interface subcommand that enables (with **no** option) or disables DAI on the interface. Defaults to enabled once the **ip arp inspection** global command has been configured.
ip arp inspection filter *arp-acl-name* **vlan** *vlan-range* [**static**]	Global command to refer to an ARP ACL that defines static IP/MAC addresses to be checked by DAI for that VLAN (Step 2 in the preceding list).

Table 21-5 *Cisco IOS Switch Dynamic ARP Inspection Commands (Continued)*

Command	Purpose
ip arp inspection validate {[**src-mac**] [**dst-mac**] [**ip**]}	Enables additional optional checking of ARP messages (per Steps 3–5 in the preceding list).
ip arp inspection limit {**rate** *pps* [**burst interval** *seconds*] | **none**}	Limits the ARP message rate to prevent DoS attacks carried out by sending a large number or ARPs.

Because DAI causes the switch to perform more work, an attacker could attempt a DoS attack on a switch by sending large numbers of ARP messages. DAI automatically sets a limit of 15 ARP messages per port per second to mitigate that risk; the settings can be changed using the **ip arp inspection limit** interface subcommand.

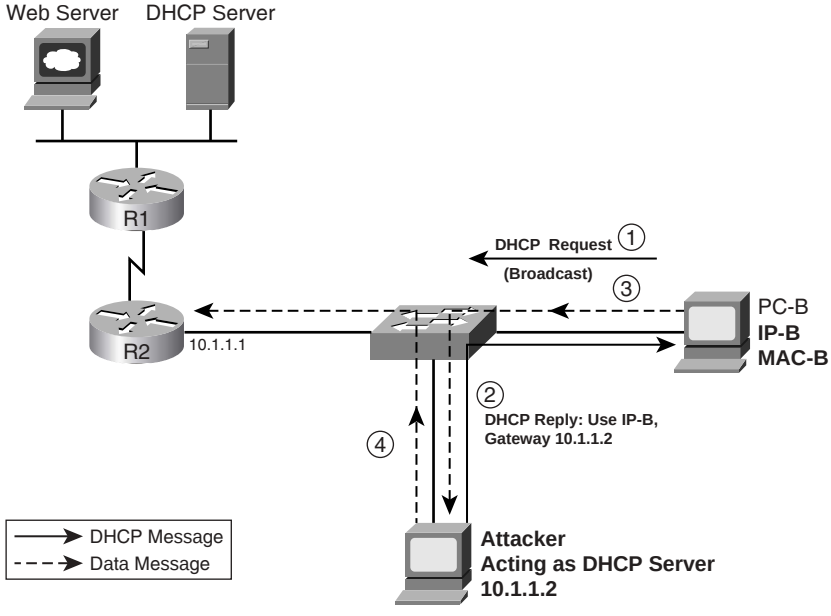
DHCP Snooping

DHCP snooping prevents the damage inflicted by several attacks that use DHCP. DHCP snooping causes a switch to examine DHCP messages and filter those considered to be inappropriate. DHCP snooping also builds a table of IP address and port mappings, based on legitimate DHCP messages, called the *DHCP snooping binding table*. The DHCP snooping binding table can then be used by DAI and by the IP Source Guard feature.

Figure 21-6 shows a man-in-the-middle attack that leverages DHCP. The legitimate DHCP server sits at the main site, whereas the attacker sits on the local LAN, acting as a DHCP server.

Figure 21-6 *Man-in-the-Middle Attack Using DHCP*

KEY POINT



The following steps explain how the attacker's PC can become a man in the middle in Figure 21-6:

1. PC-B requests an IP address using DHCP.
2. The attacker PC replies, and assigns a good IP/mask, but using its own IP address as the default gateway.
3. PC-B sends data frames to the attacker, thinking that the attacker is the default gateway.
4. The attacker forwards copies of the packets, becoming a man in the middle.

NOTE PC-B will use the first DHCP reply, so with the legitimate DHCP server only reachable over the WAN, the attacker's DHCP response should be the first response received by PC-B.

DHCP snooping defeats such attacks for ports it considers to be untrusted. DHCP snooping allows all DHCP messages on trusted ports, but it filters DHCP messages on untrusted ports. It operates based on the premise that only DHCP clients should exist on untrusted ports; as a result, the switch filters incoming DHCP messages that are only sent by servers. So, from a design perspective, unused and unsecured user ports would be configured as untrusted to DHCP snooping.

DHCP snooping also needs to examine the DHCP client messages on untrusted ports, because other attacks can be made using DHCP client messages. DHCP servers identify clients based on their stated *client hardware address* as listed in the DHCP request. A single device could pose as multiple devices by sending repeated DHCP requests, each with a different DHCP client hardware address. The legitimate DHCP server, thinking the requests are from different hosts, assigns an IP address for each request. The DHCP server will soon assign all IP addresses available for the subnet, preventing legitimate users from being assigned an address.

For untrusted ports, DHCP snooping uses the following general logic for filtering the packets:

KEY POINT

1. It filters all messages sent exclusively by DHCP servers.
2. The switch checks DHCP *release* and *decline* messages against the DHCP snooping binding table; if the IP address in those messages is not listed with the port in the DHCP snooping binding table, the messages are filtered.
3. Optionally, it compares a DHCP request's client hardware address value with the source MAC address inside the Ethernet frame.

Of the three entries in this list, the first takes care of the fake DHCP server man-in-the-middle attack shown in Figure 21-6. The second item prevents an attacking host from releasing a legitimate host's DHCP lease, then attempting to request an address and be assigned the same IP address—thereby taking over any existing connections from the original host. Finally, the last item in the list prevents the DoS attack whereby a host attempts to allocate all the IP addresses that the DHCP server can assign in the subnet.

Table 21-6 lists the key configuration commands for configuring DHCP snooping on a Cisco 3550 switch.

Table 21-6 *Cisco IOS Switch Dynamic ARP Inspection Commands*

Command	Purpose
ip dhcp snooping vlan *vlan-range*	Global command to enable DHCP snooping for one or more VLANs
[**no**] **ip dhcp snooping trust**	Interface command to enable or disable a trust level on an interface; **no** version (enabled) is the default
ip dhcp snooping binding *mac-address* **vlan** *vlan-id ip-address* **interface** *interface-id* **expiry** *seconds*	Global command to add static entries to the DHCP snooping binding database
ip dhcp snooping verify mac-address	Interface subcommand to add the optional check of the Ethernet source MAC address to be equal to a DHCP request's client ID
ip dhcp snooping limit rate *rate*	Sets the maximum number of DHCP messages per second to mitigate DoS attacks

IP Source Guard

The Cisco IOS switch IP Source Guard feature adds one more check to the DHCP snooping logic. When enabled along with DHCP snooping, IP Source Guard checks the source IP address of received packets against the DHCP snooping binding database. Alternatively, it checks both the source IP and source MAC addresses against that same database. If the entries do not match, the frame is filtered.

To better appreciate this feature, consider the example DHCP snooping binding database shown in Example 21-9. Note that each of the entries lists the MAC address and IP address, VLAN, and interface. These entries were gleaned from ports untrusted by DHCP snooping, with the DHCP snooping feature building these entries based on the source MAC address and source IP address of the DHCP requests.

Example 21-9 *Sample DHCP Snooping Binding Database*

KEY POINT

```
SW1# show ip dhcp snooping binding
Mac Address          Ip Address        Lease(sec)  Type           VLAN  Interface
------------------   ---------------   ----------  -------------  ----  --------------------
02:00:01:02:03:04    172.16.1.1        3412        dhcp-snooping     3  FastEthernet0/1
02:00:AA:BB:CC:DD    172.16.1.2        4916        dhcp-snooping     3  FastEthernet0/2
```

IP Source Guard is enabled using interface subcommands. To check just the source IP address, use the **ip verify source** interface subcommand; alternately, the **ip verify source port-security**

interface subcommand enables checking of both the source IP and MAC addresses. Optionally, you can use the **ip source binding** *mac-address* **vlan** *vlan-id ip-address* **interface** *interface-id* global command to create static entries that will be used in addition to the DHCP snooping binding database. For example, with IP Source Guard enabled using the **ip verify source** command under interface fa0/1, the only packets allowed coming into interface fa0/1 would be those with source IP address 172.16.1.1.

802.1X Authentication Using EAP

Switches can use IEEE 802.1X to perform user authentication, rather than the types of device authentication performed by many of the other features described in this section. User authentication requires the user to supply a username and password, verified by a RADIUS server, before the switch will enable the switch port for normal user traffic. Requiring a username and password prevents the attacker from simply using someone else's PC to attack the network without first breaking the 802.1X authentication username and password.

IEEE 802.1X defines some of the details of LAN user authentication, but it also uses the Extensible Authentication Protocol (EAP), an Internet standard (RFC 3748), as the underlying protocol used for authentication. EAP includes the protocol messages by which the user can be challenged to provide a password, as well as flows that create one-time passwords (OTPs) per RFC 2289. Figure 21-7 shows the overall flow of LAN user authentication, without the details behind each message.

Figure 21-7 *802.1X for LAN User Authentication*

KEY POINT

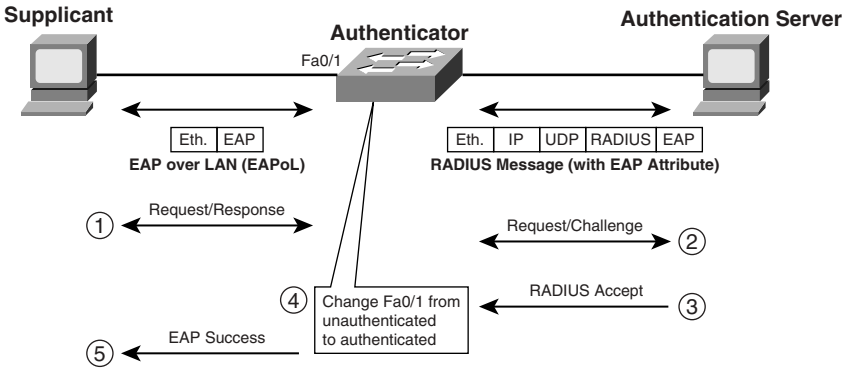


Figure 21-7 introduces a couple of general concepts plus several new terms. First, EAP messages are encapsulated directly inside an Ethernet frame when sent between the 802.1X *supplicant* (user device) and the 802.1X *authenticator* (switch). These frames are called *EAP over LAN (EAPoL)* frames. However, RADIUS expects the EAP message as a data structure called a *RADIUS attribute*, with these attributes sitting inside a normal RADIUS message. To support the two protocols, the switch translates between EAPoL and RADIUS for messages that need to flow between the supplicant and authentication server.

The rest of Figure 21-7 shows a simplistic view of the overall authentication flow. The switch and supplicant create an OTP using a temporary key, with the switch then forwarding the authentication request to the authentication server. The switch, as authenticator, must be aware of the results (Step 3), because the switch has a duty to enable the port once authenticated.

The 802.1X roles shown in Figure 21-7 are summarized as follows:

KEY POINT

- **Supplicant**—The 802.1X driver that supplies a username/password prompt to the user and sends/receives the EAPoL messages
- **Authenticator**—Translates between EAPoL and RADIUS messages in both directions, and enables/disables ports based on the success/failure of authentication
- **Authentication server**—Stores usernames/passwords and verifies that the correct values were submitted before authenticating the user

802.1X switch configuration resembles the AAA configuration covered in the section titled "Using a Default Set of Authentication Methods" earlier in this chapter. The switch configuration treats 802.1X user authentication as another option for AAA authentication, using the following steps:

KEY POINT

Step 1 As with other AAA authentication methods, enable AAA with the **aaa new-model** global command.

Step 2 As with other configurations using RADIUS servers, define the RADIUS server(s) IP address(es) and encryption key(s) using the **radius-server host** and **radius-server key** commands.

Step 3 Similar to login authentication configuration, define the 802.1X authentication method (RADIUS only today) using the **aaa authentication dot1x default** command or, for multiple groups, the **aaa authentication dot1x group** *name* global command.

Step 4 Enable 802.1X globally using the **dot1x system auth-control** global command.

Step 5 Set each interface to use one of three operational settings using the **dot1x port-control** {**auto** | **force-authorized** | **force-unauthorized**} interface subcommand:

- Using 802.1X (**auto**)
- Not using 802.1X, but the interface is automatically authorized (**force-authorized**) (default)
- Not using 802.1X, but the interface is automatically unauthorized (**force-unauthorized**)

Example 21-10 shows a simple 802.1X configuration on a Cisco 3550 switch. The example shows a reasonable configuration based on Figure 21-3 earlier in the chapter, with servers off ports fa0/1 and fa0/2, and two users off ports fa0/3 and fa0/4. Also, consider fa0/5 as an unused port. Note that at the time of writing this chapter, the only available authentication method for 802.1X in Cisco 3550 switches is to use RADIUS.

Example 21-10 *Example Cisco 3550 802.1X Configuration*

```
! The first three commands enable AAA, define that 802.1x should use the RADIUS
! group comprised of all defined RADIUS servers, and enable 802.1X globally.
aaa new-model
aaa authentication dot1x default group radius
dot1x system auth-control
! Next, commands shown previously are used to define the default radius group.
! These commands are unchanged compared to earlier examples.
radius-server host 10.1.1.1 auth-port 1812 acct-port 1646
radius-server host 10.1.1.2 auth-port 1645 acct-port 1646
radius-server key cisco
! The server ports (fa0/1 and fa0/2), inside a secure datacenter, do not require
! 802.1x authentication.
int fa0/1
 dot1x port-control force-authorized
int fa0/2
 dot1x port-control force-authorized
! The client ports (fa0/3 and fa0/4) require 802.1x authentication.
int fa0/3
 dot1x port-control auto
int fa0/4
 dot1x port-control auto
! The unused port (fa0/5) is configured to be in a permanently unauthorized
! state until the dot1x port-control command is reconfigured for this port. As
! such, the port will only allow CDP, STP, and EAPoL frames.
int fa0/5
 dot1x port-control force-unauthorized
```

General Layer 2 Security Recommendations

Recall that the beginning of the "Layer 2 Security" section outlined the Cisco SAFE Blueprint recommendations for user and unused ports and some general recommendations. The general recommendations include configuring VTP authentication globally on each switch, putting unused switch ports in an unused VLAN, and simply not using VLAN 1. The underlying configuration for each of these general recommendations is covered in Chapter 2.

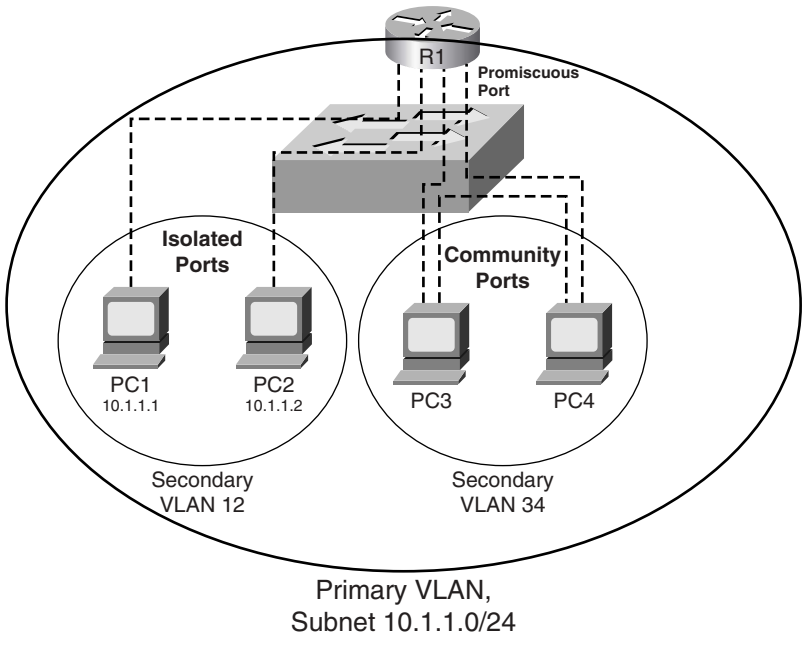
Additionally, Cisco recommends not using the native VLANs on trunks. The reason is that in some cases, an attacker on an access port might be able to hop from its access port VLAN to a trunk's native VLAN by sending frames that begin with multiple 802.1Q headers. This attack has been proven to be ineffective against Cisco switches; however, the attack takes advantage of unfortunate

sequencing of programming logic in how a switch processes frames, so best practices call for not using native VLANs on trunks anyway. Simply put, by following this best practice of not using the native VLAN, even if an attacker managed to hop VLANs, if there are no devices inside that native VLAN, no damage could be inflicted. In fact, Cisco goes on to suggest using a different native VLAN for each trunk, to further restrict this type of attack.

The last general Layer 2 security recommendation covered in this chapter is to consider the use of private VLANs to further restrict traffic. As covered in Chapter 2, private VLANs restrict hosts on some ports from sending frames directly to each other. Figure 21-8 shows the allowed flows as dashed lines. The absence of a line between two devices means that private VLANs would prevent them from communicating. For example, PC1 and PC2 are not allowed to send frames to one another.

Figure 21-8 *Private VLAN Allowed Flows*

KEY POINT



Private VLANs are created with some number of promiscuous ports in the primary VLAN, with other isolated and community ports in one or more secondary VLANs. Isolated ports can send frames only to promiscuous ports, whereas community ports can send frames to promiscuous ports and other community ports in the same secondary VLAN.

Private VLANs could be applied generally for better security by making user ports isolated, only allowing them access to promiscuous ports like routers, servers, or other network services. However, other, more recent additions to Cisco switches, like DHCP snooping, DAI, and IP Source Guard, are typically better choices.

If private VLANs are used, Cisco also recommends additional protection against a trick by which an attacker can use the default gateway to overcome the protections provided by private VLANs. For example, in Figure 21-8, PC1 could send a frame with R1's destination MAC address, but with PC2's destination IP address (10.1.1.2). The switch forwards the frame to R1 because R1's port is promiscuous. R1 then routes the packet to PC2, effectively getting around the private VLAN intent. To solve such a problem, the router simply needs an inbound ACL on its LAN interface that denies traffic whose source and destination IP addresses are in the same local connected subnet. In this example, an **access-list 101 deny ip 10.1.1.0. 0.0.0.255 10.1.1.0 0.0.0.255** command would prevent this attack. (Of course, a few **permit** clauses would also be appropriate for the ACL.)

Layer 3 Security

The Cisco SAFE Blueprint also lists several best practices for Layer 3 security. The following list summarizes the key Layer 3 security recommendations from the SAFE Blueprint.

KEY POINT

1. Enable secure Telnet access to a router user interface, and consider using Secure Shell (SSH) instead of Telnet.
2. Enable SNMP security, particularly adding SNMPv3 support.
3. Turn off all unnecessary services on the router platform.
4. Turn on logging to provide an audit trail.
5. Enable routing protocol authentication.
6. Enable the CEF forwarding path to avoid using flow-based paths like fast switching.

Additionally, RFCs 2827 and 3704 outline other recommended best practices for protecting routers, Layer 3 forwarding (IP routing), and the Layer 3 control plane (routing protocols). RFC 2827 addresses issues with the use of the IP Source and Destination fields in the IP header to form some kind of attack. RFC 3704 details some issues related to how the tools of 2827 may be best deployed over the Internet. Some of the details from those RFCs are as follows:

KEY POINT

1. If a company has registered a particular IP prefix, packets with a source address inside that prefix should not be sent into that autonomous system from the Internet.
2. Packets should never have anything but a valid unicast source IP address, so packets with source IP addresses of loopback (127.0.0.1), 127.x.x.x, broadcast addresses, multicast addresses, and so on, should be filtered.
3. Directed (subnet) broadcasts should not be allowed unless a specific need exists.
4. Packets for which no return route exists to the source IP address of the packet should be discarded (reverse-path-forwarding [RPF] check).

This section does not attempt to cover every portion of Layer 3 security, given the overall purpose of this book. The remainder of this chapter first provides some reference information regarding IP ACLs, which of course are often used to filter packets. This section ends with coverage of some of the more common Layer 3 attacks, and how Layer 3 security can mitigate those attacks.

IP Access Control List Review

A relatively deep knowledge of IP ACL configuration and use is assumed to be pre-requisite knowledge for readers of this book. In fact, many of the examples in the earlier sections of the book did not take the space required to explain the detailed logic of ACLs used in the examples. However, some reference information, as well as statements regarding some of the rules and practices regarding IP ACLs, is useful for general CCIE Routing and Switching exam study. Those details are presented in this section.

First, Table 21-7 lists the majority of the Cisco IOS commands related to IP ACLs.

Table 21-7 *IP ACL Command Reference*

Command	Configuration Mode and Description
access-list *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**]	Global command for standard numbered access lists.
access-list *access-list-number* [**dynamic** *dynamic-name* [**timeout** *minutes*]] {**deny** | **permit**} *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**tos** *tos*] [**log** | **log-input**] [**time-range** *time-range-name*] [**fragments**]	Generic syntax used with a wide variety of protocols. The options beginning with **precedence** are also included for TCP, UDP, and ICMP.
access-list *access-list-number* [**dynamic** *dynamic-name* [**timeout** *minutes*]] {**deny** | **permit**} **tcp** *source source-wildcard* [*operator* [*port*]] *destination destination-wildcard* [*operator* [*port*]] [**established**]	Version of **access-list** command with TCP-specific parameters; identical options exist for UDP, except for the **established** keyword.
access-list *access-list-number* {**deny** | **permit**} **icmp** *source source-wildcard destination destination-wildcard* [*icmp-type* [*icmp-code*] | *icmp-message*]	Version of **access-list** command to match ICMP packets.
access-list *access-list-number* **remark** *text*	Defines a remark.
ip access-list {**standard** | **extended**} *access-list-name*	Global command to create a named ACL.
[*sequence-number*] **permit** | **deny** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**tos** *tos*] [**log** | **log-input**] [**time-range** *time-range-name*] [**fragments**]	Named ACL subcommand used to define an individual entry in the list; similar options for TCP, UDP, ICMP, and others.
ip access-group {*number* | *name* [**in** | **out**]}	Interface subcommand to enable access lists.

continues

Table 21-7 *IP ACL Command Reference (Continued)*

Command	Configuration Mode and Description
access-class *number* | *name* [**in** | **out**]	Line subcommand for standard or extended access lists.
access-list compiled	Global command to compile ACLs on Cisco 7200s/7500s.
ip access-list resequence *access-list-name starting-sequence-number increment*	Global command to redefine sequence numbers for a crowded ACL.
show ip interface [*type number*]	Includes a reference to the access lists enabled on the interface.
show access-lists [*access-list-number* | *access-list-name*]	Shows details of configured access lists for all protocols.
show ip access-list [*access-list-number* | *access-list-name*]	Shows IP access lists.

ACL Rule Summary

Cisco IOS processes the *Access Control Entries (ACEs)* of an ACL sequentially, either permitting or denying a packet based on the first ACE matched by that packet in the ACL. For an individual ACE, all the configured values must match before the ACE is considered a match. Table 21-8 lists several examples of named IP ACL **permit** and **deny** commands that create an individual ACE, along with their meanings.

Table 21-8 *Examples of ACL ACE Logic and Syntax*

KEY POINT

Access List Statement	What It Matches
deny ip any host 10.1.1.1	IP packets with any source IP and destination IP = 10.1.1.1 only.
deny tcp any gt 1023 host 10.1.1.1 eq 23	IP packets with a TCP header, with any source IP, a source TCP port greater than (**gt**) 1023, plus a destination IP of 10.1.1.1, and a destination TCP port of 23.
deny tcp any host 10.1.1.1 eq 23	Same as previous example except that any source port matches, as that parameter was omitted.
deny tcp any host 10.1.1.1 eq telnet	Same results as the previous example; the syntax uses the **telnet** keyword instead of port 23.
deny udp 1.0.0.0 0.255.255.255 lt 1023 any	A packet with a source address in network 1.0.0.0/8, using UDP with a source port less than 1023, with any destination IP address.

The Port Number field is only matchable when the protocol type in an extended IP ACL ACE is UDP or TCP. In these cases, the port number is positional in that the source port matching parameter occurs right after the source IP address, and the destination port parameter occurs right after the destination IP address. Several examples were included in Table 21-8. Table 21-9 summarizes the matching logic used to match UDP and TCP ports.

Table 21-9 *IP ACE Port Matching*

KEY POINT

Keyword	Meaning
gt	Greater than
lt	Less than
eq	Equals
ne	Not equal
range ***x-y***	Range of port numbers, inclusive

ICMP does not use port numbers, but it does include different message types, and some of those even include a further message code. The IP ACL commands allow these to be matched using a rather long list of keywords, or with the numeric message type and message code. Note that these parameters are also positional, following the destination IP address. For example, the named ACL command **permit icmp any any echo-reply** is correct, but the command **permit icmp any echo-reply any** is syntactically incorrect and would be rejected.

Several other parameters can also be checked. For example, the IP precedence bits can be checked, as well as the entire ToS byte. The **established** parameter matches if the TCP header has the ACK flag set—indicative of any TCP segment except the first segment of a new connection setup. (The **established** keyword will be used in an example later in the chapter.) Also, the **log** and **log-input** keywords can be used to tell Cisco IOS to generate periodic log messages when the ACE is matched—one message on initial match, and one every 5 minutes afterwards. The **log-input** option includes more information than the **log** option, specifically information about the incoming interface of the packet that matched the ACE.

For ACL configuration, several facts need to be kept in mind. First, standard ACLs can only match the source IP address field. Numbered standard ACLs are identified with ACL numbers of either 1–99 or 1300–1999, inclusive. Extended numbered IP ACLs range from 100–199 and 2000–2699, again inclusive. Additionally, newly configured ACEs in numbered IP ACLs are always added at the end of the existing ACL, and ACEs in numbered IP ACLs cannot be deleted one at a time. As a result, to insert a line into the middle of a numbered ACL, the entire numbered ACL may need to be deleted (using the **no access-list** *number* global command) and then reconfigured. Named ACLs overcome that problem by using an implied or explicit sequence number, with Cisco IOS listing and processing the ACEs in an ACL in sequence number order.

Wildcard Masks

ACEs use *wildcard masks* (WC masks) to define the portion of the IP address that should be examined. WC masks represent a 32-bit number, with the mask's 0 bits telling Cisco IOS that those corresponding bits in the IP address must be compared when performing the matching logic. The binary 1s in the WC mask tell Cisco IOS that those bits do not need to be compared; as a result, these bits are often called "don't care" bits. Table 21-10 lists several example WC masks, and the implied meanings.

Table 21-10 *Sample Access List Wildcard Masks*

Wildcard Mask	Description
0.0.0.0	The entire IP address must match.
0.0.0.255	Just the first 24 bits must match.
0.0.255.255	Just the first 16 bits must match.
0.255.255.255	Just the first 8 bits must match.
255.255.255.255	Automatically considered to match because all 32 bits are "don't care" bits.
0.0.15.255	Just the first 20 bits must match.
0.0.3.255	Just the first 22 bits must match.
17.44.97.33	A valid WC mask, it means match all bits except bits 4, 8, 11, 13, 14, 18, 19, 24, 27, and 32.

That last entry is unlikely to be useful in an actual production network, but unlike IP subnet masks, the WC mask does not have to list a single unbroken set of 0s and another unbroken string of 1s. A much more likely WC mask is one that matches a particular mask or prefix length. To find a WC mask to match hosts in a known prefix, use the following simple math: in decimal, subtract the subnet mask from 255.255.255.255. The result is the "right" WC mask to match that prefix length. For instance, a subnet mask of 255.255.255.0, subtracted from 255.255.255.255, gives you 0.0.0.255 as a WC mask. This mask only checks the first 24 bits—which in this case is the network and subnet part of the address. Similarly, if the subnet mask is 255.255.240.0, subtracting from 255.255.255.255 gives you 0.0.15.255.

General Layer 3 Security Considerations

This section explains a few of the more common ways to avoid Layer 3 attacks.

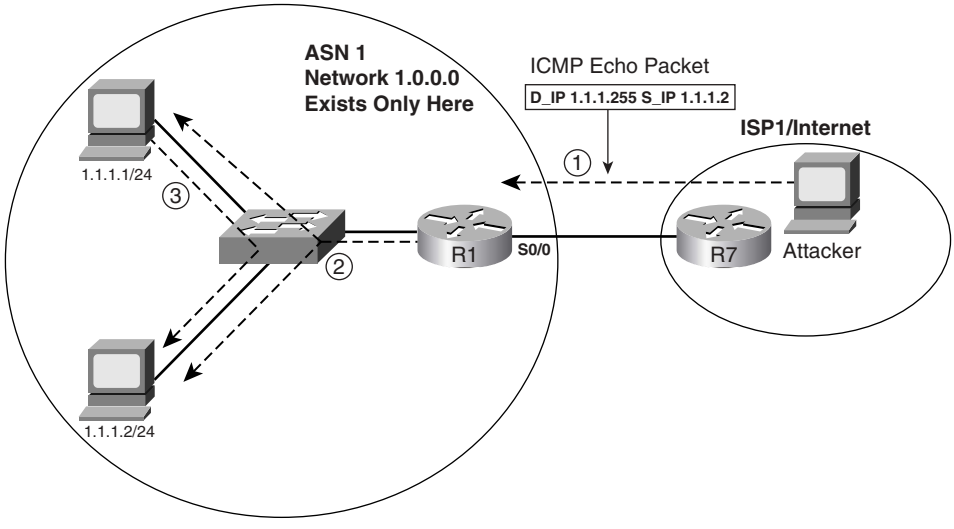
Smurf Attacks, Directed Broadcasts, and RPF Checks

A smurf attack occurs when a host sends a large number of ICMP Echo Requests with some atypical IP addresses in the packet. The destination address is a *subnet broadcast address*, also known as a *directed broadcast address*. Routers forward these packets based on normal matching of the IP routing table, until the packet reaches a router connected to the destination subnet. This

final router then forwards the packet onto the LAN as a LAN broadcast, sending a copy to every device. Figure 21-9 shows how the attack develops.

Figure 21-9 *Smurf Attack*

KEY POINT



1. Attacker sends packet destined to subnet broadcast, source 1.1.1.2 (for secondary attack).
2. R1 forwards packet as LAN broadcast.
3. R1 replies with ICMP echo reply packet sent to 1.1.1.2.

The other feature of a smurf attack is that the source IP address of the packet sent by the attacker is the IP address of the attacked host. For example, in Figure 21-9, many hosts may receive the ICMP Echo Request at Step 2. All those hosts then reply with an Echo Reply, sending it to 10.1.1.2—the address that was the source IP address of the original ICMP Echo at Step 1. Host 10.1.1.2 receives a potentially large number of packets.

Several solutions to this problem exist. First, as of Cisco IOS Software version 12.0, IOS defaults each interface to use the **no ip directed-broadcast** command, which prevents the router from forwarding the broadcast onto the LAN (Step 2 in Figure 21-9). Also, a reverse-path-forwarding (RPF) check could be enabled using the **ip verify unicast source reachable-via** {**rx** | **any**} [**allow-default**] [**allow-self-ping]** [*list*] interface subcommand. This command tells Cisco IOS to examine the source IP address of incoming packets on that interface. Two styles of check can be made with this command:

KEY POINT

- **Strict RPF**—Using the **rx** keyword, the router checks to see if the matching route uses an outgoing interface that is the same interface on which the packet was received. If not, the packet is discarded. (An example scenario using Figure 21-9 will be explained shortly.)
- **Loose RPF**—Using the **any** keyword, the router checks for any route that can be used to reach the source IP address.

The command can also ignore default routes when it performs the check (default) or use default routes when performing the check by including the **allow-default** keyword. Also, although not recommended, the command can trigger a ping to the source to verify connectivity. Finally, the addresses for which the RPF check is made can be limited by a referenced ACL.

For example, in Figure 21-9, if R1 used strict RPF on s0/0, it would notice that its route to reach 1.1.1.2 (the source IP address of the packet at Step 1) did not refer to s0/0 as the outgoing interface—thereby discarding the packet. However, with loose RPF, R1 would have found a connected route that matched 1.1.1.2, so it would have allowed the packet through. Finally, given that AS1 should never receive packets with source addresses in network 1.0.0.0, as it owns that entire class A network, R1 could simply use an inbound ACL to discard any packets sourced from 1.0.0.0/8 as they enter s0/0 from the Internet.

Fraggle attacks use similar logic as smurf attacks, but instead of ICMP, fraggle attacks use the UDP Echo application. These attacks can be defeated using the same options as listed for smurf attacks.

Inappropriate IP Addresses

Besides smurf and fraggle attacks, other attacks involve the use of what can be generally termed inappropriate IP addresses, both for the source IP address and destination IP address. By using inappropriate IP addresses, the attacker can remain hidden and elicit cooperation of other hosts to create a distributed denial-of-service (DDoS) attack.

One of the Layer 3 security best practices is to use ACLs to filter packets whose IP addresses are not appropriate—for instance, the smurf attack listed a valid source IP address of 1.1.1.2, but packets with that source address should never enter AS1 from the Internet. The Internet Assigned Numbers Authority (IANA) manages the assignment of IP prefix ranges. It lists the assigned ranges in a document found at http://www.iana.org/assignments/ipv4-address-space. A router can then be configured with ACLs that prevent packets based on known assigned ranges and on known unassigned ranges. For example, in Figure 21-9, an enterprise router should never need to forward a packet onto the Internet if that packet has a source IP address from another company's registered IP prefix. In the smurf attack case, such an ACL used at the attacker's ISP would have prevented the first packet from getting to AS1.

Routers should also filter packets that use IP addresses that should be considered bogus or inappropriate. For example, a packet should never have a broadcast or multicast source IP address in normal use. Also, an enterprise router should never receive a packet from an ISP with that packet's source IP address being a private network per RFC 1918. Additionally, that same router should not receive packets sourced from IP addresses in ranges currently unallocated by IANA. These types of IP addresses are frequently called *bogons*, which is a derivation of the word bogus.

Creating an ACL to match these bogon IP addresses is not particularly difficult, but it does require a lot of administrative effort, particularly to update it based on changes to IANA's assigned

prefixes. You can use freeware called the Router Audit Tool (RAT) that makes recommendations for router security, including bogon ACLs. You can also use the Cisco IOS *AutoSecure* feature, which automatically configures ACLs to prevent the use of such bogus IP addresses.

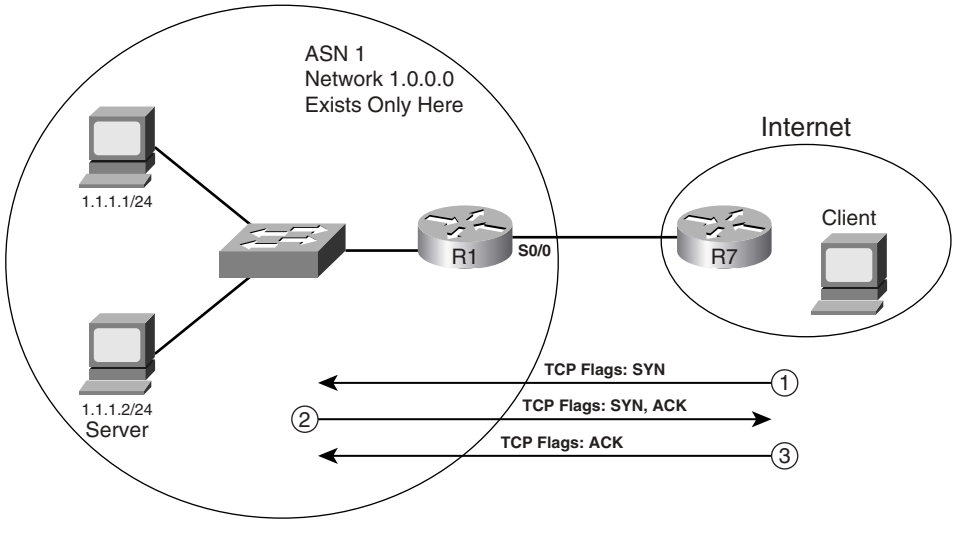
TCP SYN Flood, the Established Bit, and TCP Intercept

A TCP SYN flood is an attack directed at servers by initiating large numbers of TCP connections, but not completing the connections. Essentially, the attacker initiates many TCP connections, each with only the TCP SYN flag set, as usual. The server then sends a reply (with TCP SYN and ACK flags set)—but then the attacker simply does not reply with the expected third message in the three-way TCP connection setup flow. The server consumes memory and resources while waiting on its timeouts to occur before clearing up the partially initialized connections. The server might also reject additional TCP connections, and load balancers in front of a server farm might unbalance the load of actual working connections as well.

Stateful firewalls can prevent TCP SYN attacks. Both the Cisco PIX Firewall and the Cisco IOS Firewall feature set can be used to do this. The methods used are not part of the CCIE Routing and Switching written exam, but instead are covered in the CCIE Security exam; the impact of TCP SYN attacks can be reduced or eliminated by using a few other tools in Cisco IOS.

One way to prevent SYN attacks is to simply filter packets whose TCP header shows only the SYN flag set—in other words, filter all packets that are the first packet in a new TCP connection. In many cases, a router should not allow TCP connections to be established by a client on one side to a server on the other, as shown in Figure 21-10. In these cases, filtering the initial TCP segment prevents the SYN attack.

Figure 21-10 *Example Network: TCP Clients in the Internet*



Cisco IOS ACLs cannot directly match the TCP SYN flag. However, an ACE can use the **established** keyword, which matches TCP segments that have the ACK flag set. The **established** keyword essentially matches all TCP segments except the very first TCP segment in a new connection. Example 21-11 shows the configuration that would be used on R1 to deny new connection requests from the Internet into the network on the left.

Example 21-11 *Using an ACL with the* **established** *Keyword*

```
! The first ACE matches TCP segments that are not the first segment, and permits
! them. The second ACE matches all TCP segment between the same set of IP
! addresses, but because all non-initial segments have already been matched, the
! second ACE only matches the initial segments.
ip access-list extended prevent-syn
 permit tcp any 1.0.0.0 0.255.255.255 established
 deny tcp any 1.0.0.0 0.255.255.255
 permit (whatever)
!
interface s0/0
 ip access-group prevent-syn in
```

The ACL works well when clients outside a network are not allowed to make TCP connections into the network. However, in cases where some inbound TCP connections are allowed, this ACL cannot be used. Another Cisco IOS feature, called *TCP intercept*, provides an alternative that allows TCP connections into the network, but monitors those TCP connections for TCP SYN attacks.

TCP intercept operates in one of two different modes. In *watch mode*, it keeps state information about TCP connections that match a defined ACL. If a TCP connection does not complete the three-way handshake within a particular time period, TCP intercept sends a TCP reset to the server, cleaning up the connection. It also counts the number of new connections attempted over time, and if a large number occurs in 1 second ("large" defaulting to 1100), the router temporarily filters new TCP requests to prevent a perceived SYN attack.

In *intercept mode*, the router replies to TCP connection requests instead of forwarding them to the actual server. Then, if the three-way handshake completes, the router creates a TCP connection between itself and the server. At that point, the router knits the two connections together. This takes more processing and effort, but it provides better protection for the servers.

Example 21-12 shows an example using TCP intercept configuration, in watch mode, plus a few changes to its default settings. The example allows connections from the Internet into AS1 in Figure 21-10.

Example 21-12 *Configuring TCP Intercept*

```
! The following command enables TCP intercept for packets matching ACL
! match-tcp-from-internet. Also, the mode is set to watch, rather than the
! default of intercept. Finally, the watch timeout has been reset from the
! default of 30 seconds; if the TCP connection remains incomplete as of the
! 20-second mark, TCP intercept resets the connection.
ip tcp intercept-list match-tcp-from-internet
ip tcp intercept mode watch
ip tcp intercept watch-timeout 20
! The ACL matches packets sent into 1.0.0.0/8 that are TCP. It is referenced by
! the ip tcp intercept-list command listed above.
ip access-list extended match-tcp-from-internet
 permit tcp any 1.0.0.0 0.255.255.255
! Note below that the ACL is not enabled on any interfaces.
interface s0/0
! Note: there is no ACL enabled on the interface!
```

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most Cisco Press *Exam Certification Guides*, this book does not repeat information presented in the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 21-11 lists some of the key protocols covered in this chapter.

Table 21-11 *Protocols and Standards for Chapter 21*

Name	Standard
RADIUS	RFC 2865
Port-Based Network Access Control	IEEE 802.1X
EAP	RFC 3748
A One-Time Password System	RFC 2289
Router Security	RFCs 2827 and 3704

Table 21-12 lists some of the most popular router IOS commands related to the topics in this chapter.

Table 21-12 *Router IOS Commands Related to Chapter 21*

Command	Description
service password-encryption	Global command to enable simple encryption of passwords
server *ip-address* [**auth-port** *port-number*] [**acct-port** *port-number*]	Global command to define a RADIUS server and ports used
aaa group server radius | **tacacs+** *group-name*	Global command to create the name of a group of AAA servers
server *ip-address*	AAA group mode; defines a TACACS+ server
server *ip-address* [**auth-port** *port-number*] [**acct-port** *port-number*]	AAA group mode; defines a RADIUS server and ports used
radius-server host {*hostname* | *ip-address*} [**auth-port** *port-number*] [**acct-port** *port-number*] [**timeout** *seconds*] [**retransmit** *retries*] [**key** *string*] [**alias**{*hostname* | *ip-address*}]	Global mode; defines details regarding a single RADIUS server

Table 21-12 *Router IOS Commands Related to Chapter 21 (Continued)*

Command	Description
radius-server key {**0** *string* | *7 string* | *string*}	Global mode; defines the key used to encrypt RADIUS passwords
tacacs-server host {*host-name* | *host-ip-address}* [**key** *string*] [**nat**] [**port** [*integer*]] [**single-connection**] [**timeout** [*integer]*]	Global mode; defines details regarding a single TACACS+ server
tacacs-server key *key*	Global mode; defines the key used to encrypt the TACACS+ payload
aaa authentication enable default *method1* [*method2...*]	Global mode; defines the default authentication methods used by the **enable** command
aaa authentication login {**default** | *list-name*} *method1* [*method2...*]	Global mode; defines the default authentication methods used by console, vty, and aux logins
aaa authentication ppp {**default** | *list-name*} *method1* [*method2...*]	Global mode; defines the default authentication methods used by PPP
aaa new-model	Global mode; enables AAA globally in a router/switch
login authentication {**default** | *list-name*}	Line mode; defines the AAA group to use for authentication
ppp authentication {*protocol1* [*protocol2...*]} [**if-needed**] [*list-name* | **default**] [**callin**] [**one-time**] [**optional**]	Interface mode; defines the type of AAA authentication used by PPP
auto secure [**management** | **forwarding**] [**no-interact**]	Global mode; automatically configures IOS with Cisco's recommended device security configuration
enable password [**level** *level*] {*password* | [*encryption-type*] *encrypted-password*}	Global mode; defines the enable password
enable secret [**level** *level*] {*password* | [*encryption-type*] *encrypted-password*}	Global mode; defines the enable password that is MD5 hashed
ip verify unicast reverse-path [*list*]	Interface subcommand; enables strict RPF
ip verify unicast source reachable-via {**rx** | **any**} [**allow-default**] [**allow-self-ping**] [*list*]	Interface subcommand; enables strict or loose RPF
username *name* {**nopassword** | **password** *password*}	Global mode; defines local usernames and passwords
username *name* **secret** {[**0**] *password* | **5** *encrypted-secret*}	Global mode; defines local usernames and MD5-hashed passwords

continues

Table 21-12 *Router IOS Commands Related to Chapter 21 (Continued)*

Command	Description
ip tcp intercept list *access-list-number*	Global mode; identifies an ACL to be used by TCP intercept
ip tcp intercept mode {**intercept** | **watch**}	Global mode; defines the mode used by TCP intercept
ip tcp intercept watch-timeout *seconds*	Global mode; defines the timeout used before acting to clean up an incomplete TCP connection

Table 21-13 lists some of the Cisco 3550 switch commands used in this chapter. Also, refer to Tables 21-4 through 21-7. Note that all commands in Table 21-13 were copied from the version 12.2(25)SEB 3550 Command Reference at Cisco.com; the syntax may vary on different Cisco IOS–based switches.

Table 21-13 *Catalyst IOS Commands Related to Chapter 21*

Command	Description
spanning-tree guard root	Interface mode; enables Root Guard.
aaa authentication dot1x {**default**} *method1*	Global mode; defines the default authentication method for 802.1X. Only one method is available, because only RADIUS is supported.
arp access-list *acl-name*	Global command; creates an ARP ACL with the stated name.
dot1x system-auth-control	Global command that enables 802.1x.
dot1x port-control {**auto** | **force-authorized** | **force-unauthorized**}	Interface subcommand to define 802.1x actions on the interface.
dot1x timeout {**quiet-period** *seconds* | **reauth-period** *seconds* | **server-timeout** *seconds* | **supp-timeout** *seconds* | **tx-period** *seconds*}	Global command to set 802.1x timers.

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation

Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

AAA, authentication method, RADIUS, TACACS+, MD5 hash, enable password, enable secret, ACS, SAFE Blueprint, DAI, port security, IEEE 802.1X, DHCP snooping, IP Source Guard, man-in-the-middle attack, sticky learning, fraggle attack, DHCP snooping binding database, EAP, EAPoL, OTP, Supplicant, authenticator, authentication server, smurf attack, TCP SYN flood, TCP intercept, ACE

Refer to the CD-based glossary to check your answers.

Further Reading

The topics in this chapter tend to be covered in slightly more detail in CCNP Switching exam preparation books. For more details on these topics, refer to the *CCNP BCMSN Exam Certification Guide* and the *CCNP Self-Study: CCNP BCMSN Exam Certification Guide*, Second Edition, listed in the introduction to this book.

CCSP Self-Study: Securing Cisco IOS Networks (SECUR), by John Roland

Network Security Principles and Practices, by Saadat Malik

Network Security Architectures, by Sean Convery

Cisco SAFE Blueprint Introduction: http://www.cisco.com/go/safe

Part VIII: Enterprise Wireless Mobility

Chapter 22 IEEE 802.11 Fundamentals

Chapter 23 Wireless LAN Solutions

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- Enterprise Wireless Mobility
  - — Standards
  - — Hardware
  - — SWAN
  - — RF Troubleshooting
  - — VoWLAN
  - — Products

CHAPTER 22

IEEE 802.11 Fundamentals

To effectively deploy and support wireless LANs, you need at least a fundamental understanding of the IEEE 802.11 standard, which defines the operation and configuration settings of a wireless LAN. This knowledge allows you to study more advanced topics dealing with wireless LAN solutions and implementation strategies. This chapter takes a look at how 802.11-based devices operate and what you should consider when configuring a wireless LAN.

"Do I Know This Already?" Quiz

Table 22-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 22-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
802.11 Physical Layer Standards	1	
Wireless System Configuration	2	
Wireless Hardware Components	3	
Infrastructure Mode Operation	4	
Ad Hoc Mode Operation	5	
Wireless Configuration Parameters	6–7	
Wireless Medium Access	8	
Wireless Security	9	
RF Signal Concepts	10	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. Which 802.11 physical layer offers the highest capacity?
   a. 802.11a
   b. 802.11b
   c. 802.11g
   d. 802.11a and 802.11g offer the same capacity.

2. Which of the following is correct regarding 802.11 data frames traveling through an infrastructure wireless LAN?
   a. Data frames travel directly from one wireless user to another wireless user without going through the access point.
   b. Data frames going from one wireless user to another user must travel through the access point.
   c. Infrastructure wireless LANs do not carry 802.11 data frames.
   d. The 802.11 data frames are sent through the access point, and a wireless server coordinates transfer of the frames to the wireless user.

3. What is a disadvantage of using a wireless LAN repeater when deploying wireless LANs?
   a. Repeaters have much less range than access points.
   b. The data rates of repeaters are much lower than the data rates of wireless user devices.
   c. A repeater doubles the number of data frames sent over the network.
   d. Repeaters do not implement the entire 802.11 standard, so they are not interoperable with most access points and radio cards.

4. Why is the overhead relatively high and the corresponding throughput low when transmitting data over a wireless LAN?
   a. Every 802.11 data frame requires the destination to send an acknowledgement.
   b. All 802.11 data frame transmissions require the use of the RTS/CTS function.
   c. The 802.11 data frames carry highly redundant information in the frame body to counter errors from RF interference.

5. How are beacons sent in an ad hoc wireless LAN?
   a. No beacons are transmitted in an ad hoc network because the stations operate independently.
   b. The first 802.11 station forming an ad hoc network assumes the responsibility for sending beacons and continues sending beacons when other stations associate with the ad hoc network.
   c. Each 802.11 station will take turns sending beacons based on the order in which they associated with the ad hoc network.

d. After receiving a beacon, each radio card will wait a random period of time and send a beacon if a beacon is not heard from another station.

6. Which 802.11 configuration parameter is set only in the access point, and not in the radio cards, with an infrastructure wireless LAN?

a. RF channel

b. SSID

c. Fragmentation

d. RTS/CTS

e. Transmit power

7. How often are multicast frames sent to 802.11 stations implementing power-save mode?

a. After every beacon

b. After every DTIM interval

c. Never

d. Immediately after each data frame

8. When does an 802.11 station access the medium when using the distributed coordination function?

a. Immediately after the station senses no other traffic on the medium

b. After 20 μs when the station senses an idle medium

c. After the network allocation vector value is zero

d. When the access point polls the station

9. Which security mechanism is best for operating over public wireless LANs?

a. 802.11i

b. WPA2

c. VPN

d. WEP

10. What does using a 6-dBi antenna (as compared to a 3-dBi antenna) on an access point provide?

a. It doubles the effective transmit power and increases range.

b. It has no impact on the range of the system.

c. It increases the range of the system.

d. It reduces the effective transmit power.

Foundation Topics

802.11 Physical Layer Standards

A wireless LAN enables mobile, portable, and stationary devices to easily communicate with each other within an enterprise facility and throughout a campus environment. For example, retail stores have been using wireless LANs since the early 1990s to enable wireless bar code scanning when performing price marking and inventory applications. Despite the relatively high cost for wireless LAN components at that time, the retail stores were still able to achieve significant returns on investment due to the tremendous gains in efficiency and accuracy that wireless LANs provided.

As wireless LAN prices fell dramatically in 2003, many enterprises began to deploy wireless LANs to support common office applications, such as wireless access to e-mail from conference rooms and the ability to support visiting employees with ease. In addition, companies began installing public wireless LANs at airports, hotels, restaurants, and other hotspots to enable people to have wire-free access to the Internet while away from their offices and homes.

The IEEE 802.11 standard, which is similar in scope and functionality to IEEE 802.3 (Ethernet), is a common basis for wireless LAN operation. As with 802.3, the 802.11 standard defines a common Media Access Control (MAC) and multiple physical layers, such as 802.11a, 802.11b, and 802.11g.

The initial 802.11 wireless LAN standard, ratified in 1997, specifies the use of both direct sequence spread spectrum (DSSS) and frequency hopping spread spectrum (FHSS) for delivering 1- and 2-Mbps data rates in the 2.4-GHz frequency band. DSSS and FHSS are different forms of transmitting data over a wireless LAN. This is plenty of bandwidth to support bar code applications, the first commercial use of wireless LANs. To provide higher data rates when operating in the 2.4-GHz band, the 802.11 group ratified the 802.11b physical layer in 1999, enhancing the initial DSSS physical layer to include additional 5.5- and 11-Mbps data rates. Also in 1999, the 802.11 group ratified the 802.11a standard, which offers data rates up to 54 Mbps in the 5-GHz band using orthogonal frequency division multiplexing (OFDM). 802.11g, ratified in 2004, is the most recent 802.11 physical layer, which further enhances 802.11b to include data rates up to 54 Mbps in the 2.4-GHz band using OFDM. For more details on spread spectrum and OFDM, refer to the following sections found in this chapter: “Spread Spectrum” and “Orthogonal Frequency Division Multiplexing.”

802.11a

Even though the 802.11a standard was available in 1999, 802.11a access points and radio cards did not become commercially available until several years later. The primary reasons for the delay to market were the difficulties in developing 5-GHz 802.11 hardware and the weak market

potential for wireless LAN components that do not interoperate with existing DSSS wireless LANs. 802.11a products are available now, but their use is somewhat limited to specialized applications, especially where high performance is necessary.

A significant advantage of 802.11a is that it offers very high capacity as compared to the other physical layers. The reason is that the 802.11a 5-GHz spectrum defines 12 RF channels that do not overlap in frequency. As a result, it is possible to have up to 12 802.11a access points set to different channels and operating within the same room. This produces up to 12 separate radio cells, each of which can support its own group of wireless users. Most indoor 802.11a access points implement only eight of these RF channels, but that still provides a large potential capacity as compared to 802.11b and 802.11g. If your wireless application needs very high performance, then 802.11a may be the best way to go.

Another advantage of 802.11a is that it operates in the 5-GHz band, which is mostly free from sources of RF interference. Microwave ovens, Bluetooth devices, most cordless phones, and the majority of neighboring wireless LANs operate in the 2.4-GHz band of frequencies. The lower noise floor in the 5-GHz band affords lower retransmission rates and higher resulting throughput as compared to 802.11b and 802.11g systems.

On the other hand, however, 802.11a has limited regulatory acceptance around the world. In fact, some countries do not yet allow 802.11a networks. This could certainly impact the selection of the 802.11 physical layer (a, b, or g) for products that you want to use worldwide.

Also, relatively few user devices currently incorporate 802.11a interfaces. Most new laptops today, for example, implement 802.11b or 802.11g, which do not interoperate with 802.11a access points. 802.11a is not common. If it is not practical to use 802.11a radio cards in user devices, then 802.11a is probably not a good alternative for the wireless LAN.

802.11b

Very soon after ratification of the 802.11b standard, 802.11b access points and radio cards began shipping. It was a fairly easy modification to existing 802.11 DSSS devices to become 802.11b-compliant. In fact, most users could upgrade their existing access points and radio cards with simple firmware upgrades. For several years, 802.11b devices were the best ones on the market, so they proliferated throughout the industry and became the most commonly installed wireless LAN hardware.

An advantage of 802.11b, then, is that it interoperates well with the majority of installed wireless LANs. That is why most wireless user devices include 802.11b. It is not compatible with 802.11a, but 802.11b does interface with 802.11g systems.

Much more RF interference, however, resides in the 2.4-GHz band, which impacts 802.11b users. As mentioned before, a microwave oven can cause significant degradation in throughput because

radio waves from a microwave oven can block 802.11b and 802.11g radio cards from accessing the medium or create bit errors in the 802.11 frames in transit. The potential for RF interference in the 2.4-GHz band is one reason why a company should strongly consider using 802.11a solutions.

A limiting factor of 802.11b is that it supports only up to three nonoverlapping radio cells in the same area. The 2.4-GHz frequency spectrum is roughly 84-MHz wide, and an 802.11b radio card or access point uses approximately 30 MHz when transmitting. To avoid interference among access points, 802.11b access points must be set to specific channels. For example, access points in the United States can be set to channels 1, 6, and 11 to avoid overlap and mutual interference. This is especially important if there are many active wireless users. As a result of this frequency plan and limited data rates, 802.11b has limited capacity.

802.11g

802.11g is backward compatible with 802.11b, which is referred to as *802.11b/g mixed-mode operation*. For example, an 802.11b radio card can associate with an 802.11g access point. Most organizations today are deploying 802.11g wireless LANs. Because of its support for data rates up to 54 Mbps, 802.11g offers higher performance than 802.11b systems. Capacity is still somewhat limited, however, because 802.11g operates in the 2.4-GHz band, which limits the number of nonoverlapping channels to three, as with 802.11b. As a result, 802.11g systems have less capacity than 802.11a wireless LANs. 802.11g, for example, can have up to three nonoverlapping channels with 54 Mbps per channel, whereas 802.11a can have up to 12 nonoverlapping channels with 54 Mbps per channel.

A single 802.11b station associating with an 802.11g access point invokes the use of protection mechanisms, such as request –To Send/Clear To Send (RTS/CTS), which is discussed later in the chapter in the section "RTS/CTS." The use of protection mechanisms is necessary because 802.11b and 802.11g use different modulation, which means that they cannot interoperate and coordinate transmissions according to the 802.11 protocol. The access point informs all stations that an 802.11b station is present by setting an applicable bit in the body of each beacon frame. As a result, all stations begin using protection mechanisms.

The RTS/CTS protection mechanism requires each station to implement the entire RTS/CTS process for each data frame needing transmission. The problem with this requirement is that throughput suffers because of the RTS and CTS frames. Thus, a mixed environment of 802.11b and 802.11g users significantly degrades the throughput of the wireless LAN, often by as much as 30 percent.

This is why most vendors allow administrators to configure access points to allow only 802.11g station associations, referred to as *802.11g-only mode*. Of course, the problem with this is that all users must have 802.11g radio cards. 802.11b-equipped devices will not be able to associate with the access point, but at least the throughput will remain relatively high.

Some vendors also allow you to disable protection mechanisms in mixed mode, which supports both 802.11b and 802.11g connections. This is a good approach if there are a limited number of active users, because the probability of 802.11b and 802.11g devices transmitting at the same time is minimal.

If the need exists to effectively support 802.11b and 802.11g users, especially when there are larger numbers of active users, then you should consider a dual-mode access point. These access points provide separate 802.11b and 802.11g radios in the access points that are set to different, nonoverlapping RF channels. The 802.11g radios in the access points in this case are set to 802.11g-only. This forces the 802.11b users to associate with the 802.11b side of the access point and the 802.11g users to associate with the 802.11g side of the access point. Because each side is to a different channel, there is no need for protection mechanisms.

802.11n

The 802.11 working group is actively developing a new physical layer known as 802.11n. This standard will be the basis of the next generation of wireless LANs, with data rates well above 100 Mbps and much better throughput. At this point, no decision has been made regarding which proposal for this standard will move forward as the draft standard, but it will certainly include multiple input-multiple output (MIMO) antenna technology. MIMO enables multiple antennas to create simultaneous RF channels that increase the performance of wireless LAN, similar to serial-to-parallel conversion techniques used with wired networks. The 802.11 working group is expected to ratify this standard in 2006 or 2007.

Comparison of 802.11 Standards

Table 22-2 provides a comparison of the different characteristics of the 802.11a, 802.11b, and 802.11g standards.

Table 22-2 *802.11 Standards Comparison*

KEY POINT

Standard	RF Spectrum	Max Speed	Compatibility	RF Interference Impacts	Date Ratified
802.11a	5 GHz	54 Mbps	Does not work with 802.11b or 802.11g	Slight	1999
802.11b	2.4 GHz	11 Mbps	Works with 802.11g	Moderate	1999
802.11g	2.4 GHz	54 Mbps	Works with 802.11b	Moderate	2004

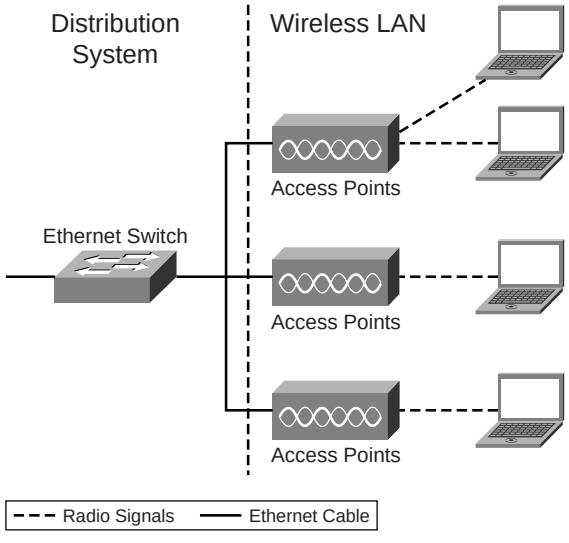
Wireless System Configuration

When deploying a wireless LAN, determine which configuration makes most sense to install. The 802.11 standard refers to two different wireless system configurations, infrastructure mode and ad hoc mode, which vary in terms of usage and operation. This section explores each configuration.

Infrastructure Mode Configuration

An *infrastructure wireless LAN* (sometimes referred to as *infrastructure mode*) is what most companies, public hotspots, and home users implement. An infrastructure wireless LAN, as depicted in Figure 22-1, offers a means to extend a wired network. In this configuration, one or more access points interface wireless mobile devices to the distribution system. Each access point forms a radio cell, also called a *basic service set (BSS)*, which enables wireless users located within the cell to have connectivity to the access point. This allows users to communicate with other wireless users and with servers and network applications connecting to the distribution system. A company, for example, might use this configuration to enable employees to wirelessly access corporate applications and the Internet from anywhere within the facility.

Figure 22-1 *Infrastructure Wireless LAN*



Each access point in the infrastructure wireless LAN creates a radio cell, with a coverage area that depends on the following variables:

KEY POINT

- The construction of the facility
- The physical layer
- Transmit power
- Antenna type

The range of the coverage area is typically 150 feet in most enterprise facilities. The desired level of performance, however, can impact the effective range of the access points. Chapter 23, "Wireless LAN Solutions," covers this in more detail.

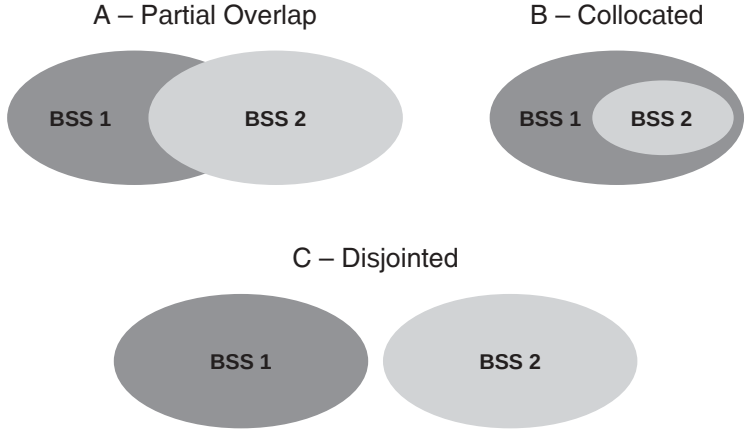
KEY POINT

With the infrastructure configuration, data traffic going from one wireless user to another user must travel through the access point. The access point switches the data traffic going from user A to user B,

for example, and retransmits the data to user B. In infrastructure mode, data transmissions do not occur directly between the wireless users. As a result, significant data traffic between wireless users decreases throughput because the access point must relay the data to the destination user. In this application, you can think of the access point as a Layer 2 switch. It provides access to a common medium that two hosts can use to communicate with each other, but there is no direct connection between the hosts. If the source wireless user is sending data to a node on the distribution system, then the access point does not need to retransmit the data to other wireless users.

Figure 22-2 illustrates three different 802.11 radio cell configurations, which include partial overlap, collocated, and disjointed cells. If the company installs access points with overlapping radio cells, as shown in part A of Figure 22-2, then users are able to roam throughout the facility. The radio card within the user's mobile device will automatically reassociate with access points having stronger signals. For example, a user might begin downloading a file when associated with access point A. As the user walks out of range of access point A and within range of access point B, the wireless LAN automatically reassociates the user to access point B and the user's file download continues through access point B. The user generally does not experience any noticeable delay, but voice-over-WLAN phones might drop connections if the roaming delay exceeds 100 milliseconds.

Figure 22-2 *Various 802.11 Radio Cell Configurations*



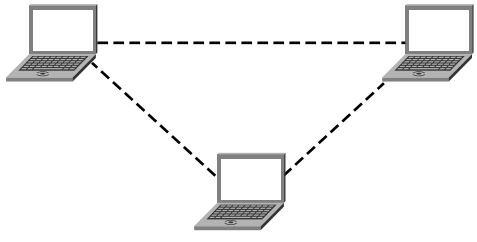
The 802.11 standard also supports collocated (part B of Figure 22-2) and disjointed (part C) radio cells. A company might install disjointed access points if complete coverage throughout the facility is not necessary. For example, the company might install an access point in each conference room but not in the rest of the building. If the radio cells are disjointed, then users will temporarily lose the network connection and then reassociate when coming within range of another access point. An 802.11 network, though, supports this form of network in a manner similar to a wireless network that supports roaming with the overlapping radio cells. The reassociation delay is a function of the time it takes the user to move into range of the next access point. The wireless application in use, however, might or might not be able to tolerate this longer roaming delay.

The co-located radio cell configuration is useful if a company needs greater capacity than what a single access point can deliver. In this scenario, two or more access points are set up so that their radio cells overlap significantly. This works well, assuming that the access points are set to nonconflicting radio channels. A portion of the users in the area, for example, associate with access point A, and the other users associate with access point B. This boosts the wireless LAN capacity in that particular area. For example, with 802.11g, you could install up to three co-located cells, each using a non-overlapping channel, to have three times 54 Mbps of available bandwidth.

Ad Hoc Mode Configuration

Instead of forming an infrastructure wireless LAN, the 802.11 standard allows users to optionally connect directly to each other in what is referred to as ad hoc mode, illustrated in Figure 22-3. The rationale behind this form of networking is to enable users to spontaneously set up wireless LANs. This optional mode is available to users on most radio cards. With ad hoc mode, there is no need for access points. The wireless connection is made directly between the users in a peer-to-peer fashion.

Figure 22-3 *Ad Hoc Wireless LAN*



Ad hoc mode is beneficial when a user needs to send a file to another user within the same room, and no other networking is practical. Both users can enable ad hoc mode on their radio cards, and the users can communicate wirelessly. The use of an ad hoc wireless LAN is also valuable when users need to rapidly establish a wireless network when responding to emergencies in areas where no network is in place.

Wireless Hardware Components

The primary components of a wireless LAN include radio cards and access points, which operate at Layers 1 and 2 of the network architecture. Some wireless LAN components, however, include higher-layer functionality. A wireless LAN router, for instance, includes network layer functions, such as DHCP and NAT. The sections that follow take a closer look at each of the following components that comprise a wireless LAN:

- Radio cards
- Access points
- Antennas

- Repeaters
- Bridges
- Routers
- Radio frequency peripherals

Radio Cards

The radio card implements the 802.11 MAC functions and a specific physical layer, such as 802.11a, 802.11b, or 802.11g. The firmware on the card implements the MAC functionality, and a transceiver provides the physical layer transmitting and receiving tasks. The 802.11 standard refers to a radio card as an *802.11 station*.

The availability of multimode radio cards makes it possible for users to associate with 802.11b/g and 802.11a wireless LANs. The majority of installed wireless LANs are 802.11b, but the growing number of 802.11a wireless LANs offer significant performance advantages over 802.11b/g networks. With the tri-mode capability, a user is fully equipped with all available wireless LAN interfaces, which maximizes interoperability.

Access Points

Similar to a radio card, an access point implements the common MAC functions and specific physical layers. In addition, the access point contains an Ethernet station and access point system functions. With Cisco enterprise access points, the system functions include enhanced management, improved security controls, and performance upgrades. The traditional access points are relatively intelligent and implement enough functions to allow the access point to interconnect with other access points via conventional Ethernet switches.

Some access points, such as the Cisco 1200, contain multiple radio card slots. This enables a company to deploy collocated wireless LANs. For example, the access point can include both 802.11a and 802.11b, which can serve different applications; 802.11b could be dedicated to supporting data traffic, whereas 802.11a could satisfy wireless connectivity for voice-over-WLAN phones.

As an alternative to the traditional intelligent access point, some companies offer “thin” access points that implement the basic 802.11 functions. These thin access points connect to an intelligent wireless switch, which provides enhancements for management, security, and performance. With the acquisition of Airespace, Cisco offers a thin access point solution.

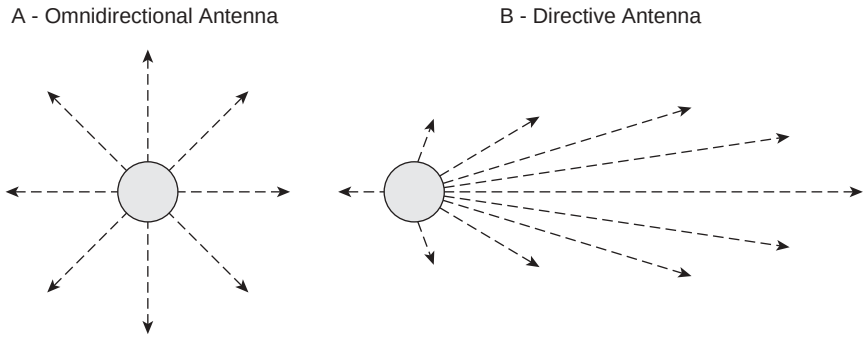
Antennas

The antenna couples radio waves between the radio card’s transceiver and the air medium. The transceiver converts digital data from the computer to a radio frequency (RF) signal (and vice versa). The antenna transmits and receives RF signals that convey information between user

devices. Omnidirectional antennas, which come with most access points, form a circular coverage pattern around the antenna, as part A of Figure 22-4 illustrates. In most cases, a company will install multiple access points within a facility, with omnidirectional antennas creating individual radio cells that overlap with each other to provide roaming. This generally offers optimum coverage over a large indoor area with a minimum number of access points.

Figure 22-4 *Wireless LAN Antenna Types and RF Signal Patterns*

KEY POINT



As an alternative to omnidirectional antennas, directive antennas focus RF energy more in one direction than others. Part B of Figure 22-4 illustrates the coverage pattern of a directional antenna. These types of antennas are ideal for providing coverage over a larger area for which it is not practical to install access points. For example, a utility company can install a directional antenna to cover an outdoor area that is storing power transformers. The directional antenna transmits and receives radio waves over longer distances within the coverage pattern, much as a flashlight focuses light in one direction.

Some antennas are removable and others are integrated, so it might not be possible to change an access point's antenna type. When purchasing access points, ensure that the antennas are removable regardless of whether there are current plans to use different antennas. It might be necessary in the future to use the access points with directional antennas.

Repeaters

A wireless LAN repeater simply receives 802.11 data frames and retransmits them on the same channel. They implement the entire 802.11 standard, similar to radio cards. This enables a company to extend coverage of the wireless LAN to areas where it is not practical to install Ethernet data cabling. When deploying a repeater, the RF channel is set to the same channel to the access point to which it associates. An issue that comes with repeater use is that a repeater doubles the number of data frames sent over the air because of the retransmission process. The result is a significant bandwidth reduction. This might not be a problem, however, if the wireless application does not need the entire capacity of the wireless LAN.

NOTE For information about RF channels, see the section "RF Channels," later in the chapter.

Bridges

Bridges provide a direct wireless connection between two wireless LANs or between a wired network and a wireless LAN. With remote wireless bridges, the system provides a point–to-point link between two networks. This is useful for linking two buildings separated by as much as a few miles. Highly directive antennas provide relatively long range, up to 20–30 miles.

Routers

Wireless LAN routers are available for deploying wireless LANs in small offices and homes. These devices, such as the Linksys WRT54GS, include network layer functions in addition to the access point's wireless functionality. An integrated Dynamic Host Configuration Protocol (DHCP) server assigns IP addresses to wireless LAN user devices, and Network Address Translation (NAT) is available to bind the local private addresses to a single public IP address provided by an Internet service provider (ISP). The wireless LAN router simply connects to a broadband modem, such as digital subscriber line (DSL) or cable modem, and the default settings offer immediate wireless LAN connections to the Internet to wireless users.

Most larger enterprises install access points and not routers. The DHCP and NAT services are provided from a central server to all users within the company. The Cisco 350, for example, allows a company to switch the access point into a router. This might be necessary if a large enterprise needs to outfit a small remote office with wireless LAN access to an ISP.

Radio Frequency Peripherals

In addition to changing antennas to improve wireless LAN range and performance, wireless LANs can use various RF peripherals. For example, an RF splitter routes a single RF source down two separate paths. This is useful when an access point must cover two separate areas. Bear in mind that each splitter inserts significant attenuation, which reduces the effective range of the antenna; however, using a splitter might make sense if you must cover two areas with a single access point.

Wireless LAN amplifiers are available to increase the signal power feeding the antenna. This improves the range within the beamwidth of the antenna. Instead of replacing the antenna with one having higher gain (which decreases the beamwidth), an amplifier pushes the radio signals farther over the entire beamwidth. Amplifiers are available up to 6 watts. In the United States, FCC licensing is necessary for amplifiers, as well as when you are using RF components that are different from what is certified with the FCC by the access point and radio card vendor. Canadian regulations for wireless LAN frequency allocations are nearly identical to U.S. regulations, but they are governed by Industry Canada. Other countries might have differing regulations, so consult the appropriate rules before installing accessories or changing antennas on your wireless LAN.

Infrastructure Mode Operation

Infrastructure mode operation involves the use of access points that connect to a distribution system. Because this is the most common wireless LAN configuration in use today, it is important that you have a thorough understanding of its operation. The sections that follow cover various aspects of infrastructure mode operation, including:

- Scanning
- Connecting with a network
- Data transfer
- Roaming

Scanning

In an infrastructure wireless LAN, each radio card implements a scanning function to find access points. Scanning occurs after booting the user device, and periodically afterward to support roaming. The 802.11 standard defines two scanning methods:

- Passive scanning
- Active scanning

Each radio card vendor, however, implements them differently.

Passive Scanning

The following list describes the process that occurs when a radio card uses passive scanning:

KEY POINT

1. The radio card automatically tunes to each RF channel, listens for a period of time, and records information it finds regarding access points on each channel.
2. By default, each access point transmits a beacon frame every 100 milliseconds on a specific RF channel, which the administrator configures.
3. While tuned to a specific channel, the radio card receives these beacon frames if an access point is in range and transmitting on that channel.
4. The radio card records the signal strength of the beacon frame and continues to scan other channels.
5. After scanning each of the RF channels, the radio card makes a decision about the access point with which it will associate.

In general, the radio card chooses the access point with the strongest beacon signal. Some radio vendors might include other parameters, such as noise levels and utilization, when making this decision.

Active Scanning

The following list describes the process that occurs when a radio card uses active scanning:

KEY POINT

1. The radio card sends probe request frames on each RF channel.
2. If able to do so, any Access Point receiving the probe request sends a probe response.
3. The radio card uses the signal strength and possibly other information corresponding to the probe response frame to make a decision as to the access point to which it will associate.

The probe response is similar to a beacon frame. Active scanning, however, enables the radio card to receive information about nearby access points in a timely manner, without waiting for beacons. Again, each vendor implements passive and active scanning differently.

Connecting with a Network

After obtaining a list of potential access points via either passive or active scanning, the radio card moves forward with joining the network by tuning to the RF channel of the chosen access point. To initiate association, the radio card sends an authentication request frame, and the access point responds with an authentication response frame. This is the default authentication that 802.11 refers to as *open system authentication*. In most cases, this form of authentication is desirable. 802.11 offers the optional shared key authentication, which uses a *Wired Equivalent Privacy (WEP)* key to authenticate the radio card, but WEP is relatively easy to hack. As a result, companies should use stronger forms of authentication, such as 802.1x, that rely on authentication servers.

After performing the authentication handshake, the radio card sends an association request frame to the access point. This request contains information about the radio card, including the *service set identifier (SSID)* and the radio card's supported data rates. The SSID must match the one configured in the access point for association to complete. The access point replies to the radio card with an association response frame containing an *association identifier (AID)*, which is a number that represents the radio card's association. At this point, the radio card is considered associated with the access point, and the radio card can then begin sending data frames to the access point and communicating with other nodes on the network.

Data Transfer

The exchange of data in an 802.11 network is bidirectional between the radio card and access point. As mentioned earlier, data frames in an infrastructure wireless LAN do not travel directly between wireless users. Instead, the access point relays the data.

KEY POINT

A radio card or access point (802.11 station) having the destination MAC address of the data frame replies with an acknowledgement (ACK) frame. This adds significant overhead to a wireless LAN as compared to an Ethernet network that does not require ACKs for every data frame. The ACKs are necessary with 802.11 due to the nature of the shared radio medium; data loss is much more likely with the wireless medium, so wireless LANs perform error detection and error correction at Layer 2.

If an 802.11 station sending a data frame does not receive an ACK after a specific period of time, the station retransmits the frame. These retransmissions occur up to a particular limit, which is generally three to seven times. After that, higher-layer protocols, such as Transmission Control Protocol (TCP), must provide error recovery.

To allow for extended range, 802.11 includes automatic data rate shifting. For example, an 802.11 station generally lowers its transmission data rate if a retransmission is necessary. Access points support multiple data rates to facilitate this kind of operation, where different remote stations might transmit data upstream at different rates.

Roaming

Periodically, each radio card performs scanning, either active or passive, to update its access point list. Some radio cards might limit the scanning function first to only RF channels where access points formerly had been found. This enables the radio card to offer higher throughput, because the card cannot send or receive data frames while scanning other channels.

If the associated access point signal becomes too weak, then the radio card will implement a reassociation process. The radio card sends a reassociation frame to the new access point and a disassociation frame to the old access point. 802.11 does not require the authentication frame handshake when reassociating. If the old access point has buffered data frames destined to the radio card, then the old access point will forward them to the new access point for delivery to the radio card.

Ad Hoc Mode Operation

With ad hoc wireless LANs, there are no access points; therefore, the radio cards must send beacons. The ad hoc mode of operation transpires as follows:

KEY POINT

1. After a user switches to ad hoc mode, the radio card begins sending beacons if one is not received within a specific period of time.
2. After receiving a beacon, each radio card waits a random period of time.
3. If a beacon is not heard from another station in this time, then the station sends a beacon. The random wait period causes one of the stations to send a beacon before any other station. Over time, this distributes the job of sending beacons evenly across all 802.11 stations.

The sharing of the transmission of beacons among all ad hoc stations is necessary to ensure that beacons are still sent if a particular station becomes unavailable. If a station becomes disassociated with the network, then another station will send the beacon.

With ad hoc networks, there is no direct connection to a wired network, which, of course, limits applications. A user, however, can configure an 802.11-equipped device as an ad hoc station, such as a PC, to provide a shared connection to a wired network. Thus, with specialized software or

functions within the PC operating system, the PC can offer functions similar to those of an access point. All of the other ad hoc stations needing to reach devices on the wired network funnel their packets through the PC's connection to the network.

Wireless Configuration Parameters

The 802.11 standard specifies several configuration parameters, which are set in the radio card or the access point (or both). Figures 25-5 and 25-6 show wireless configuration screens of an access point. To optimize performance and security of a wireless LAN, you must choose configuration settings that best satisfy application requirements. Most of these configurations apply to the access points, but some settings are necessary on the radio cards within the end-user devices.

Figure 22-5 *Basic Setup Screen of a Linksys WRT55AG Dual-Band Access Point*

Figure 22-6 *Advanced Setup Screen of a Linksys WRT55AG Dual-Band Access Point*

SSID

KEY POINT

The service set identifier (SSID) is an alphanumeric value set in access points and radio cards to distinguish one wireless LAN from another. For example, the default SSID for Cisco access points is "tsunami." The SSID provides a name for the wireless LAN. The beacon frame includes the SSID. Microsoft Windows extracts the SSID from the radio card, which obtains SSIDs from the beacon frames. Windows displays a list of available wireless networks (by SSID) to the user. If the user chooses to connect to one of the wireless LANs, Windows initiates the association process.

NOTE Keep in mind that not all radio cards support the feature that Microsoft Windows offers. When using mobile device operating systems or radio cards that do not extract the SSID from beacons, then the user must manually configure the radio card with the SSID of the access points that they wish to associate with.

When installing a wireless LAN with multiple access points, such as in the case of an enterprise network, set all access points to the same SSID. This minimizes issues with users having incorrect SSIDs and ensures that roaming between access points works as smoothly as possible and without user intervention.

Some access points, such as the Cisco 1200, allow administrators to set the access point to disable SSID broadcasting. When SSID broadcasting is disabled, the access point does not include the SSID in beacon frames. As a result, Windows is not able to obtain the SSID and display it as a candidate network for association. This is beneficial in corporate networks to keep casual snoopers and wardrivers from finding the network. For example, a user with a Windows-based laptop would not see the corporate network as an association option.

With SSID broadcasting disabled, however, the user must manually configure the radio card with the correct SSID in order to associate with the access point. This is acceptable in corporate and home environments, but it is not desirable with public wireless hotspots. In the latter case, public hotspots can use the SSID for advertising. For example, T-Mobile uses the SSID "T-Mobile" to inform users (through Windows) that its network is available and to distinguish its service from competitors. Disabling SSID broadcasting in public networks would prevent users from determining whether wireless LAN coverage for a particular service provider is available.

Disabling SSID broadcasting is not a strong security mechanism. A hacker, for example, can easily monitor 802.11 frames on the wireless LAN and wait until an association frame is sent by a radio card as a mandatory part of the association process. The hacker will find the SSID in association request frames and some probe requests. In fact, commercial wireless LAN analyzers look inside association frames and automatically display SSIDs once they are found.

The 802.11 standard specifies the need for each access point to have an SSID, but several vendors, including Cisco, allow multiple SSID assignments for each access point. Administrators can map

each SSID to a different virtual LAN (VLAN), which makes it possible to support diverse applications on a common wireless LAN infrastructure. This creates multiple virtual wireless LANs operating over the same physical layer.

For example, an airport might want to share the same wireless LAN infrastructure to support both airport check-in applications and wireless service providers offering public Internet access to passengers. The airport can disable SSID broadcasting for the private applications and map this SSID to a VLAN that is accessible only through strong security mechanisms. As a result, passengers would not see the airport's SSID through Windows. Each access point can be set to broadcast the SSIDs of the public service providers so that users can find the service offerings. Each SSID would interface to a VLAN that connects only with the applicable service provider.

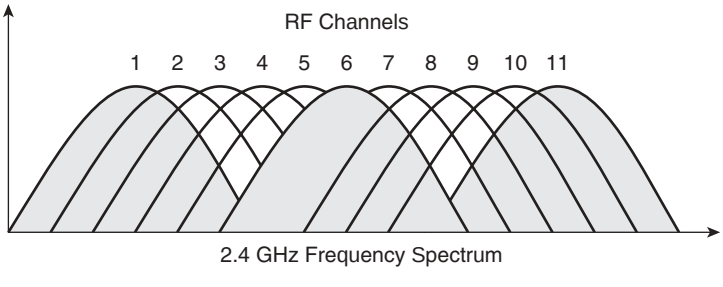
The first radio card in the area that is set to ad hoc mode includes its SSID in the beacon. This establishes an ad hoc wireless LAN that other users (set to ad hoc mode) can join. As with an infrastructure network, each radio card must be set to the same SSID to associate in an ad hoc network. As with infrastructure wireless LANs, Windows displays a list of SSIDs corresponding to ad hoc networks. With some radio cards and when using non-Windows operating systems, it might be necessary for each ad hoc user to manually configure their radio card with the desired SSID. SSID disabling is generally not available to ad hoc users.

RF Channels

Each 802.11 physical layer defines a set of RF channels. For example, the 802.11b/g standard defines 14 RF channels in the 2.4-GHz band, with varying numbers of these channels available in specific countries. In the United States, for instance, the FCC's rules allow the use of only channels 1 through 11. In the case of 802.11b/g, these channels overlap with each other, as shown in Figure 22-7. As a result, companies installing 802.11b/g wireless LANs should set adjacent access points (where their radio cells overlap) to nonconflicting channels. These are channels that do not overlap with each other, such as channels 1, 6, and 11 in the U.S. This minimizes interference among access points, which can significantly reduce throughput when user traffic is high. Other 802.11 standards, such as 802.11a, define separate RF channels that do not overlap.

Figure 22-7 *Overlapping 802.11 RF Channels in the 2.4-GHz Band*

KEY POINT



It is important to define the appropriate RF channels by first conducting an RF site survey, as explained in Chapter 23. The survey provides information necessary to identify the most optimum channels for avoiding interference sources, such as microwave ovens and neighboring wireless LANs. In infrastructure wireless LANs, the RF channel is set in the access point only. Cisco access points include automatic channel selection as an option. When this feature is enabled, the access point automatically listens to each RF channel and makes a decision about which channel to use to avoid interference. As conditions change, the access point responds accordingly. This significantly reduces the time necessary to initially determine optimum channel settings and make applicable updates when supporting the network.

In ad hoc networks, the first radio card active on the network is set by the user to a specific channel. After joining a particular ad hoc wireless LAN, each radio card belonging to that particular ad hoc network stays on the same channel. The channel in use by the ad hoc network is the one that the first ad hoc user chose.

Transmit Power

Most access points and radio cards allow the setting of transmit power. The highest value is generally 100 mW (0.1 W), with increments of lower power available. Some devices enable settings as low as 1 mW. In most cases, it is best to set all wireless LAN devices to the highest transmit power, which is generally the default setting.

To configure a wireless LAN for optimum capacity, you can set the transmit power to a lower value, which effectively reduces the size of the radio cells surrounding each access point and radio card. More access points are necessary to cover an entire facility, as compared to using higher transmit power levels, but fewer wireless users will then associate with each access point. The result is better performance due to fewer users competing for access to the medium. The use of lower power settings and a greater number of access points is beneficial for supporting voice-over-Wi-Fi applications, assuming that roaming delays between the access points is kept to a minimum by careful system design.

Data Rates

The default data rate setting on access points is generally **auto**, which allows radio cards to use any of the data rates of the given physical layer. For example, 802.11b allows data rates of 1, 2, 5.5, and 11 Mbps. The 802.11g standard extends these data rates up to 54 Mbps. The radio card usually attempts to send data frames at the highest supported rate, such as 11 Mbps for 802.11b stations and 54 Mbps for 802.11g stations. When set to **auto**, the radio card automatically rate shifts to the highest data rate that the connection can support. A lower data rate, for example, might be necessary if the radio card encounters too many retransmissions.

It is possible to set the access point to a specific data rate, such as 1 Mbps, which forces the access point to send all frames at 1 Mbps. This effectively increases the range of the access point, because

802.11 stations can detect access points over longer ranges. In general, a radio card is able to communicate successfully with lower data rates over longer ranges.

The access point data rate setting, though, does not affect the data rate of the radio cards. If the radio card is set to **auto** data rates (the default setting), then the radio card can still use the highest possible data rate when sending frames to the access point. To maximize the range with fewer retransmissions, set the radio cards to lower, fixed data rates. These data rate settings impact only the transmit data rate. The radio card will still receive frames at higher data rates if necessary.

Power-Save Mode

Most radio cards employ an optional 802.11 power-save mode that users can enable. Access points do not implement power-save mode, except for the buffering functions necessary to support power-saving functions of the radio cards. If power-save mode is enabled, the radio card enters sleep mode, which draws much less current than when the card is operating actively. Thus, power-save mode can conserve batteries on mobile devices. In fact, power-save mode often lengthens battery life by 20 to 30 percent. The actual savings, however, depends on the applications and other variables.

Before switching to power-save mode, the radio card notifies the access point by setting the Power Management bit in the Frame Control field of an upstream frame. The access point receives this frame and starts buffering applicable data frames. The buffering takes place until the radio card awakens and requests that the access point send the saved frames to the radio card.

After entering sleep mode, the radio card keeps track of time and wakes up periodically to receive each beacon coming from the access point. The radio card must wake to discover whether the access point is buffering any frames that need delivery to the radio card. The access point notifies radio cards about buffered packets through what the 802.11 standard defines as the *traffic indication map (TIM)*.

A radio card set to power-save mode will wake just in time to receive the TIM, which resides in the beacon frames. The TIM indicates the AID of the 802.11 stations that have data frames buffered at the access point. If a station discovers it has frames at the access point, then the station stays awake and sends a power-save poll frame to the access point requesting that the data frames be forwarded to the station. The station will stay awake long enough to receive all of the buffered frames. The amount of time required to transfer all of the buffered frames depends on the current utilization of the access point and the radio link quality. A large number of stations implementing power-save mode can cause a surge in traffic after each beacon due to power-save poll frames and corresponding data frames.

Once the access point delivers the buffered frames, the radio card enters sleep mode again, unless the beacon frame corresponds with the delivery traffic indication map (DTIM). The DTIM is set in the access point to determine how many beacons must pass before the access point delivers multicast frames. A common default DTIM interval is 3, which means that the access point sends

multicast frames after every third beacon. The DTIM interval, however, can be set to other values, such as 1, which enables the access point to send multicast frames after every beacon. Based on the DTIM interval setting, a station implementing power-save mode will stay awake long enough after the beacon transmission to receive the multicast frames in addition to unicast frames.

The use of power-save mode can make batteries last longer in user devices, but throughput decreases for data moving from the access point to the user device. The radio card will awaken immediately and send data going from the user device to the access point, however. As a result, upstream throughput remains unchanged in low-power mode.

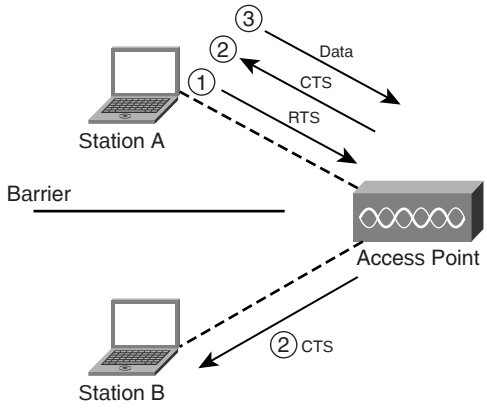
RTS/CTS

The 802.11 standard defines request-to-send/clear-to-send (RTS/CTS) as an optional function of 802.11 to regulate the transmission of data on the wireless LAN. RTS/CTS can be set in the access point or a radio card individually, or on both devices at the same time. In most cases, the RTS/CTS function is helpful in counteracting collisions between hidden nodes. To gain access to the shared wireless medium, a station can only transmit if no other station is transmitting. Within a particular access point radio cell, it is possible that two stations associated with the same access point might be out of range of each other.

For example, Figure 22-8 illustrates the scenario in which either Station A is far away from Station B or a barrier is blocking the radio signals between the two stations. The problem is that Station A might be in the middle of transmitting a frame to the access point when Station B wants to send a frame. Station B will listen to the medium to determine whether another station is already transmitting. Because Station B cannot hear Station A, Station B starts transmitting the frame. A collision then occurs at the access point, which destroys both frames. As a result, the access point will not acknowledge reception of the frames. Both stations will have to retransmit their respective frames, which will likely result in another collision.

Figure 22-8 *Hidden-Node Problem in Wireless LANs*

KEY POINT



The RTS/CTS function is a handshaking process that minimizes the occurrence of collisions when hidden nodes are operating on the network. In addition, protection mechanisms can use RTS/CTS to avoid collisions between 802.11b and 802.11g radio cards. If hidden nodes are not causing significant retransmissions or hidden nodes are not present, then RTS/CTS is generally not necessary (unless protection mechanisms are in use).

KEY POINT

RTS/CTS works by enabling each station to explicitly request a time slot for data transmission. For example, if RTS/CTS is enabled in the radio cards of Station A and Station B, then Station A will first send an RTS frame to the access point before attempting to transmit a data frame. The access point receives the RTS frame and responds with a CTS frame. Both stations receive the CTS frame. This gives clearance for Station A to transmit a data frame. The CTS frame carries a duration value that informs all other stations, including Station B, to not transmit during the specified time interval. This delay equals the amount of time that Station A requires to send the data frame and for the access point to respond with an acknowledgement. As a result, collisions due to hidden nodes are much less likely. The only time it is possible with RTS/CTS enabled is when hidden nodes also miss CTS frames from the access point because of interference or weak signals.

Enabling RTS/CTS requires setting RTS/CTS to a specific threshold value in the access point or radio card. The threshold is the frame length that invokes the RTS/CTS process. If the data frame that needs transmission is larger than the threshold, then the station implementing RTS/CTS uses the RTS/CTS process.

If you suspect hidden nodes might be causing collisions on the network and degrading network performance, consider activating RTS/CTS. It is best to first set the threshold to approximately 750 bytes, which is roughly the halfway point (50 percent of the maximum frame size), and assess changes in throughput. If throughput increases, then the use of RTS/CTS is beneficial. If throughput decreases, then RTS/CTS likely is not worthwhile. If RTS/CTS is in fact improving performance, consider tweaking the threshold value to maximize throughput.

Keep in mind that the use of RTS/CTS adds overhead to the network because the transmission of data frames might require RTS and CTS frames as well. As a result, the lessening of collisions by using RTS/CTS might not offer enough improvement in throughput to compensate for the additional overhead of RTS/CTS frames. The ultimate goal of using RTS/CTS is to improve throughput. If RTS/CTS causes throughput to decrease due to the additional overhead, then do not use it. Even if the use of RTS/CTS improves performance, changes within the facility and users that are roaming to other areas will change the hidden-node situations. Thus, RTS/CTS might not be necessary, and can even cause degradation in throughput as conditions change. 802.11 does not offer adaptive mechanisms that automatically adjust RTS/CTS thresholds.

Fragmentation

A radio card or access point can be set to optionally use fragmentation, which divides 802.11 data frames into smaller pieces (fragments) that are sent separately to the destination. Each fragment consists of a MAC layer header, frame check sequence (FCS), and a fragment number indicating its ordered position within the frame. Because the source station transmits each fragment independently, the receiving station replies with a separate acknowledgement for each fragment.

An 802.11 station applies fragmentation only to frames having a unicast destination address. This includes any data frame directed toward a specific station. To minimize overhead on the network, 802.11 does not fragment broadcast and multicast frames.

The destination station reassembles the fragments into the original frame using fragment numbers. After ensuring that the frame is complete, the station hands the frame up to higher layers for processing. Even though fragmentation involves more overhead, its use can result in better performance if you tune it properly.

Fragmentation can increase the reliability of frame transmissions when significant RF interference is present. When transmitting smaller frames, collisions are less likely to occur. In addition, frames that do encounter errors can be retransmitted faster because they are smaller. The fragment size value can typically be set between 256 and 2048 bytes, although this value is user-configurable. In fact, you activate fragmentation by setting a particular frame size threshold (in bytes). If the frame that the access point is transmitting is larger than the threshold (similar to RTS/CTS), it will trigger fragmentation. If the packet size is equal to or less than the threshold, the access point will not use fragmentation.

Consider enabling fragmentation on radio cards and access points. As with configuring RTS/CTS, first set the fragmentation threshold to 750 bytes. If throughput increases, then fragmentation is beneficial. If throughput decreases, then fragmentation probably won't be worthwhile. Because of the additional overhead required for frame headers on each fragment, the reduction in frame retransmissions might not be enough to counteract the additional overhead necessary.

RTS/CTS and Fragmentation Summary

Table 22-3 summarizes some of the benefits and drawbacks of both RTS/CTS and fragmentation when trying to improve throughput in a wireless network.

Table 22-3 *Improving Throughput Using RTS/CTS and Fragmentation*

KEY POINT

	RTS/CTS	Fragmentation
Method to Improve Throughput	Reduce collisions	Reduce percentage of frames with transmission errors
How to Enable	Configuration on AP	Configuration on AP

Table 22-3 *Improving Throughput Using RTS/CTS and Fragmentation (Continued)*

	RTS/CTS	Fragmentation
Subset of Data Frames	Frames under a statically defined length	Frames under a statically defined length
Overhead Created	RTS/CTS process frames	Addition of fragmentation headers

Wireless Medium Access

Before transmitting frames, a station must first gain access to the medium, which is a radio channel that stations share. The 802.11 standard defines two forms of medium access:

- Distributed coordination function (DCF)
- Point coordination function (PCF)

DCF is mandatory and based on the carrier sense multiple access with collision avoidance (CSMA/CA) protocol. With DCF, 802.11 stations contend for access and attempt to send frames when there is no other station transmitting. If another station is sending a frame, stations are polite and wait until the channel is free.

The following are details on how DCF works:

KEY POINT

1. As a condition of accessing the medium, the MAC layer checks the value of its network allocation vector (NAV), which is a counter resident at each station that represents the amount of time that the previous frame needs to send its frame. The NAV must be zero before a station can attempt to send a frame. Prior to transmitting a frame, a station calculates the amount of time necessary to send the frame based on the frame's length and data rate. The station places a value representing this time in the Duration field in the header of the frame. When stations receive the frame, they examine this Duration field value and use it as the basis for setting their corresponding NAVs. This process reserves the medium for the sending station.
2. An important aspect of the DCF is a random Back-off timer that a station uses if it detects a busy medium. If the channel is in use, the station must wait a random period of time before attempting to access the medium again. This ensures that multiple stations that want to send data do not transmit at the same time. The random delay causes stations to wait different periods of time, which avoids the situation in which all the stations sense the medium at exactly the same time, find the channel idle, transmit, and collide with each other. The Back-off timer significantly reduces the number of collisions and corresponding retransmissions, especially when the number of active users increases.
3. With radio-based LANs, a transmitting station cannot listen for collisions while sending data, mainly because the station cannot have its receiver on while transmitting the frame. As a result, the receiving station needs to send an acknowledgement (ACK) if it detects no errors

in the received frame. If the sending station does not receive an ACK after a specified period of time, it assumes that there was a collision (or RF interference) and retransmits the frame.

To support time-bounded delivery of data frames, the 802.11 standard defines the optional point coordination function (PCF), which enables the access point to grant access to an individual station to the medium by polling the station during the contention-free period. Stations cannot transmit frames unless the access point polls them first. The period of time for PCF-based data traffic (if enabled) occurs alternately between contention (distributed coordination function [DCF]) periods.

The access point polls stations according to a polling list, and then switches to a contention period when stations use DCF. This process enables support for both synchronous (for example, video applications) and asynchronous (for example, e-mail and web-browsing applications) modes of operation. No known wireless NICs or access points on the market today, however, implement PCF.

Without effective quality of service (QoS), the existing version of the 802.11 standard does not optimize the transmission of voice and video. Currently, no effective mechanism exists to prioritize traffic within 802.11. As a result, the 802.11e task group is currently refining the 802.11 MAC layer to improve QoS for better support of audio and video (such as MPEG-2) applications. The 802.11e group should finalize the standard by 2006.

Because 802.11e falls within the MAC layer, it will be common to all 802.11 physical layers and will be backward compatible with existing 802.11 wireless LANs. As a result, the lack of 802.11e in place today does not impact your decision on which physical layer to use. In addition, you should be able to upgrade your existing 802.11 access points to comply with 802.11e through relatively simple firmware upgrades once they are available. In the meantime, some wireless LAN vendors offer proprietary QoS mechanisms that improve performance when there is a mix of applications with differing QoS needs present on the network, such as voice and data.

Wireless Security

Security is one of the most important features of a wireless LAN. Companies and home users must protect their information from hackers. The 802.11 standard offers security mechanisms that have undergone significant scrutiny over the past few years. Recently, improvements have been made by the Wi-Fi Alliance and via newer versions of the 802.11 standard.

The most common wireless security mechanisms currently in use today are as follows:

KEY POINT

- Wired Equivalent Privacy (WEP)
- Temporal Key Integrity Protocol (TKIP)
- Advanced Encryption Standard (AES)

- Wi-Fi Protected Access (WPA)
- Open system authentication
- Shared key authentication
- Virtual Private Networks (VPNs)

The sections that follow describe each of these mechanisms in more detail.

WEP

802.11 Wired Equivalent Privacy uses a common key to encrypt and decrypt data frame contents between 802.11 stations at Layer 2. 802.11, however, does not define a mechanism for distributing WEP keys to the stations. This requires the administrator or users to manually configure their radio cards with the encryption key, and it is not practical to change the key. Thus, WEP keys remain the same on most wireless LANs for months or years. This allows enough time for a hacker to exploit the vulnerabilities of WEP and crack the encryption.

WEP has been a target for hackers. In fact, tools freely available from the Internet, such as WEPCrack and Airsnort, are able to crack the WEP encryption mechanism. As a result, WEP is not strong enough for enterprise security.

If you plan to use WEP, configure the encryption key in the access point. Each radio card needs to be set to the same WEP key before it can associate with the access point. When using Windows XP, you are prompted to enter the WEP key if you try to connect to an access point that implements WEP. Windows displays this WEP key in the wireless networks list after you choose View Available Wireless Networks.

TKIP

The 802.11i working group has improved the security of 802.11 wireless LANs with an update to the 802.11 standard in 2004. The Temporal Key Integrity Protocol, for example, fixes the key-reuse problem of WEP. The TKIP process begins with a 128-bit “temporal key” shared among clients and access points. TKIP combines the temporal key with the client’s MAC address and then adds a relatively large 16-octet initialization vector to produce the key used to encrypt data. This procedure ensures that each station uses different key strings to encrypt data.

TKIP uses RC4 to perform the actual encryption of data frames, which is the same as WEP. A major difference from WEP, however, is that TKIP changes temporal keys periodically, according to a setting configured in the access point by an administrator. This provides a dynamic distribution method that significantly enhances network security.

AES

In addition to the TKIP solution, the 802.11i standard includes the Advanced Encryption Standard protocol. AES offers much stronger encryption than WEP or TKIP. In fact, the U.S. Commerce Department's National Institutes of Standards and Technology (NIST) organization chose AES to replace the aging Data Encryption Standard (DES). AES is now a Federal Information Processing Standard, FIPS Publication 197, that defines a cryptographic algorithm for use by U.S. government organizations to protect sensitive, unclassified information. The secretary of commerce approved the adoption of AES as an official government standard in May, 2002. Some of the older access points and radio cards do not support AES because it requires a specialized math coprocessor.

WPA

Before the ratification of the 802.11i standard, the Wi-Fi Alliance released the Wi-Fi Protected Access (WPA) standard, which most wireless LAN vendors rapidly adopted. WPA is actually a snapshot of the pre-ratified 802.11i standard involving TKIP and IEEE 802.1x standards. Eventually, the Wi-Fi Alliance released WPA2, which includes AES. This mirroring of standards has been effective for end users because the Wi-Fi Alliance requires special interoperability testing before a wireless LAN vendor can claim that its radio cards and access points are Wi-Fi Alliance certified.

Open System Authentication

Open system authentication is the default mode that 802.11 uses to authenticate radio cards to an access point. In this mode, a radio card sends an authentication frame to the access point, and the access point returns an authentication response. This form of authentication does not offer any real security. It is mainly part of the standard as a baseline authentication method.

When in the process of joining a network, the radio card completes open system authentication with the two-way handshaking process described earlier in the chapter. The radio card begins by sending an authentication frame to the access point, and the access point responds with an authentication frame. No credentials are passed during open system authentication; however, some vendors might implement provisions that must be met to authenticate stations.

Shared Key Authentication

802.11 shared key authentication goes a step further than open system authentication by using the common WEP key to authenticate radio cards. This is a four-way handshaking process:

KEY POINT

1. The radio card sends an authentication request.
2. The access point responds with an authentication frame containing challenge text, which is a string of unencrypted text.

3. The radio card encrypts the challenge text with the WEP key and sends the result to the access point.
4. The access point decrypts the challenge text with the common WEP key. If the challenge text is the same text that the access point initially sent, then the access point assumes that the radio card has the correct WEP key and that the radio card is a legitimate user.

Unfortunately, shared key authentication is easy to hack. In fact, a hacker can use freely available tools to readily find the WEP key. As a result, it is strongly advisable to not use shared key authentication.

Virtual Private Networks

To fully secure wireless connections, many companies require the use of VPN software on each user device to encrypt all communications between the user device and the remote system. The use of VPN software is especially important when users are communicating over public wireless LANs. Public Wi-Fi hotspots, for instance, do not implement any encryption over the wireless portion of the network. The VPN must protect data traffic.

In fact, some companies treat all wireless users as though they are operating from a public network—even those users who are inside the company's building. In this case, the wireless LAN access points connect to a distribution system that falls outside the firewall. This approach, however, might require an impractical number of VPN connections, which can be costly to deploy and support. For internal communications, it is possible to fully secure wireless users through the use of Layer 2 mechanisms.

Comparing Wireless Security

There are many options available for security, and you will need to make a decision on which one to use. Table 22-4 compares the various security mechanisms.

Table 22-4 *Wireless Security Mechanisms*

KEY POINT

	Strength	Keys	Standardization
WEP	Can crack with freely available tools	Static keys common to both the radio cards and access point	Part of initial 802.11 standard
TKIP	Adequate security for most wireless LANs	Unique keys automatically assigned to radio cards, and keys change periodically	Included in the 802.11i standard

continues

Table 22-4 *Wireless Security Mechanisms (Continued)*

	Strength	Keys	Standardization
AES	Very strong, good enough for some government systems	Unique keys automatically assigned to radio cards, and keys change periodically	Included in the 802.11i standard
WPA	Adequate for most wireless LANs and good enough for some government systems	Unique keys automatically assigned to radio cards, and keys change periodically	Ratified by the Wi-Fi Alliance

RF Signal Concepts

A major difference between wireless and wired networks is that wireless LANs use radio waves to transport information through the air. This introduces several new concepts, especially dealing with RF signals. When you are deploying wireless LANs, you must understand the attributes associated with RF signals in order to configure access points, avoid RF interference, and troubleshoot problems as they arise.

The key RF signal attributes that you need to understand are as follows:

KEY POINT

- Modulation
- RF signal characteristics
- Gain
- Signal-to-noise ratio (SNR)
- Spread spectrum
- Orthogonal frequency division multiplexing (OFDM)
- FCC rules
- RF interference
- Multipath

The sections that follow describe each of these in more detail.

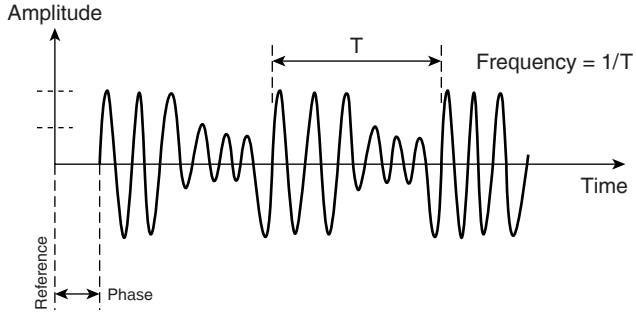
Modulation

An RF signal has characteristics that enable it to be sent from an antenna, through the air medium, and received by another antenna at the destination. RF signals are analog in nature. A computer, though, uses digital signals to represent bits of information.

Before transmitting data through the air, the transceiver within the radio cards and access points must convert digital signals into analog signals suitable for transmission through the air medium. As part of receiving an 802.11 frame, a radio card or access point must convert the analog signal back into a digital form that is understood by the computing device. This conversion process is known as *modulation* and *demodulation*, respectively.

802.11 defines several types of modulation, depending on which physical layer and data rate is in use by the 802.11 station. In general, it is possible to modulate an RF carrier signal by changing its *amplitude*, *frequency*, or *phase*, as shown in Figure 22-9. For example, 802.11b uses *phase shift keying (PSK)* to represent digital data. 802.11a and 802.11g, though, implement a combination of amplitude and phase shifts, which is referred to as *quadrature amplitude modulation (QAM)*.

Figure 22-9 *Attributes of an RF Signal*



RF Signal Characteristics

RF signals are cyclic and vary in time continuously. The number of cycles that occur in the signal per second is its frequency, which can vary throughout what is referred to as the *frequency spectrum*. The unit of frequency is hertz (Hz), and wireless LAN signals fall roughly into the 2.4-GHz and 5-GHz portions of the frequency spectrum. The process of modulation causes the RF signal to occupy a portion of the frequency spectrum, which is known as *bandwidth*.

In addition to frequency, an RF signal at any time has specific *amplitude*. There are many ways to represent signal amplitude, but the most common with RF systems is signal power. The applicable unit for power is watt (W) or decibels relative to 1 milliwatt (dBm). The FCC has rules for maximum transmitter output depending on the standard in use. For example, the maximum transceiver output for 802.11b is 1 W. In general, higher transmit power enables longer-range operation.

Most wireless LAN systems have RF signals that fall into the milliwatt (mW) range, which makes the addition and multiplication of RF signals mathematically difficult. As a result, it is

advantageous to convert watts to dBm, which is a logarithmic value that references the signal power to 1 mW. The conversion formula is as follows:

dBm = 10 log (mW)

For example, 100 mW equals 20 dBm.

Gain

The components of a wireless LAN offer varying degrees of *gain*, which represents how much a signal changes from one point to another. The gain in dB is simply the signal level at the output of a device (in dBm) minus the signal level at the input of the device (in dBm). The decibel is a unit that represents change in signal amplitude. A signal experiences a gain of 3 dB, for example, when it increases from 50 mW (17 dBm) to 100 mW (20 dBm). An amplifier or antenna may offer this gain to the signal.

Attenuation, or loss, is the inverse concept to gain. If the signal goes from 20 dBm to 17 dBm, then the signal experiences attenuation of 3 dB. This can also be expressed as –3 dB gain. Antenna cabling and obstacles in a facility, such as walls and furniture, introduce attenuation. In addition, freespace loss, which is dependent on frequency and path distance, is a form of attenuation. The freespace loss occurs due to attenuation of the air medium and contributes to the majority of the total loss from the transmitter to the receiver.

With wireless LANs, the RF signal amplitude must be of a specific minimum value before the radio card will detect the signal. This value depends on the 802.11 physical layer and data rate in use, but is approximately –85 dBm.

Antennas have gain, which impacts directivity. An omnidirectional antenna, for instance, has a gain of 6 dB, or more, depending on the antenna design. Higher-gain antennas become more directive, with higher gains providing narrower beamwidths and longer range, as shown in part B of Figure 22-4. Antenna gain comes from focusing a given amount of RF power into a narrower pattern, or beamwidth—again, much as a flashlight does.

Signal-to-Noise Ratio

If noise at the radio card is high, the radio card will have difficulty recovering the signal, which results in bits errors and retransmissions. An important signal measurement is the *signal-to-noise ratio (SNR)*. The SNR (in dB) at a particular point in the network is simply the signal power (in dBm) minus the noise power (in dBm). A signal power of –65 dBm and noise power of –90 dBm yields an SNR of 25 dB. The noise power is anything other than signals corresponding to the access point or radio card.

Table 22-5 includes several SNR values and the resulting performance of an 802.11b network while using a Windows XP laptop browsing websites and downloading files. The results in Table 22-5 depict general end-user performance. The use of SNR to define the range boundary of an access point radio cell is more effective than either data rates or signal amplitudes. The higher degrees of noise cause more errors in frames and corresponding retransmissions. Note that Windows XP shows an indication of the signal strength in a series of bars on a graph, showing no bars as poor quality and five bars as the best quality.

Table 22-5 *Correlation of SNR Values to Wireless LAN Performance*

KEY POINT

SNR Value	Signal Indication (Windows XP)	Performance
> 40 dB	Excellent signal strength (5 bars); always connected with the access point	Extremely fast web browsing and file download
25–40 dB	Very good signal strength (3 to 4 bars); always connected with the access point	Very fast web browsing and file download
15–25 dB	Low signal strength (2 bars); always connected with the access point	Usually fast web browsing and file download
10–15 dB	Very low signal strength (1 bar); sometimes disconnected from the access point	Mostly slow web browsing and file download
5–10 dB	No signal strength (no bars); not connected with the access point	No network services

Spread Spectrum

In 1985, the U.S. FCC adopted regulations that specify the availability of license-free frequency bands in the 900-MHz, 2.4-GHz, and 5-GHz portions of the frequency spectrum. To be compliant with the rules for these bands, however, equipment must use spread spectrum or OFDM methods to spread the signal power over a relatively wide portion of the frequency spectrum. This approach promotes frequency reuse—otherwise known as *sharing*—of these bands by multiple users, with a low statistical probability of interference.

Spread spectrum was the first method in use by wireless LAN vendors. There are two types of spread spectrum:

KEY POINT

- **Frequency hopping spread spectrum (FHSS)**—With FHSS in the 2.4-GHz band, for example, the transceiver periodically tunes its transmitter and receiver to a different carrier frequency within approximately 84 MHz of bandwidth. Hopping from one frequency to another is done according to a hopping sequence programmed in each of the stations. The other stations receiving the frames tune their receivers to a specific frequency based on the

hopping sequence. The RF signal occupies approximately a 2-MHz channel. Because the hopping occurs very often (many times per second) and uniformly over the entire band, the signal appears to occupy the entire 84 MHz. The 802.11 frequency hopping physical layer standard enables data rates of 1 Mbps and 2 Mbps.

- **Direct sequence spread spectrum (DSSS)**—DSSS uses a coding technique to spread the signal over the frequency spectrum. 802.11b uses direct sequence, which spreads the carrier signal over approximately one third (30 MHz) of the 2.4-GHz band. With DSSS, a chipping code represents each data bit that needs transmission. This increases the signal rate by the number of bits in the chipping code (11 total). The increase in signal rate effectively spreads the RF signal. The differences between frequency hopping and direct sequence had been under debate for a number of years, but the 802.11 working group finally selected direct sequence for extending the initial 1-Mbps and 2-Mbps 802.11 data rates to include rates up to 11 Mbps.

Orthogonal Frequency Division Multiplexing

OFDM is not a form of spread spectrum. Instead, OFDM divides a data signal across 48 separate subcarriers within a 20-MHz channel to provide transmissions of 6, 9, 12, 18, 24, 36, 48, or 54 Mbps. Data rates of 6 Mbps, 12 Mbps, and 24 Mbps are mandatory for all 802.11-compliant products. OFDM is extremely efficient, which enables it to provide the higher data rates. In addition, OFDM is highly immune to multipath propagation problems that cause significant performance issues with spread-spectrum techniques.

An 802.11a modulator converts the binary signal into an analog OFDM waveform through the use of different modulation types, depending on which data rate is chosen. For example, with 6-Mbps operation, the PMD uses *binary phase shift keying (BPSK)*, which shifts the phase of the transmit center frequency to represent different data bit patterns. The higher data rates, such as 54 Mbps, employ *quadrature amplitude modulation (QAM)* to represent data bits by varying the transmit center frequency with different amplitude levels in addition to phase shifts.

Table 22-6 summarizes the primary attributes of the various wireless LAN technologies.

Table 22-6 *Wireless LAN Technology Comparison*

KEY POINT

	Spread Spectrum?	Max Data Rate	Standards
FHSS	Yes	2 Mbps	802.11 FHSS
DSSS	Yes	11 Mbps	802.11b
OFDM	No	54 Mbps	802.11a and 802.11g

FCC Rules

In general, the U.S. FCC does not require users to license wireless LAN products, assuming that the user does not exceed certain emission limits. The FCC uses Effective Isotropic Radiated Power (EIRP) as a factor for determining whether a wireless LAN is in compliance with regulatory rules. EIRP equals the transmit power (in dBm) minus cable and connector losses (in dB) and plus the antenna gain (in dB). For 802.11b/g access points and radio cards, the EIRP can be up to 36 dBm, which includes a transmit power up to 30 dBm (4 watts) and 6-dBi antenna gain.

In addition, the user must obtain FCC licensing for the wireless LAN solution when using antennas or amplifiers that are not part of the access point vendor's products certified with the FCC. This is necessary to ensure that the proposed wireless system will not interfere with existing systems at the location of operation. Regulatory agencies in other countries have similar rules, but they differ slightly depending on the country. As mentioned earlier, research your country's rules and deploy your systems based on them.

RF Interference

Because of the use of radio waves, wireless LANs are susceptible to several sources of RF interference. Interfering signals cause delay and reduce throughput. If no 802.11 frame transmissions are in process when the interference is present, then the result will be medium access delays. In some cases, this delay can be indefinite.

RF interference causes data frames in transit to become corrupted, resulting in a retransmission. The destination disregards the incoming frame, because its error-checking mechanism indicates errors in the frame. As a result, the destination station does not send an acknowledgement. After a period of time, the sending station retransmits the frame. This adds delay and cuts throughput.

RF interference in the 2.4-GHz band comes from microwave ovens, cordless phones, Bluetooth devices, and other wireless LANs. Most microwave ovens, for instance, emanate RF energy over roughly one third of the 2.4-GHz band. The actual frequencies that this involves, however, vary from one microwave oven to another. If an access point is set to a channel that falls within the affected frequencies, then significant interference and decrease in throughput will occur. Microwave oven interference can occur for many tens of feet away from the microwave oven—farther if you are using a directional antenna oriented toward the interference source. It is generally possible, though, to tune the access point to a channel where the interference is minimal.

Cordless phones that operate in the 2.4-GHz band are popular, and they can cause negative impacts similar to microwave ovens. Interference is worse when someone is actively using the phone, but remote phone stations often communicate wirelessly with the base station even when no call is taking place. In addition, some cordless phones that use frequency hopping cause interference

across the entire 2.4-GHz band, making it impossible to tune the access point away from the interference. The only solution in such situations, if you want to keep the phones, is physical separation. If the interference is still causing significant interference, then consider replacing the phones with 900-MHz or 5.8-GHz phones.

Another source of RF interference is a neighboring wireless LAN operating within the same part of the frequency band you are using—for example, a nearby wireless LAN located in the same office complex. When deploying your wireless LAN, it is important to avoid assigning channels to access points that overlap with the neighboring access points. Otherwise, performance of both networks will suffer.

Because of the potential for RF interference and its negative impact on performance, it is very important to analyze the presence of potential RF interference before you install a wireless LAN. This is generally done through an initial RF site survey that measures the presence of interference sources to determine potential problems. If too many issues related to interference exist, then you should consider using a wireless LAN that operates in a different band of frequencies. For example, consider using 802.11a instead of 802.11b/g.

Multipath

Multipath interference occurs when an RF signal takes different paths when propagating from one wireless station to another. While the signal is en route, walls, chairs, desks, and other items get in the way and cause the signal to bounce in different directions. A portion of the signal might go directly to the destination, and another part might bounce from a chair to the ceiling, and then to the destination. As a result, some of the signal will encounter delay, because it has to travel over a longer path to the receiver.

Multipath causes the information symbols received by an 802.11 signal to overlap, which causes the receiver to have difficulty demodulating the signal. This effect is often referred to as *intersymbol interference (ISI)*. Because the shape of the signal conveys the information being transmitted, the receiver will demodulate errored data. If the delays are great enough, bit errors in the packet will occur. The receiver will not be able to distinguish the symbols and interpret the corresponding bits correctly. As a result, the sending station will have to retransmit the affected frames.

Because of retransmissions, users encounter lower throughput when multipath is significant. The reduction in throughput depends on the environment. As examples, 802.11 signals in homes and offices might encounter 50 nanoseconds of multipath delay, while signals in a manufacturing plant could encounter multipath delay as long as 300 nanoseconds. Based on these values, multipath is not too much of a problem in homes and offices. Metal machinery and racks in a plant, however, provide a lot of reflective surfaces from which RF signals may bounce and take erratic paths. Thus,

be wary of multipath problems in warehouses, processing plants, and other areas full of irregular, metal obstacles.

Antenna diversity can aid in combating multipath propagation. An access point may implement a spatial diversity antenna system, which consists of two antennas that interchangeably receive and transmit radio signals. An access point receives a signal on both antennas, but because of multipath propagation and interference, the same signal often does not reach both antennas at the same time and strength. The access point then performs internal calculations to optimize the received signal. The main benefits of spatial diversity antenna systems are improved coverage and signal reception.

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most Cisco Press *Exam Certification Guides*, this book does not repeat information listed in the "Foundation Topics" portion of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with Key Topic icons.

The 802.11 standard is a very important basis for understanding the operation and configuration options for a wireless LAN solution. Most wireless LANs include access points, which interface wireless users to a physical network and forward 802.11 data frames between wireless users that are associated with various access points. Ad hoc wireless LANs, however, do not use access points, and allow wireless users to send data frames directly to each other.

There are several configuration parameters that you can set in access points and radio cards to optimize performance. For example, fragmentation and RTS/CTS functions can improve the throughput of a wireless LAN in some situations. Radio waves are very different from signals that travel over a wired medium. RF interference, multipath propagation, and various sources of attenuation affect radio waves and cause errors in frame transmissions. You must be aware of these issues and plan wireless LAN deployment accordingly.

The new 802.11i standard offers solid security for wireless LANs, with TKIP and AES replacing the vulnerable WEP protocol. The 802.11 standards include several physical layers with varying degrees of interoperability and performance.

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Topic icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

infrastructure mode, ad hoc mode, passive scanning, DTIM interval, active scanning, SSID, power-save mode, RTS/CTS, fragmentation, transmit power, distributed coordination function, point coordination function, network allocation vector, WEP, TKIP, association ID, AES, WPA, SNR, spread spectrum, RF channel, FHSS, beacon, DSSS, OFDM, multipath, 802.11a, 802.11b, 802.11g, 802.11n

Refer to the CD-based glossary to check your answers.

Further Reading

For more details regarding wireless LANs, consider reading the following Cisco Press books:

- *Wireless Networks First-Step*, by Jim Geier
- *802.11 Wireless LAN Fundamentals*, by Pejman Roshan and Jonathan Leary

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- Enterprise Wireless Mobility
  - — Hardware
  - — SWAN
  - — VoWLAN
  - — Products

CHAPTER 23

Wireless LAN Solutions

The application of wireless LANs, especially for supporting voice, requires careful selection of components, device configuration, and management. This is important to create a solution that offers required levels of performance and security that enable the lowest total cost of ownership. This chapter focuses on deploying wireless LANs for various applications.

"Do I Know This Already?" Quiz

Table 23-1 outlines the major sections in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 23-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
Cisco Structured Wireless-Aware Network	1 through 4	
Applying Wireless LANs in Enterprises	5 through 7	
Public Wireless LANs	8 and 9	
Small Office and Home Wireless LANs	10	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. What is Cisco Wireless Domain Services (WDS)?

e. A collection of Cisco access points that comprises a secure wireless LAN

f. A set of Cisco IOS Software features that enhances and simplifies wireless LAN client mobility, security, deployment, and management

g. A security solution that Cisco no longer supports

h. A web-based service that Cisco offers on a paid subscription basis

2. What are required components of Cisco SWAN?
   a. Cisco Aironet Series access points and 802.1x authentication server
   b. Cisco Aironet Series access points, WLSE, and Cisco Aironet client devices
   c. WLSE and 802.1x authentication server
   d. Cisco Aironet Series access points, WLSE, and 802.1x authentication server

3. Which Cisco access point is designed for outdoor campus networks?
   a. Cisco Aironet 1300 Series
   b. Cisco Aironet 1200 Series
   c. Cisco Aironet 1100 Series
   d. Cisco Aironet 1000 Series

4. The Cisco Aironet 350 Series access point complies with which of the following standards?
   a. 802.11n
   b. 802.11b
   c. 802.11g
   d. 802.11e

5. Which layer of the network architecture should you focus on when providing wireless LAN security in enterprises?
   a. Layer 1
   b. Layer 4
   c. Layer 3
   d. Layer 2

6. What are sufficient RF signal characteristics for supporting voice services over wireless LANs?
   a. 20-dB SNR with approximately 20 percent overlap between adjacent radio cells
   b. 25-dB SNR with approximately 20 percent overlap between adjacent radio cells
   c. Minimum of 36-Mbps data-rate associations between the client devices and the access points
   d. Minimum of 11-Mbps data-rate associations between the client devices and the access points

7. How can you avoid the impacts of RF interference on voice services operating over wireless LANs?

   a. Use 802.11b/g interfaces

   b. Use additional Wi-Fi phones to drown out the interference

   c. Use 802.11a interfaces

   d. Decrease the access point transmit power

8. Why should you enable the broadcasting of SSIDs in beacon frames sent by the access points in a public wireless LAN?

   a. The network will have better security

   b. Users will see the wireless network in Windows

   c. Power management will be more efficient

   d. Network delays will decrease

9. What are the primary configuration differences between a public and enterprise wireless LAN?

   a. Security and billing components

   b. Performance and billing components

   c. Access point hardware

   d. Management mechanisms

10. Why do you need a Wi-Fi router instead of an access point when installing a wireless LAN in a small office or home?

    a. A router operates at a faster speed

    b. A router implements better security mechanisms

    c. A router will last longer

    d. The router implements DHCP and NAT

Foundation Topics

Cisco Structured Wireless-Aware Network

Cisco Structured Wireless-Aware Network (SWAN) is a framework for integrating wired and wireless networks based on the Cisco Systems product line of wireless LAN products. SWAN is an architecture that embodies the services, protocols, and tools necessary to deploy effective solutions with minimal total cost of ownership.

This section further explains the SWAN components.

Wireless Domain Services

Wireless Domain Services (WDS) is a set of Cisco IOS Software features that enhances and simplifies wireless LAN client mobility, security, deployment, and management. WDS offers the following primary services for SWAN:

KEY POINT

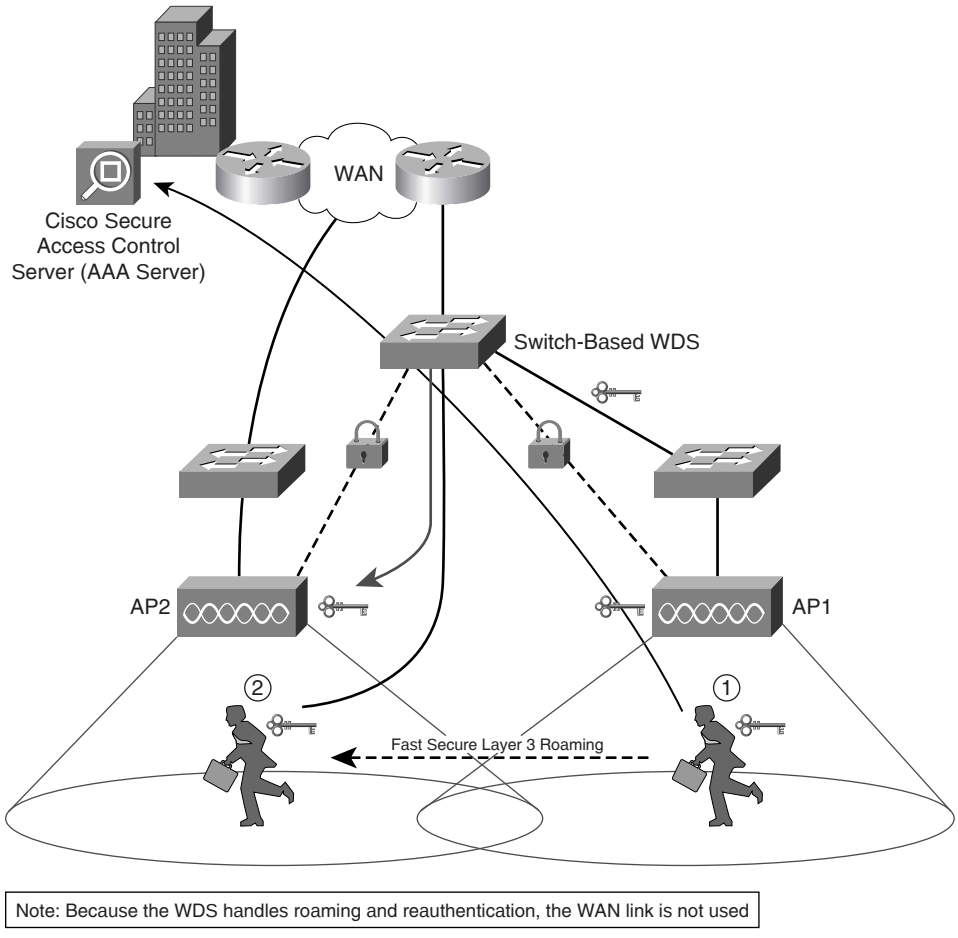
- **Fast Secure Roaming (FSR)**—For time-sensitive applications, enables a wireless client to securely roam between access points in the same subnet or between subnets, enhances channel scanning, and provides fast IEEE 802.1X rekeying. Access point handoff times are within 50 ms, which is crucial for effective VoIP applications while users are roaming about the facility.
- **Radio management aggregation**—Reduces the bandwidth necessary for radio management information, such as access point status messages, that is sent across the network, by eliminating redundant management information. Radio management information is sent to the CiscoWorks WLSE and provides the basis for monitoring functions, such as rogue access point detection and location.
- **Client tracking**—Records client authentication and roaming events, which are sent to the CiscoWorks WLSE to monitor client associations to specific access points.

Figure 23-1 illustrates how the FSR feature of WDS works:

1. AP1 must initially 802.1x authenticate with the WDS device to establish a secure connection. The initial client authentication goes to a central AAA server to authenticate the user and authorize specific services. This occurs in approximately 500 ms.
2. When the client roams, the client informs WDS that roaming is taking place, and WDS sends the applicable key to the new access point (AP2 in this example). The handoff time between the access points is approximately 50 ms.

Figure 23-1 *Cisco SWAN Fast Secure Roaming*

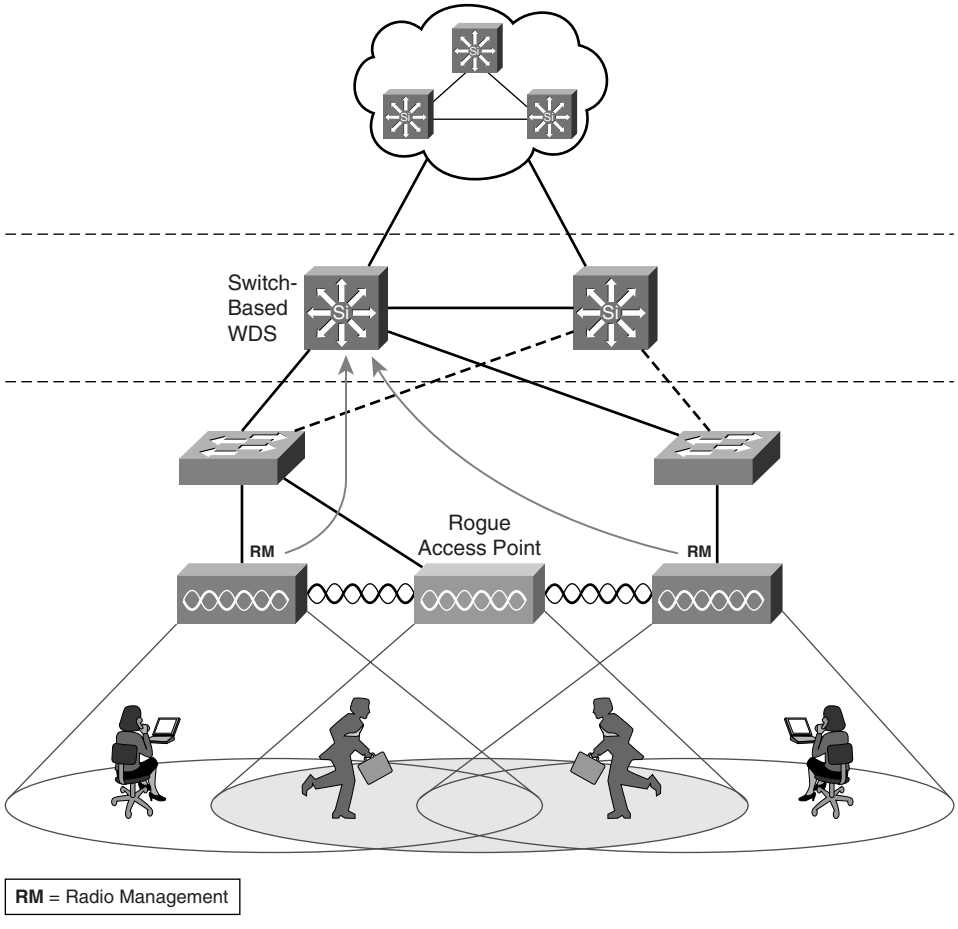
KEY POINT



Intrusion Detection System

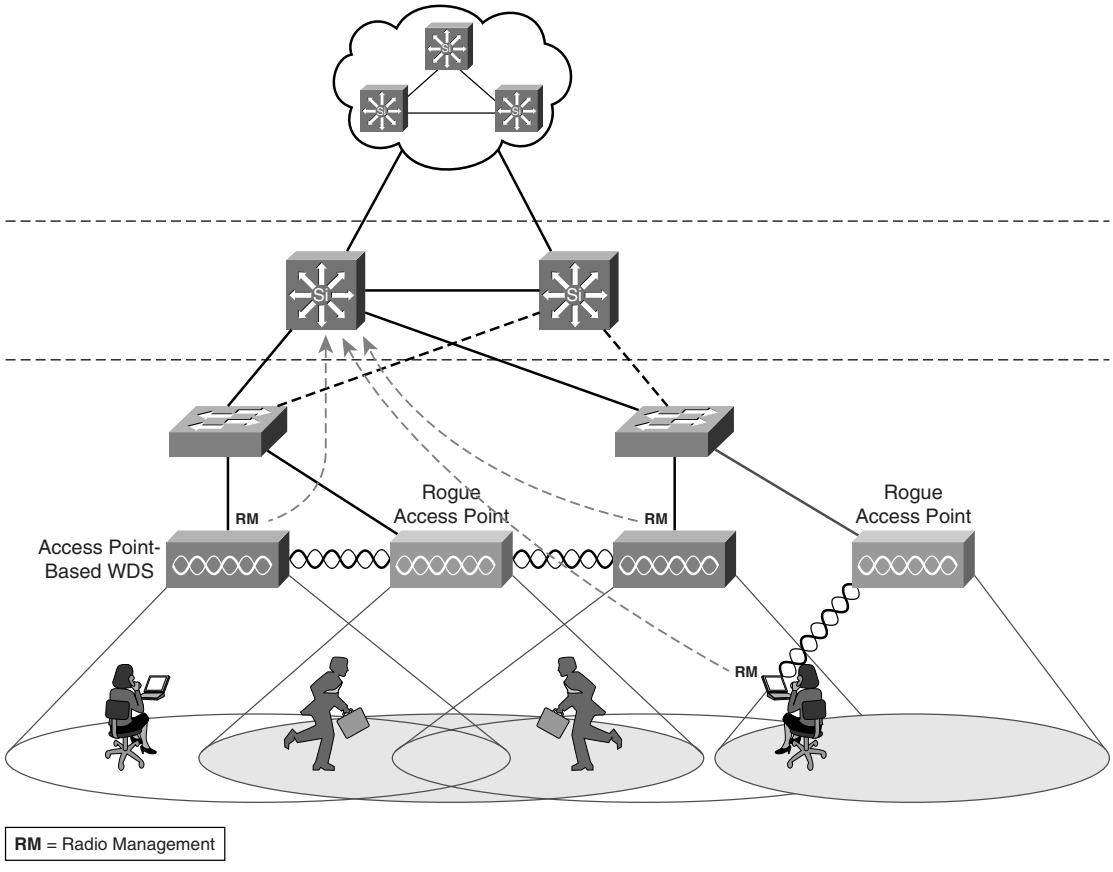
Cisco SWAN includes the Wireless LAN Threat Defense Solution, which includes an intrusion detection system (IDS) (refer to Figure 23-2). This safeguards the wireless LAN from malicious and unauthorized access. For example, the IDS detects and suppresses rogue access points by disallowing them to authenticate with the network, and identifies unassociated clients through MAC address association tables. The IDS integrates with the Cisco Self-Defending Network, the Cisco vision for network security.

Figure 23-2 *Cisco Wireless LAN Threat Defense Solution*



The IDS provides an optional capability for using Cisco Aironet and Cisco-compatible client devices to continuously scan and monitor the RF environment. The client devices work jointly with Cisco Aironet access points to regularly measure RF activity. This client-assisted rogue access point scanning and monitoring increases rogue access point detection and enhances the security of the network. As shown in Figure 23-3, the radio management (RM) element in the client device identifies a rogue access point and reports relevant findings to WLSE. The RM element looks for access point configurations that indicate a rogue, such as an unauthorized SSID and MAC address.

Figure 23-3 *Cisco IDS with Client Scanning*



Cisco SWAN Hardware

When building a wireless LAN based on Cisco SWAN, a company must choose components designed to fit into the architecture. Cisco SWAN includes the components described in Table 23-2.

Table 23-2 *Cisco SWAN Hardware*

KEY POINT

Hardware Type	Function
Cisco Aironet Series access points	These access points, which must run Cisco IOS Software to be part of SWAN, are a mandatory component of Cisco SWAN. They enable roaming throughout the network and interconnect wireless LAN users to the wired network.

continues

Table 23-2 *Cisco SWAN Hardware (Continued)*

Hardware Type	Function
Management and security servers	Cisco SWAN requires the use of the CiscoWorks Wireless LAN Solution Engine (WLSE) and an IEEE 802.1x authentication server, such as Cisco Secure Access Control Server (ACS), for management and security of the wireless LAN.
Wireless LAN client devices	The client devices must be Wi-Fi certified or IEEE 802.11 client adapters. Cisco Aironet or Cisco-compatible client devices, which are optional, offer enhanced features, such as better security, enhanced interoperability, and extended radio management.
Infrastructure devices	Cisco incorporates wireless capabilities into its switches and routers, such as the Cisco Catalyst 6500 Series Wireless LAN Services Module (WLSM), which creates a unified network system that interoperates effectively with Cisco SWAN access points. WLSM allows no loss of Layer 3 connectivity as users roam from access point to access point across a large campus. Each WLSM supports up to 300 access points and 6000 wireless clients.

Cisco Wireless LAN Hardware

Cisco has a complete line of wireless LAN hardware that addresses the needs of enterprises, public networks, and homes. The following list identifies each of these devices, by category, that integrate into SWAN:

- **Access points**:
  - **Cisco Aironet 1300 Series**—A multifunctional component that provides access point and bridge functionality for network connections within an outdoor campus area. The 1300 Series supports the 802.11b/g standards.
  - **Cisco Aironet 1230AG Series**—Has dual antenna connectors for extending range, with optional antennas for enterprise solutions.
  - **Cisco Aironet 1200 Series**—Includes a dual-slot architecture that allows flexibility when configuring radio cards for enterprise solutions. For example, it can include any combination of radio card technology, such as 802.11a and 802.11g.
  - **Cisco Aironet 1130AG Series**—Includes integrated antennas and dual 802.11a/g radios for enterprise solutions.
  - **Cisco Aironet 1100 Series**—Offers an easy-to-install, single-band, 802.11b/g access point for enterprise solutions.

  - **Cisco Aironet 350 Series**—Designed for small and medium-sized businesses, provides an ideal solution for customers who desire a non-upgradeable IEEE 802.11b solution. Supports the 802.11b standard.

- **Wireless bridges and workgroup bridges**:
  - **Cisco Aironet 1400 Series Wireless Bridge**—Connects multiple LANs in a metropolitan area. Supports both point-to-point and point-to-multipoint configurations, with data rates up to 54 Mbps.
  - **Cisco Aironet 1300 Series Outdoor Access Point/Bridge**—A multifunctional component that provides access point, bridge, and workgroup bridge functionality for network connections within an outdoor campus area. Supports the 802.11b/g standards.
- **Lightweight access points**—Formerly an Airespace product, the Cisco 1000 Series Lightweight Access Point is an 802.11a/b/g, zero-touch configuration and management access point for enterprise solutions. Works in conjunction with a Cisco Wireless LAN Controller and optional Cisco Wireless Control System (WCS) to support real-time intrusion monitoring in addition to data traffic.
- **Cisco wireless LAN client adapters**:
  - **Cisco Aironet 802.11a/b/g Wireless CardBus Adapter**—Designed for laptops and tablet PCs.
  - **Cisco Aironet 802.11a/b/g Wireless PCI Adapter**—Designed for desktop and point-of-sale devices.
  - **Cisco Aironet 350 Wireless LAN Client Adapter**—Available in both PC Card (PCMCIA) and PCI form factors and supports 802.11b connections.
  - **Cisco Aironet 5-GHz 54-Mbps Wireless LAN Client Adapter (CB20A)**—Compliant with 802.11a and supports CardBus standards.
- **Wireless LAN controllers**:
  - **Cisco 4100 Series Wireless LAN Controller**—Works in conjunction with the Cisco 1000 Series Lightweight access points and Cisco WCS to provide system-wide functions, such as intrusion detection, RF management, and security policy management. Ideal for medium-to-large enterprise facilities.
  - **Cisco 2000 Series Wireless LAN Controller**—Similar to the 4100 Series, but is best for small-to-medium enterprise facilities because they support fewer access points.

NOTE While the current CCIE Routing and Switching written exam blueprint does not typically list products, wireless is the only such case in which the blueprint does mention products. Obviously, the product mix will change over time, with rapid changes in the Cisco WLAN product mix likely between the completion date of this chapter, publication date, and into the early life of this book. You may want to focus more on the types of features implemented by types of products. You can also refer to http://www.cisco.com/en/US/partner/products/hw/wireless/index.html for more information on product changes (requires a CCO username/password).

CiscoWorks Wireless LAN Solution Engine

CiscoWorks WLSE is a centralized network management system for Cisco Aironet solutions. WLSE is a key component of SWAN and consists of the following features:

KEY POINT

- Automatic access point configuration
- Assisted site surveys
- Centralized firmware updates
- Dynamic grouping
- VLAN configuration
- Multiple service set identifier (SSID) support
- Customizable thresholds
- Fault status
- Intrusion detection system
- Security policy monitoring
- Secure user interface
- Air/RF scanning and monitoring
- Self-healing functions
- Reporting, trending, planning, and troubleshooting

These features are discussed, in turn, in the following sections.

Automatic Access Point Configuration

WLSE automatically discovers and configures Cisco Aironet access points based on access point type, subnet, and software version. This eliminates the need to manually configure each access point separately. WLSE allows the administrator to update any of the access point configurations, such as

WPA security settings, SSID, and RF channel. WLSE can also perform mass upgrades of older Cisco access points running VxWorks to newer Cisco IOS Software versions. WLSE stores the last four configuration versions for each access point so that an administrator can easily undo changes.

Assisted Site Surveys

WLSE's assisted site survey tool automatically identifies optimal RF channels and transmit power and periodically assesses performance with respect to baseline site-survey settings. These features ease wireless LAN installation by reducing the effort needed to perform RF testing in the facility prior to installing the network. WLSE generates notifications to the administrator when applicable configuration updates are necessary as the RF dynamics of the facility change over time.

Centralized Firmware Updates

WLSE allows administrators to update firmware on access points and bridges on an individual or group basis. Timely firmware updates are critical for ensuring optimum performance, reliability, and security of the network.

Dynamic Grouping

Administrators can group access points that span different subnets into different groups to enable more intuitive network management. For example, one group of access points may be named "public," and another group may be named "Engineering," regardless of where the access points physically reside on the network. This is similar to the concept of multiple VLANs at Layer 2.

VLAN Configuration

WLSE allows administrators to centrally configure and monitor VLANs on access points. This feature enables the administrator to separate traffic among different groups of users associating with the same access point. For example, one VLAN may be assigned to public users, and a different VLAN may be provided for staff members.

Multiple Service Set Identifier Support

WLSE allows the configuration of up to eight broadcast SSIDs per access point radio. Each SSID can be assigned to a particular VLAN, facilitating the use of VLANs to separate user traffic. The SSID "public," for example, may tie to the VLAN connecting to the Internet from outside the DMZ of the company network.

Customizable Thresholds

Administrators can define a variety of faults and performance thresholds, such as network load, RF usage, errors, and clients associations, and specify actions and fault priorities. If data traffic

through a particular access point reaches capacity, for instance, WLSE can send an alert to the administrator via SNMP.

Fault Status

WLSE displays a view of all access points and device groups, with color coding and group icons that indicate fault status. Fault notifications are done via Syslog messages, SNMP traps, and e-mail. This is especially important with wireless LANs, because a faulty access point, possibly due to a broken antenna, could go unnoticed for weeks or months if no monitoring functions are available. Users in this situation tend to adapt to the resulting coverage hole by moving to a different part of the facility to maintain connectivity. Careful monitoring of the fault status of access points eliminates this problem.

Intrusion Detection System

WLSE detects unauthorized access points and tracks wireless clients participating in the wireless LAN. For example, WLSE detects clients spoofing authorized MAC addresses, excessive probe requests, and unusual deauthentication frames that indicate potential man-in-the-middle or DoS attacks.

Security Policy Monitoring

WLSE monitors the network via SNMP and ensures that all access points are configured to ensure adherence to security policies. If an improper configuration is found, WLSE issues alerts via e-mail, Syslog, or SNMP trap notifications. This precludes someone from making use of a rogue access point to attach to the corporate network. WLSE detects improper configuration of the rogue device and promptly alerts the administrator.

Secure User Interface

WLSE includes a secure, role-based, HTML user interface to facilitate remote access to the management functions. As a result, an administrator can use WLSE functions while sitting in an office, when traveling, or from home. All communications between WLSE and the access points is done via SSL.

Air/RF Scanning and Monitoring

Cisco Aironet access points have integrated RF scanning and measurement features that collect information regarding the RF environment, which may include rogue access points and users. WLSE analyzes this data and provides reports and alerts when rogue devices are found or when RF coverage is not optimum. WLSE also helps determine the source of RF interference. These features reduce the need to install dedicated sensing devices to monitor for rogue access points.

Self-Healing Functions

If an access point fails, WLSE can automatically increase the power and corresponding coverage of surrounding access points to compensate for the coverage hole. This quickly fixes coverage hole problems while the administrator replaces the failed access point. WLSE also has a backup mechanism that automatically takes over and notifies the administrator if the primary WLSE fails.

Reporting, Trending, Planning, and Troubleshooting

WLSE tracks actions made by clients to aid in troubleshooting network access problems. For example, a user many be having troubles associating with an access point. The administrator or help desk can view recent transactions with the applicable access point and determine the source of the problem.

Applying Wireless LANs in Enterprises

There are many applications for wireless LANs in enterprises. For example, a company might deploy a wireless LAN to mobilize its workforce. Larger enterprises, such as Microsoft, have done this and achieved significant returns on investment through the gain in efficiencies. The ability for an employee to access e-mail and other documents from anywhere within the facility saves a significant amount of time and money. Employees are more productive and able to respond to specific events faster. A sales representative, for example, can respond to an inquiry from a customer while attending a meeting in the conference room rather than waiting until after returning to the office.

Another use for wireless LANs in the enterprise is to provide a wireless network for customers and consultants to easily access Internet services while visiting the company's facilities. In this case, a company places the access points outside the DMZ and treats the network as a public network found at typical Wi-Fi hotspots. The availability of networking services in this situation helps both the company and its visitors by strengthening the ability to communicate.

Enterprise Security

Security is one of the most important aspects of an enterprise wireless LAN. Without proper security mechanisms in place, a company is vulnerable to hackers gaining access to unauthorized information and possibly destroying network resources. The goal of effective security is to use a combination of proven security practices to ensure that the company's information systems assets are safe.

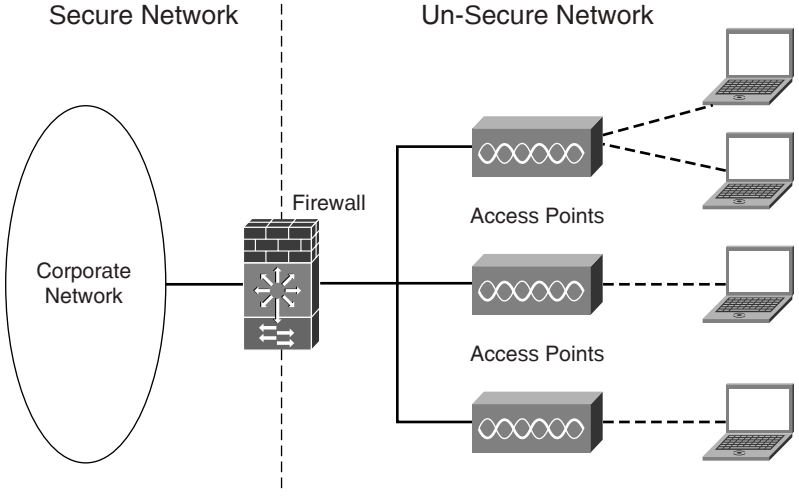
To properly secure an enterprise wireless LAN, first establish wireless security polices. Before installing the wireless LAN, consider requirements and establish proven security policies that provide adequate protection. These policies should mandate company control of the installation of wireless LAN components and address architectural elements, such as encryption and

authentication protocols, limits of RF emanation outside the facility, and access point physical mounting restrictions.

Most installations should focus on implementing Layer 2 security. Most companies deploying wireless LANs implement Layer 2 security to apply security between client devices and access points. In this case, the access points connect directly to the corporate network. This is a cost-effective method for providing security throughout the enterprise, especially when there are a relatively large number of wireless users. Wi-Fi Protected Access (WPA) is a good encryption mechanism to use for this purpose because it automatically assigns encryption keys periodically to client devices.

If a significant number of visitors need wireless access to Internet services, however, it might be more practical to connect the access points outside the DMZ and require employees to use VPN client software to access corporate resources. Figure 23-4 illustrates this approach. If many employees need wireless connections, however, this approach could be relatively expensive due to the significant number of VPN connections needed with the corporate system.

Figure 23-4 *Public Wireless LAN Within an Enterprise*



Most installations should try to limit propagation of radio signals outside the facility. As a precaution, consider designing the wireless LAN in a way that limits radio signals from being received outside the facility. This minimizes the ability for a hacker to associate with one of the wireless LAN access points. A company can reduce radio propagation outside the building by properly aligning antennas and reducing the transmit power of access points.

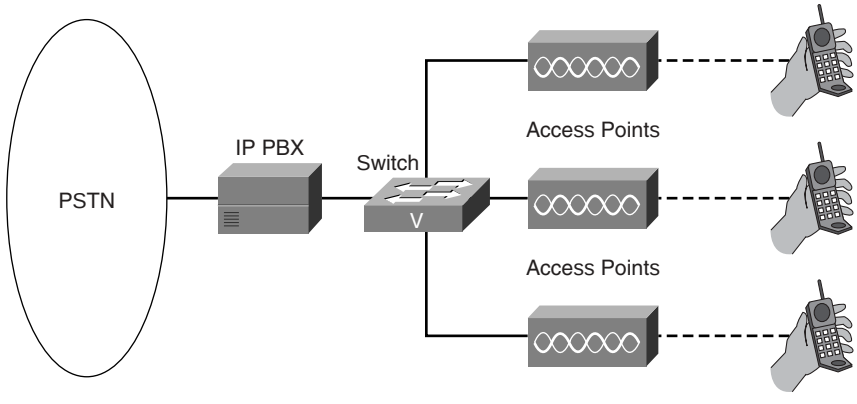
Voice Services

Voice service over wireless LANs is becoming the killer application that is causing the industry to explode. Many mobile device makers are beginning to include integrated Wi-Fi in smart phones and PDAs along with cellular interfaces, and enterprises are beginning to understand the significant advantages of using voice services, along with data, over wireless LANs. The Cisco Wireless IP Phone 7920, for instance, interfaces with 802.11b access points. Incredible gains in efficiency are possible by equipping employees with wireless phones for use throughout the enterprise, and the use of a common wireless LAN infrastructure is extremely cost-effective for supporting both voice and data services.

Figure 23-5 illustrates a typical voice over wireless LAN (VoWLAN) configuration. The Wi-Fi phone communicates with an IP PBX. The IP PBX, such as a Cisco CallManager (CCM), interfaces with the phone by using IP and provides connections to the Public Switched Telephone Network (PSTN). The IP PBX handles calls within the facility and interfaces with the PSTN for external calls. VoIP gateways are available to interface Wi-Fi phones to a legacy PBX.

Figure 23-5 *Voice over Wireless LAN Configuration*

KEY POINT



When deploying voice services over wireless LANs, consider the suggestions found in Table 23-3.

Table 23-3 *Voice over Wireless LAN Deployment Tips*

KEY POINT

Tip	Justification Details
Perform accurate RF site surveys	To avoid coverage holes and ensure proper signal levels, a company should perform an accurate RF site survey when determining the optimum positioning of access points. Cisco has guidelines for deploying Wi-Fi phones, which describe the need for a minimum of 25-dB SNR with approximately 20 percent overlap between adjacent radio cells. This is somewhat higher SNR than what is needed for data-only applications.

continues

Table 23-3 *Voice over Wireless LAN Deployment Tips (Continued)*

Tip	Justification Details
Choose access points that support fast roaming	Wi-Fi phones require a fast handoff between access points as users roam throughout the facility. In general, the handoff should be under 100 ms; however, it is best to design networks for roaming delays of 50 ms or less. The roaming delay, however, must also take into account the latency that the backend network provides, because the Wi-Fi phone is actually communicating with components residing on the wired network.
Carefully assess RF interference	The presence of RF interference can severely disrupt voice services on wireless LANs. As a result, be certain to assign RF channels to the wireless LAN to stay away from interference sources, such as microwave ovens and cordless phones. It is also important to ensure that adjacent access points are far enough apart to avoid inter-access point interference. In some cases, a company should consider the use of 802.11a for supporting voice services, because the 5-GHz band is relatively free from RF interference.

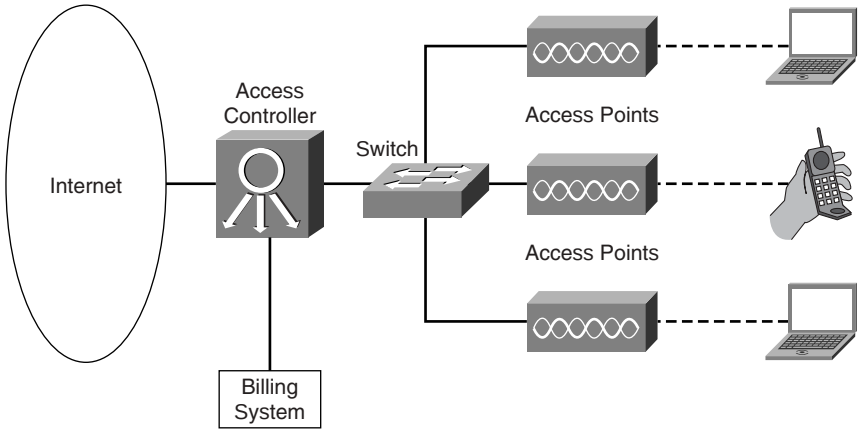
Public Wireless LANs

Many public venues, such as airports, coffee shops, hotels, and convention centers, offer Wi-Fi access to the Internet. This form of network is often referred to as a public wireless LAN (PWLAN). PWLANs allow people who are away from their homes and offices to access e-mail, browse the Internet, and interface with corporate applications. The smaller public "hotspots" may only have a single access point connecting directly to an ISP. Larger implementations usually have numerous access points, similar to an enterprise installation, and incorporate user authentication and billing functions. In some cases, access to the Internet is free. Other hotspots, especially those offering nationwide roaming, charge a fee for usage.

The implementation of a public wireless LAN is often the most complex, as compared to other networks, mainly because of the need to regulate access to the network and to support a wide variety of user interfaces. Figure 23-6 illustrates the configuration of a PWLAN. Access points interconnect to an access controller, which regulates connections to the Internet. A user, for example, initially connects to an access point and attempts to use their browser. The access controller receives the browser's URL request and, instead of returning the requested page, returns an HTML login page. The user enters their credentials, such as username and password, and the controller then opens access to the Internet for that user. If a user is not a subscriber, the controller hands them off to a billing system to collect payment before granting access to the Internet.

Figure 23-6 *Typical Public Wireless LAN Configuration*

KEY POINT



When deploying PWLANs, consider the tips found in Table 23-4.

Table 23-4 *Public Wireless LAN Deployment Tips*

KEY POINT

Tip	Justification Details
Offer an open user interface	A PWLAN needs to serve a diverse population, and the connection experience for users should be as easy as using a mobile phone. The quandary is that PWLAN users do not use a commonly configured device. As a result, incompatible user device configurations are the basis of most problems, such as trouble connecting to the network. Be sure the solution interfaces with the widest possible number of users. This maximizes the number of subscribers that the hotspot provider can attain. Most Wi-Fi users today have 802.11b client devices, but plan ahead for the larger potential proliferation of 802.11g and 802.11a and consider installing access points that support 802.11b/g and 802.11a.
Implement user authentication and billing	For authentication, deploy a controller that regulates access to the protected network services you are providing to users. Whether you should purchase a separate access controller or use a "smart" access point that provides hotspot services depends on your specific requirements. If there are lots of hotspots with only a few users at each one, then it makes sense to use lower-end access points in the facilities and a separate controller at a central point to serve the multiple hotspots. If you have many users at the hotspot, then an access point with built-in access-control features is preferable because it localizes control and improves performance.

continues

Table 23-4 *Public Wireless LAN Deployment Tips (Continued)*

Tip	Justification Details
Disable Layer 2 security	It is not practical to implement Layer 2 security as part of a PWLAN, because it is not feasible or even possible in most cases to manage passwords and provide acceptable levels of interoperability. As a result, disable link-level encryption, such as WEP and WPA, and rely on public users to secure their own connections through the use of VPNs and SSL-based websites and e-mail systems.
Broadcast SSIDs	With PWLANs, you want potential users to find the network. Thus, be certain to enable broadcasting of SSIDs from access points. Also, give the SSID a recognizable name that distinguishes your hotspot from others.
Include DHCP services	As users roam to different hotspots, their client devices need an IP address that corresponds to the local network. To enable roaming with as few end-user actions as necessary, establish DHCP services to automatically assign IP addresses to visiting users. Most versions of Windows operating systems by default activate DHCP, so users probably will not have to do anything.
Focus on increasing capacity	Many hotspots, such as those at airports and convention centers, have lots of users in relatively small areas. The aggregate throughput requirements of these densely populated areas can be very demanding on individual access points, especially those based on the relatively low-bandwidth 802.11b. As a result, pay special attention to properly sizing the PWLAN. To solve this problem, consider placing the access points closer together and lowering their transmit power. This provides higher capacity in a given area by segmenting users.
Enable broadcasting of SSIDs	In PWLANs, you should ensure that beacons sent by the access point broadcast the SSID of the network. This enables Microsoft Windows to identify and display the wireless network to the user. If the SSIDs are not broadcasted, then the user will realize that the network is present, and the user will have to manually configure the SSID in Windows.
Monitor for RF interference	As with any wireless LAN, RF interference can be an issue. So, pay special attention to the possibility of RF interference when deploying PWLANs. The most common interference sources in a public setting are other wireless LANs, mainly because of close proximity of unrelated public hotspots and office buildings with wireless LANs.

Small Office and Home Wireless LANs

It is much easier to install a wireless LAN at home than a wired network. A typical homeowner will not consider running cables throughout the house. It is time-consuming and requires stringing wires through the walls, which can be tricky and frustrating.

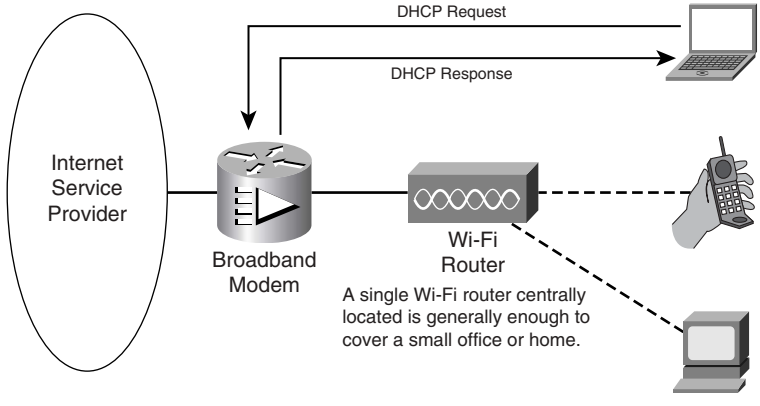
With a wireless LAN, employees can bring laptops home from work and continue working just as they do from their offices. For many professions, this makes it possible for people to work from home more effectively, whether it is to spend a few more hours researching information on the Internet or to enable telecommuting on a daily basis.

Of course, with a wireless laptop, you truly can work from anyplace in the house. There is nothing tying you down to a desk in a particular room. You are free to use the Internet or access files on other computers while relaxing in a comfy chair in front of a TV, lounging on the patio breathing fresh air, or sitting at a desk in a quiet bedroom, just like you see in the commercials.

Wireless LANs at home are good for PCs as well. Unlike companies, Ethernet cabling in homes is nearly nonexistent. That makes wireless the best way to connect stationary PCs to the network. You will have much more flexibility in locating a PC to any part of the house without being near the broadband modem.

Overall, a home wireless LAN is relatively simple (refer to Figure 23-7).

Figure 23-7 *Typical Home Wireless LAN Configuration*



When installing a wireless LAN at home, take into account the following ideas:

- **Purchase a Wi-Fi router**—With most home wireless LAN installations, you need a Wi-Fi router, not just an access point. The router includes Layer 3 functions on top of the access point. For example, Linksys (a Cisco company) specializes in offering Wi-Fi routers for home use. The router provides DHCP and NAT (Layer 3 functions), which are both necessary for allowing multiple Wi-Fi networking devices, such as laptops, PCs, and printers, to share the single broadband (DSL or cable modem) Internet connection. A single 802.11a/b/g Wi-Fi router is generally enough to fully cover most homes. For example, a single dual-band router provides full coverage and respectable performance throughout my entire two-story house with a finished basement.

- **Centralize the router installation**—Install the router within reach of the broadband modem, using Ethernet patch cable. If you have not already established a broadband connection, consider having it installed somewhere central to the areas where you will be using the wireless network. This is typically the center of the home.

  Ideally, install the broadband connection in the same room as any device (such as a printer) that you want to connect to one of the Ethernet ports provided by the router. In a home with two floors, choose installation on the floor where you will be using the network a higher percentage of the time.

- **Configure security mechanisms**—By default, most Wi-Fi routers do not have any security enabled, which means that all data packets are sent unencrypted, in the clear. An unscrupulous person sitting in a car outside your home, for example, can wirelessly monitor these transmissions and see e-mail contents, usernames, and passwords. In addition, unauthorized users can access files on computers inside the home and use the Internet through your broadband ISP connection.

  If you do not want this to happen, then activate encryption supplied within your router. Wired Equivalent Privacy (WEP) is better than nothing, but take advantage of the more advanced WPA or 802.11i if it is available on the router that you have chosen.

Foundation Summary

Unlike most Cisco Press *Exam Certification Guides*, this book does not repeat information covered in the "Foundation Topics" section of the chapter. Please take the time to review the items in the "Foundation Topics" section noted with a Key Point icon.

The Cisco SWAN framework offers a collection of components that comprises a secure, interoperable, and manageable wireless LAN solution. When deploying enterprise wireless solutions, be certain to implement proven security practices. Also, plan on supporting voice services, because many companies are now beginning to take advantage of the applicable gains in efficiencies. Public wireless LANs can be very complex, especially for the larger venues, mainly because of the need for user authentication and billing functions.

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

> client tracking, Cisco SWAN, Cisco WDS, Fast Secure Roaming, public wireless LAN, radio management aggregation, wireless LAN controller, Wireless LAN Threat Defense Solution, WLSE, IP PBX

Refer to the CD-based glossary to check your answers.

Further Reading

For more details regarding wireless LANs, consider reading the following Cisco Press books:

- *Wireless Networks First-Step*, by Jim Geier
- *802.11 Wireless LAN Fundamentals*, by Pejman Roshan and Jonathan Leary

Part IX: OSI and Cisco Device Basics

Chapter 24 Miscellaneous Networking Theory and Practices

Blueprint topics covered in this chapter:

This chapter covers the following topics from the Cisco CCIE Routing and Switching written exam blueprint:

- General Networking Theory
  - — OSI Models
  - — General Routing Concepts
  - — Standards
  - — Protocol Mechanics
  - — Commands

CHAPTER 24

Miscellaneous Networking Theory and Practices

This chapter covers a few basic topics that are not covered elsewhere in the book. In particular, this chapter covers the OSI and TCP/IP architectural models, CLI navigation, and router boot processes.

"Do I Know This Already?" Quiz

Table 24-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions.

Table 24-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section	Score
The OSI and TCP/IP Model	1–3	
Router Operation Miscellany	4–5	
Total Score		

In order to best use this pre-chapter assessment, remember to score yourself strictly. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

1. The TCP software on PC1 uses IP on PC1 to send a TCP segment to PC2. Which of the following are true regarding these actions?
   a. The actions represent an example of adjacent-layer interaction.
   b. The actions represent an example of same-layer interaction.
   c. TCP uses the same features as OSI Layer 3 in this case.
   d. TCP uses the same features as OSI Layer 4 in this case.

2. Which of the following is not a benefit of using a layered networking model?
   a. Eases learning the protocol details.
   b. Eases the development process.
   c. Allows modular engineering.
   d. Results in faster software.

3. Which of the following are names of an OSI layer?
   a. Internetwork
   b. Transmission
   c. Switched
   d. Network Interface
   e. Presentation

4. The answers list **boot system** and **config-register** commands. A router with one or more of the commands listed in the answers would boot ROMMON the next time it is reloaded. Pick the least number of answers for which, when combined, would result in the router booting in ROMMON mode.
   a. **config-register 0x2140**
   b. **config-register 0x2144**
   c. **boot system none**
   d. **boot system rom**

5. Which Cisco IOS Software CLI editing keystrokes move the cursor forward one word?
   a. Ctrl-f
   b. Esc-f
   c. Esc-m
   d. Ctrl-a
   e. Ctrl-e
   f. Esc-e

Foundation Topics

This chapter picks up a few small topics that can be loosely tied to the first section, "General Networking Theory," of the CCIE Routing and Switching written exam blueprint. The first entry in that section is "OSI Models," which implies of course that the CCIE Routing and Switching written exam can ask questions regarding the OSI model. While you could argue that questions about the OSI model are a bit basic for a CCIE exam, the types of questions, and information needed to answer those questions, can be easily scoped. Conversely, that same blueprint section lists several expansive topics, such as "Commands" and "Standards." Almost every chapter in the book covers some standards and commands. Because these general topics from the blueprint are so comprehensive, this chapter does not attempt to cover all the details, but instead relies on the individual chapters.

Beyond the OSI coverage, this chapter includes a few miscellaneous topics in the first section of the blueprint. These topics, "General Routing Concepts" and "Protocol Mechanics," simply do not have a more appropriate place elsewhere in the book.

The OSI and TCP/IP Models

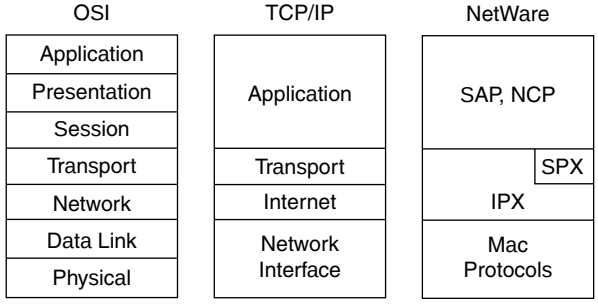
OSI is the Open System Interconnection reference model for communications, developed as a standard by the International Organization for Standardization (ISO). Some participants in OSI's creation and development expected OSI to become *the* networking protocol used by all applications on all computers in the world. The U.S. government went so far as to require OSI support on every computer it purchased, as of a certain date in the early 1990s, which certainly gave vendors some incentive to write OSI code. Some networking vendors even suggested that by the mid-1990s the TCP/IP installed base would begin to decrease, and eventually go away, with OSI as the protocol from which the 21st century Internet would be built.

Today, several of the OSI model's protocols are still used, and ISO is still a very important standards body. However, the OSI model as a whole never gained widespread market acceptance. Today, the OSI model is mainly used as a point of reference for discussing other protocol specifications.

OSI Layers

The OSI reference model consists of seven layers, each defining a set of typical networking functions. Like TCP/IP, the OSI model consists of several standards unique to OSI, as well as references to other well-known standards from other standards bodies. Figure 24-1 shows the OSI layers, as compared with TCP/IP and Novell NetWare.

Figure 24-1 *Comparing OSI, TCP/IP, and NetWare*



Because the OSI reference model does have a well-defined set of functions associated with each of its seven layers, you can examine any other networking protocol or specification, and make some determination as to whether it most closely matches OSI Layer 1, 2, or 3, and so on. For instance, the TCP/IP Internetwork layer (IP) closely equates to the OSI network layer, hence the use of the terms *network layer protocol* and *Layer 3 protocol* when referring to IP.

Although most discussions of networking focus on the functions in the lower four layers of the OSI model, the functions at all seven layers are important in most networks today. The three upper layers of the OSI reference model define functions focused on the application, whereas the lower four layers define functions focused on end-to-end delivery of the data. With the advent of switching devices that look at layers above the transport layer header—devices often called either *Layer 5-7 switches* or *Layer 7 switches*—network engineers need more knowledge and insight into the higher-layer protocols.

Table 24-2 lists the layers of the OSI model and provides some insights into the functions performed by each layer.

Table 24-2 *OSI Reference Model Layer Definitions*

Layer	Functional Description
7	The application layer defines the interface between the communications software and any applications that need to communicate outside the computer on which the application resides. For example, a web browser is an application on a computer. The browser needs to get the contents of a web page; OSI Layer 7 defines the protocols used on behalf of the application to get the web page.
6	The presentation layer's main purpose is to define data formats, such as ASCII text, EBCDIC text, binary, BCD, and JPEG. Encryption is also defined by OSI as a presentation layer service. For example, FTP enables you to choose binary or ASCII transfer. If binary is selected, the sender and receiver do not modify the contents of the file. If ASCII is chosen, the sender translates the text from the sender's character set to a standard ASCII and sends the data. The receiver translates back from the standard ASCII to the character set used on the receiving computer.

Table 24-2 *OSI Reference Model Layer Definitions (Continued)*

Layer	Functional Description
5	The session layer defines how to start, control, and end conversations (called *sessions*). This includes the control and management of multiple bidirectional messages so that the application can be notified if only some of a series of messages are completed. This allows the presentation layer to have a seamless view of an incoming stream of data. The presentation layer can be presented with data if all flows occur in some cases. For example, an automated teller machine transaction in which you withdraw cash from your checking account should not debit your account and then fail before handing you the cash, recording the transaction even though you did not receive money. The session layer creates ways to imply which flows are part of the same session and which flows must complete before any are considered complete.
4	Transport layer protocols provide a large number of services, as discussed in Chapter 5, "IP Services." Whereas Layers 5 to 7 focus on issues related to the application, Layer 4 focuses on issues related to data delivery to the other computer—for instance, error recovery, segmentation of large application data blocks into smaller ones for transmission, and reassembly of those blocks of data on the receiving computer.
3	The network layer defines end-to-end delivery of packets. To accomplish this, the network layer defines logical addressing so that any endpoint can be identified. It also defines how routing works and how routes are learned so that the packets can be delivered. Chapter 4, "IP Addressing," examines Layer 3 concepts in detail. The network layer defines most of the details that a Cisco router considers when routing. For example, Layer 3 running in a Cisco router is responsible for examining the destination IP address of a packet, comparing that address to the IP routing table, fragmenting the packet if the outgoing interface requires smaller packets, and queuing the packet to be sent out to the interface.
2	The data link layer specifications deliver data across one particular link or medium. These protocols are necessarily concerned with the type of media in question; for example, 802.3 and 802.2 define Ethernet for the IEEE, which are referenced by OSI as valid data link layer protocols. Other protocols, such as High-Level Data Link Control (HDLC) for a point-to-point WAN link, deal with the different details of a WAN link.
1	Physical layer specifications, which are also typically standards from other organizations that are referred to by OSI, deal with the physical characteristics of the transmission medium. Connectors, pins, use of pins, electrical currents, encoding, and light modulation are all part of different physical layer specifications. Multiple specifications are sometimes used to complete all details of the physical layer. For example, RJ-45 defines the shape of the connector and the number of wires or pins in the cable. Ethernet and 802.3 define the use of wires or pins 1, 2, 3, and 6. So, to use a Category 5 cable with an RJ-45 connector for an Ethernet connection, Ethernet and RJ-45 physical layer specifications are used.

Table 24-3 lists the OSI reference model layers and examples of some of the more popular protocols that most closely match them.

Table 24-3 *OSI Reference Model—Example Protocols*

Layer Name	Examples
Application (Layer 7)	Telnet, HTTP, FTP, NFS, SMTP gateways (Eudora, Exchange), SNMP
Presentation (Layer 6)	JPEG, ASCII, EBCDIC, TIFF, GIF, PICT, encryption, MPEG, MIDI
Session (Layer 5)	RPC, SQL, NFS, NetBIOS, AppleTalk ASP, DECnet SCP, RADIUS, TACACS+, RTP
Transport (Layer 4)	TCP, UDP, SPX
Network (Layer 3)	IP, IPX, AppleTalk DDP
Data link (Layer 2)	IEEE 802.3/802.2, HDLC, Frame Relay, PPP, FDDI, ATM, IEEE 802.5/802.2
Physical (Layer 1)	EIA/TIA-232, V.35, EIA/TIA-449, RJ45, Ethernet, 802.3, 802.5, B8ZS, T1, E1

The most popular internetworking forwarding devices can also be considered to be operating at a particular layer. The following is a list of devices and the highest layer at which they operate.

- Hub, repeater: Layer 1
- Bridge, switch: Layer 2
- Router: Layer 3
- Multilayer switch: Layer 3
- Content switch: Layers 4–7

OSI Layering Concepts and Benefits

Many benefits can be gained by defining protocol functions in layers, and then defining standard interfaces between those layers. The layers break a large, complex set of concepts and protocols into smaller pieces, making the complex set easier to talk about, easier to implement with hardware and software, and easier to troubleshoot. The following list summarizes the benefits of layered protocol specifications:

- **Easier to learn**—Humans can more easily discuss and learn about the many details of a protocol specification.
- **Easier to develop**—Reduced complexity allows easier program changes and faster product development.

- **Multivendor interoperability**—By creating products to meet the same networking standards, computers and networking gear from multiple vendors can work in the same network.
- **Modular engineering**—One vendor can write software that implements higher layers—for example, a web browser—and another vendor can write software that implements the lower layers—for example, Microsoft's built-in TCP/IP software in its operating systems.

OSI Terminology

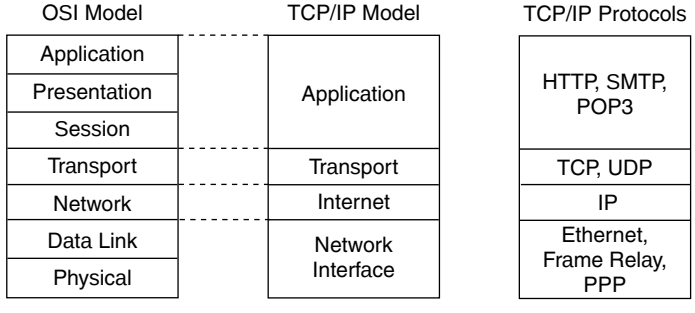
OSI includes many terms not used by other protocol architectures. Many of you have memorized the names of the layers simply through repetition. However, if you find mnemonic devices useful for memorizing such details, the following list shows four 7-word phrases whose first letters correspond to the first letters of the names of the OSI layers:

- All People Seem To Need Data Processing (Layer 7 to 1)
- Please Do Not Take Sausage Pizzas Away (Layer 1 to 7)
- Pew! Dead Ninja Turtles Smell Particularly Awful (Layer 1 to 7)
- Please Do Not Tell Sales People Anything (Layer 1 to 7)

Because the OSI layers' names and numbers are used so pervasively throughout networking terminology, it is important to memorize the numbers and names of each layer. Additionally, it is important to know which layers in a non-OSI protocol architecture best match the OSI layers. Figure 24-2 depicts these details for TCP/IP, along with some of the typical TCP/IP protocols at each layer.

Figure 24-2 *Using OSI Layers to Reference Other Protocols*

KEY POINT

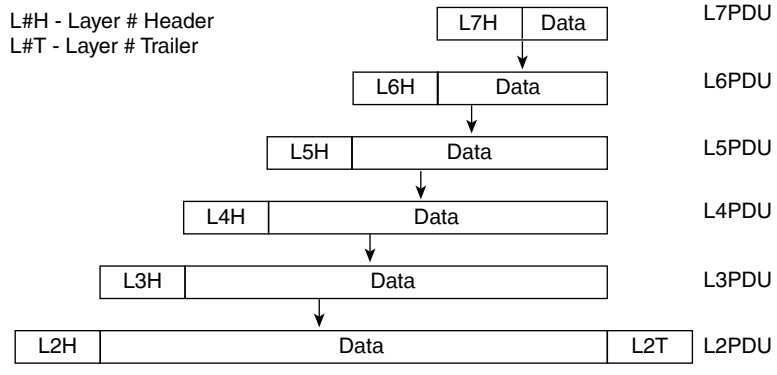


As you can see, not all TCP/IP layers correspond to a single OSI layer, and vice versa. For instance, the TCP/IP *network interface* layer defines both the physical network specifications and the protocols used to control the physical network. (Note that the term *network access* is sometimes used to describe the lowest TCP/IP layer instead of the term *network interface*.) OSI separates the physical network specifications into the physical layer and the control functions into the data link layer. (In fact, many people consider TCP/IP to actually have a separate physical and data link layer.)

The term *protocol data unit (PDU)* is an OSI term that makes generic reference to some data structure used when sending data over a network. A PDU represents the bits that include the headers and trailers for some layer, as well as the encapsulated data. For instance, an IP packet is a PDU. In that case, it is called a *Layer 3 PDU (L3PDU)* by virtue of IP being a Layer 3 protocol.

Figure 24-3 represents the encapsulation process used by OSI on a computer when sending data. The figure points out the names of the PDUs at each layer.

Figure 24-3 *OSI Encapsulation and Protocol Data Units*



OSI Layer Interactions

Every layered networking architecture, like OSI or TCP/IP, is implemented by a series of protocols. Part of the definition of a protocol defines how that protocol works between multiple devices. Other parts of the protocol define that protocol's interface to another protocol running on the same computer. For example, consider the HTTP protocol. Figure 24-4 shows the simple interaction of HTTP on two computers (Bob and Larry), with Bob retrieving a web page from web server Larry.

Figure 24-4 *Basic Application Logic to Get a Web Page*

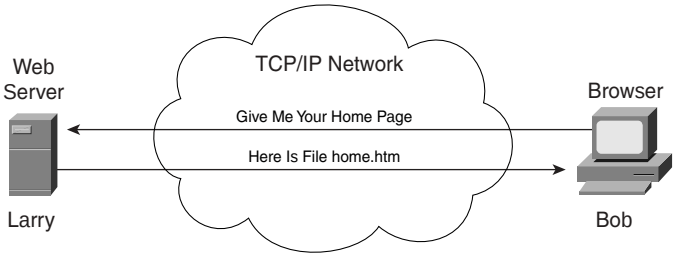
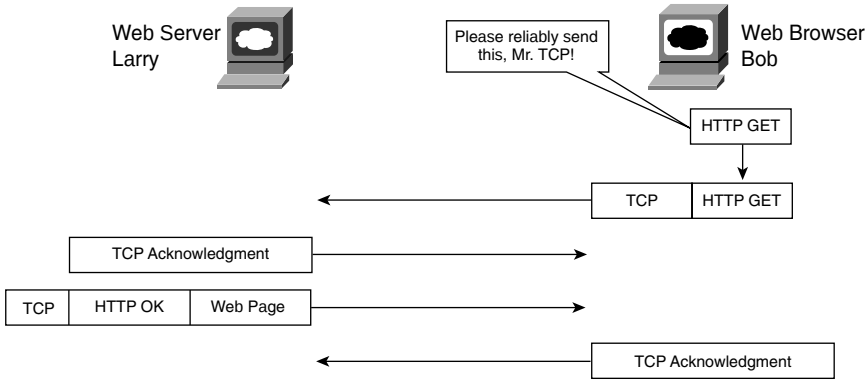


Figure 24-4 shows an example of same-layer interaction. The HTTP protocol is a TCP/IP application layer protocol. The HTTP software on each computer in the figure sits at the same

layer, and the computers interact by exchanging messages and data—hence the term *same-layer interaction*.

Next, consider how adjacent layers on the same computer work together, with each layer providing a service to the layer above it. For example, in Figure 24-4, Bob and Larry use HTTP to transfer the home page from Larry to Bob. But what would happen if Bob's HTTP Get request was lost in transit through the TCP/IP network? Or what would happen if Larry's response, which includes the contents of the home page, was lost? Well, the page would not show up in Bob's browser, as you might expect. So, HTTP uses a protocol from its lower adjacent layer (TCP at the transport layer) to provide a service of guaranteed delivery across the network, as shown in Figure 24-5.

Figure 24-5 *Example of Adjacent-Layer Interaction: TCP and HTTP*



This example outlines the concepts of how protocols in adjacent layers, and on the same computer, work together. In this case, the higher-layer protocol (HTTP) needed to do something it could not do (error recovery), so it asked the next-lower-layer protocol (TCP) for help. TCP provided a guaranteed delivery service to HTTP. The interaction between these adjacent layers is aptly called *adjacent-layer interaction*.

Table 24-4 summarizes the key points of same-layer interaction and adjacent-layer interaction.

Table 24-4 *Summary: Same-Layer and Adjacent-Layer Interactions*

KEY POINT

Concept	Description
Same-layer interaction on different computers	The two computers use a protocol with which to communicate with the same layer on another computer. The protocol defined by each layer uses a header that is transmitted between the computers to communicate what each computer wants to do.
Adjacent-layer interaction on the same computer	On a single computer, one layer provides a service to a higher layer. The software or hardware that implements the higher-layer requests that the next lower layer perform the needed function.

Router Operation Miscellany

The second part of this chapter covers a few minor miscellaneous topics, all related to some degree to basic operations of a Cisco router.

Cisco IOS Software Boot Sequences and the Configuration Register

Cisco routers use the following basic process at boot time:

1. The router performs a *power-on self-test (POST)* to discover and verify the hardware.
2. The router loads and runs *bootstrap* code from ROM.
3. The router finds the Cisco IOS Software or other software and loads it.
4. The router finds the configuration file and loads it into the running configuration.

Step 3 contains three major options for the type of software loaded into the router, as listed in Table 24-5.

Table 24-5 *Three OS Categories for Routers*

KEY POINT

Operating System	Typically Stored In	Purpose
Full-featured IOS	Flash	Full-featured normal IOS image used in production
Limited-function IOS (RXBOOT mode)	ROM	Basic IP connectivity, used when Flash memory is broken and you need IP connectivity to copy a new IOS image into Flash memory
ROMMON	ROM	Low-level debugging, usually by the Cisco TAC, and password recovery

The router must be configured to point to the software that should be loaded, and if an IOS image should be loaded, the configuration must define which file to load. Multiple IOS files can be stored in Flash, and a router can load IOS images over a network. Two configuration commands impact a router's choice of what software to load:

- The **config-register** command
- The **boot system** command

The Configuration Register

The *configuration register* is a 16-bit software register in the router, and its value is set using the **config-register** global configuration command. (Some older routers had a hardware configuration register with jumpers on the processor card, to set bits to a value of 0 or 1.) On most Cisco routers,

the default configuration register setting is hexadecimal 2102 (0x2102). Figure 24-6 shows an example binary breakdown of the default value for the configuration register.

Figure 24-6 *Binary Version of Configuration Register, Value Hex 2102*

Bit Position, in Decimal	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default Binary Value	0	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0

The term *boot field* refers to the low-order 4 bits of the configuration register, because only those 4 bits impact the router's choice of what software to load. The boot field is often considered as a separate 4-bit value, represented as a single hexadecimal digit. (Cisco represents hexadecimal values by preceding the hex digit[s] with 0x—for example, 0xA would mean a single hex digit A.)

When deciding which software to load, a router first looks at the configuration register's boot field, using the following logic to determine which to use:

KEY POINT

1. If the boot field is hex 0, the router loads ROMMON.
2. If the boot field is hex 1, the router uses RXBOOT mode.
3. For anything else, the router loads an OS based on the configured **boot system** commands.

The boot system Command

When the boot field is not a hex 0 or 1, routers choose the OS through the use of the **boot system** configuration command. If the configuration register calls for a full-featured IOS (boot field 0x2-F), the router reads the startup-configuration file for **boot system** commands. If present, the router tries each **boot system** command, in succession, until it finds an IOS to load. If there are no **boot system** commands, the router takes the default action, which is to load the first file in Flash memory.

Table 24-6 provides some example **boot system** commands and the logic used when the configuration register's boot field is set to 0x2.

Table 24-6 *Impact of the* **boot system** *Command on Choice of IOS; Boot Field Between 2 and F*

Boot System Commands	Result
No **boot** command	Router tries loading the following, in order: first file in Flash; broadcasts looking for a TFTP server and a default filename; IOS in ROM; or ROM Monitor.
boot system ROM	Router loads IOS from ROM.
boot system flash	Router loads the first file from Flash memory.
boot system flash *filename*	Router loads IOS with the name *filename* from Flash memory.

continues

Table 24-6 *Impact of the* **boot system** *Command on Choice of IOS; Boot Field Between 2 and F (Continued)*

Boot System Commands	Result
boot system tftp *filename* **10.1.1.1**	Router loads IOS with the name *filename* from the TFTP server 10.1.1.1.
Multiple **boot system** commands, any variety	Router attempts to load IOS based on the first **boot** command in configuration. If that fails, the second **boot** command is used, and so on, until one is successful.

CLI Help Features

The Cisco IOS CLI includes several methods to get help at the CLI. Many of you already use these features out of habit, but for completeness, the full set of features is listed in Table 24-7.

Table 24-7 *Cisco IOS Software Command Help*

What You Type[1]	The Help You Get
?[2]	Help for all commands available in this mode.
help	Text describing how to get help. No actual command help is given.
*command***?**	Text help describing all the first parameter options for the command.
*com***?**	A list of commands that start with *com*.
*command parm***?**[2]	A list of all parameters beginning with *parm*. (Notice that no spaces exist between *parm* and the ?.)
command parm<Tab>	If you press the Tab key midword, the CLI will either spell the rest of this parameter at the command line or do nothing. If the CLI does nothing, it means that this string of characters represents more than one possible next parameter, so the CLI does not know which to spell out.
*command parm*1 **?**	If a space is inserted before the question mark, the CLI lists all the next parameters and gives a brief explanation of each.

[1] *command* represents any command, not the word command. Likewise, *parm* represents a command's parameter, not the word parameter.

[2] When you type the **?**, the Cisco IOS CLI reacts immediately; that is, you don't need to press the Enter key or any other keys. The router also redisplays what you typed before the **?**, to save you some keystrokes. If you press Enter immediately after the **?**, IOS tries to execute the command with only the parameters that you have typed so far.

Cisco IOS stores the commands that you type in a history buffer, storing ten commands by default. You can change the history size with the **terminal history size** *x* user exec command, where *x* is the number of commands for the CLI to recall; this can be set to a value between 0 and 256. You

can then retrieve commands so that you do not have to retype the commands. Table 24-8 lists the commands used to manipulate retrieved commands.

Table 24-8 *Key Sequences for Command Edit and Recall*

Keyboard Command	What the Router Displays
Up arrow or Ctrl-p	Displays the most recently used command. If it is pressed again, the next most recent command appears until the history buffer is exhausted. (The p stands for previous.)
Down arrow or Ctrl-n	If you have gone too far back into the history buffer, these keys will go forward, in order, to the more recently typed commands. (The n is for next.)
Left arrow or Ctrl-b	Moves the cursor backward in the currently displayed command without deleting characters. (The b stands for back.)
Right arrow or Ctrl-f	Moves the cursor forward in the currently displayed command without deleting characters. (The f stands for forward.)
Backspace	Moves the cursor backward in the currently displayed command, deleting characters.
Ctrl-a	Moves the cursor directly to the first character of the currently displayed command.
Ctrl-e	Moves the cursor directly to the end of the currently displayed command.
Esc-b	Moves the cursor backward one word in the currently displayed command.
Esc-f	Moves the cursor forward one word in the currently displayed command.
Ctrl-r	Creates a new command prompt, followed by all the characters typed since the last command prompt was written. This is particularly useful if system messages have made the screen confusing and it is unclear what you have typed so far.

The key sequences in Table 24-8 are part of what Cisco calls *enhanced editing* mode. Cisco IOS enables enhanced editing mode by default, and has for a long time. However, you can turn off these keystrokes with the **no terminal editing** exec command, and turn them back on with the **terminal editing** command. Why would you bother? On occasion, you might be using a scripting language to automatically run commands on the router through a Telnet session, and enhanced editing mode can sometimes interfere with the scripts.

Foundation Summary

This section lists additional details and facts to round out the coverage of the topics in this chapter. Unlike most Cisco Press *Exam Certification Guides*, this book does not repeat information presented inside the "Foundation Topics" section of the chapter. Please take the time to read and study the details in this section of the chapter, as well as review the items in the "Foundation Topics" section noted with a Key Point icon.

Table 24-9 lists some of the most popular router IOS commands related to the topics in this chapter.

Table 24-9 *Router IOS Commands Related to Chapter 24*

Command	Description
terminal editing	Exec command that enables advanced editing keystrokes (default)
terminal no editing	Exec command that disables advanced editing keystrokes
terminal history	Exec command that enables the feature of remembering a number of previously typed commands (default)
terminal no history	Exec command that disables the feature of remembering a number of previously typed commands
terminal history size *number-of-lines*	Exec command that sets the size of this user's terminal history buffer
config-register *value*	Global config command that sets the value of the router's configuration register
boot system flash [*flash-fs:*][*partition-number:*][*filename*]	Global config command that tells the router the location in Flash of a candidate IOS file
boot system rom	Global config command that tells the router to load the ROMMON OS
boot system *file-url*	Global config command that tells the router a URL location of a candidate IOS file
boot system {**rcp** | **tftp** | **ftp**} *filename* [*ip-address*]	Global config command that tells the router the location of a candidate IOS file that is accessible via a file transfer protocol

Memory Builders

The CCIE Routing and Switching written exam, like all Cisco CCIE written exams, covers a fairly broad set of topics. This section provides some basic tools to help you exercise your memory about some of the broader topics covered in this chapter.

Fill in Key Tables from Memory

First, take the time to print Appendix F, "Key Tables for CCIE Study," which contains empty sets of some of the key summary tables from the "Foundation Topics" section of this chapter. Then, simply fill in the tables from memory, checking your answers when you review the "Foundation Topics" section tables that have a Key Point icon beside them. The PDFs can be found on the CD in the back of the book, or at http://www.ciscopress.com/title/1587201410.

Definitions

Next, take a few moments to write down the definitions for the following terms:

boot field, configuration register, terminal history, enhanced editing, adjacent-layer interaction, same-layer interaction, PDU, L*x*PDU, ROMMON, RXBOOT

Refer to the CD-based glossary to check your answers.

Part X: Appendixes

Appendix A Answers to the “Do I Know This Already?” Quizzes

Appendix B CCIE Routing and Switching Exam Updates: Version 1.0

Appendix C MPLS

Appendix D Decimal to Binary Conversion Table

APPENDIX A

Answers to the "Do I Know This Already?" Quizzes

Chapter 1

1. c, e
2. a
3. c. If a Cisco switch port has only **speed** or **duplex** configured, the interface still uses auto-negotiation Fast Link Pulses (FLP) to negotiate the setting that was not configured. There is no explicit command to disable auto-negotiation.
4. b, c. Cisco switches disable auto-negotiation after the speed and duplex have been configured. The other switch attempts auto-negotiation and fails. However, the unconfigured switch can detect the speed even without auto-negotiation. By default, 10-Mbps and 100-Mbps ports use half duplex when they are unable to auto-negotiate a duplex setting.
5. c, d. The half-duplex switch will leave its loopback circuitry enabled, erroneously detecting a collision when it is both sending and receiving a frame.
6. b, c
7. b
8. a, b, d

Chapter 2

1. a. The **switchport access vlan 28** command, entered in interface configuration mode, also creates the VLAN. The **vlan 28 name fred** command is valid in VLAN database mode, but not in configuration mode.
2. a, b. Of the three incorrect answers, one refers to VTP pruning, and another refers to VTP's ability to make VLAN configuration more consistent through the advertisement of VLAN configuration. Private VLANs do have a positive effect on the reduction of broadcasts, but that is not a primary motivation for using private VLANs.
3. c. VTP works only for normal-range VLANs. Reserved VLAN numbers 1 and 1002–1005 cannot be pruned.

4. a. Because the new switch is a VTP client, it will consider using the VTP updates it receives from the two pre-existing VTP servers. However, because the new switch's configuration revision number is higher, the new switch ignores the VTP updates with revision number 201.

5. b. The new switch will send VTP updates with revision number 301, and the two original VTP servers will send updates with revision number 201. Because the new switch's revision number is higher, the older ten switches will update their configuration.

6. a. VLAN 1 is the native VLAN by default. The switch ports to which the PCs connect also sit in VLAN 1 by default. 802.1Q does not add any header when passing frames in the native VLAN.

7. a, b. The commands mentioned in the question have statically configured one switch to use 802.1Q trunking with no DTP autonegotiation. Due to the **nonegotiate** option, the other switch must be statically configured to trunk, using **dot1q**, which requires the two commands listed as correct answers. The **nonegotiate** option is not required on the second switch, because both switches statically agree to the same trunking settings.

8. a, b, c. The subinterface number does not identify the VLAN, so either fa 0/1.1 or fa 0/1.2 could be used with the native VLAN. Also, native VLAN IP addresses can be configured under the physical interface as well.

9. a, c, d. 802.1Q inserts a 4-byte tag, but it does not encapsulate the original frame. VTP Version 2, in and of itself, is restricted to normal-range VLANs. Finally, DTP chooses ISL over 802.1Q if both are enabled.

Chapter 3

1. b. The root switch includes the maxage timer in its advertised Hellos, with non-root switches using the timer value advertised by the switch. Maxage must expire before a non-root switch believes that connectivity to the root has been lost, thereby triggering reconvergence, and the possible election of a new Root Port.

2. c. The root switch includes the Forward Delay timer in its advertised Hellos. When receiving a Hello with TCN set, the non-root switch uses an aggressive time-out value for CAM entries based on the advertised forward delay timer.

3. a, c. MST uses RSTP; RSTP waits three times the Hello time, as advertised in the Hello BPDU, before deciding to act. The timer settings on non-root switches, as usual, do not impact the process.

4. b, c. Multiple STP instances are certainly supported. The answer relating to multiple PVST+ domains is partially true. However, the mechanics involved use multicasts for each STP instance, instead of encapsulating in the native VLAN's Hellos, making that answer technically incorrect.

5. c. When STP converges, MAC address table entries need to be timed out quickly, because their associated interfaces may no longer be valid. The Topology Change Notification (TCN) BPDU from the non-root switch causes the root to react, marking a TCN bit in future Hellos, thereby making all switches time out their CAMs based on the Forward Delay timer.

6. d. All the links on one switch will become the Designated Port on their respective segments, making four ports forward. Only one of the ports on the other switch will be a Root Port, making the total 5.

7. a. The links must all be in the same operational trunking state (trunking or not). If trunking, they must be using the same type of trunking. However, the settings related to trunk negotiation do not have to match.

8. c, d, f. IEEE 802.1D (STP) using states of forwarding, blocking, listening, learning, and disabled. IEEE 802.1w RSTP uses states of forwarding, learning, and discarding.

9. d, e. When grading yourself, if you did not pick the answer IEEE 802.1w, give yourself credit if you knew RSTP technically does not use Maxage, but instead waits three times the Hello interval instead of using Maxage.

10. b. UniDirectional Link Detection (UDLD) uses Layer 2 messaging to determine when it can no longer hear from the neighbor. These messages allow a switch to recognize a unidirectional link and react by error-disabling at least one end of the link. Loop Guard does not use messages, but rather it changes how a switch reacts to the loss of incoming Hellos. When they are no longer received, the port is placed into an STP loop-inconsistent state.

Chapter 4

1. d

2. c. Summary 10.1.1.0/21 would include addresses 10.1.0.0–10.1.7.255. Summary 10.1.0.0/22 would include 10.1.0.0–10.1.3.255, only including three of the four subnets listed in the question.

3. d. 10.22.12.0/22 includes addresses 10.22.12.0–10.22.15.255, which covers exactly the same range of addresses as subnets 10.22.12.0/23 and 10.22.14.0/23 listed in the problem statement. Similarly, summary 10.22.16.0/22 covers exactly the set of IP addresses inside the other two subnets in the problem statement. As for the wrong answers, 10.22.12.0/21 is not actually a valid summary; 10.22.8.0/21 and 10.22.16.0/21 are. Summary 10.22.8.0/21 includes address 10.22.8.0–10.22.15.255, which includes address ranges not covered by subnets in the problem statement. Similarly, summary 10.22.16.0/21 includes IP addresses outside the listed subnets.

4. a, d. 10.22.21.128/26 implies a range of 10.22.21.128–10.22.21.191; 10.22.20.0/23 implies a range of 10.22.20.0–10.22.21.255.

5. a. By definition, CIDR allows Internet routes to group large blocks of IP addresses, based largely on the assignment of IP network numbers to particular ISPs or to ISPs in particular worldwide geographic locations.

6. c, d. Port Address Translation (PAT) and dynamic NAT with overloading both refer to the same feature, in which each TCP or UDP flow is mapped to a small number of IP addresses by using different port numbers. The other terms do not refer to features that reduce the number of IP addresses used by NAT.

7. b. The four terms in the answers have two pair of contrasting words. The word "inside" implies a host inside the enterprise that is using NAT, whereas "outside" refers to a host outside the enterprise. The word "local" refers to an IP address used for packets as they flow through the enterprise (where local private addressing can be used), and "global" refers to an IP address used in packets as they flow over the Internet (which requires globally unique IP addresses). The question refers to a packet's destination address, with the packet going to a host inside the enterprise—hence the term "inside" is correct. The packet is on the Internet, per the question, so the term "global" also applies.

8. a, b. Typically, a NAT overload configuration using a single public IP address would use the style that refers to the interface in the command. However, a NAT pool with a single IP address in the pool works the same and is valid.

9. a, b, e. Leading 0s in a quartet can be omitted, and one instance of :: can be used to represent consecutive quartets of 0000.

Chapter 5

1. e. The ICMP Unreachable message has several subtypes, with the Host Unreachable message type meaning that the subnet is reachable but that specific host cannot be reached.

2. c. When routers discard a packet because the packet's TTL field was decremented to 0, the routers send an ICMP Time Exceeded message back to the originator of the discarded packet.

3. e. When a host receives a packet, and the destination IP address of the packet is that host's IP address, the host needs to next perform transport layer processing. Then, if the packet contains a TCP or UDP segment, the host needs to pass the data to the application listening on the destination port of the packet. If that host does not have an application listening on that port, the host discards the packet and returns an ICMP Unreachable message, with a subtype of Port Unreachable, to the sender of the packet.

4. c, d. LAN-attached hosts use Address Resolution Protocol (ARP) when they need to find the MAC address of another host, when the host thinks that the other host is on the same subnet. In order for R1 to have been performing a proxy ARP reply for PC2's ARP request, the request must have been for a host on a different subnet—most likely, an ARP looking for the web

server's MAC address. In order for PC2 to make such an ARP request, PC2 must have believed that the web server was on the same subnet; if PC2's mask was 255.255.0.0, PC2 would have indeed thought that the web server was in the same subnet.

5. a, e. When the router receives the DHCP request, it changes the destination IP address of the packet to the value set with the **ip helper-address** command. Because the PC does not yet have an IP address, the DHCP request (as sent by PC3) has an IP address of 0.0.0.0. The router then changes the source IP address so that the DHCP response packet can be routed back to the original subnet, and then broadcast back onto that subnet. To make that happen, the router changes the source IP address of the DHCP request to be the subnet broadcast address for that subnet, namely 10.4.7.255.

6. b, d. RARP and BOOTP require a static reservation of an IP address for each specific MAC address. Because BOOTP encapsulates its messages inside an IP packet, the packets can be routed to a BOOTP server; RARP does not use an IP header, so its messages cannot be routed. Also, RARP only supports the assignment of the IP address, whereas BOOTP allows the assignment of other settings, such as the mask and default gateway.

7. d. With default settings on R2, preemption would not be allowed. Therefore, even in cases for which R2 would have a better (higher) HSRP priority, R2 would not take over from R1 until R2 believed that R1 had failed.

8. b, d. Routers using NTP server mode do not rely on outside devices for clock synchronization, so they do not need to know another NTP server's IP address. Routers in NTP broadcast client mode expect to receive NTP updates via LAN broadcasts, so they do not need to know an IP address of an NTP server to which to send NTP queries.

Chapter 6

1. a, d. Both protocols use a source and destination port number, as well as a header checksum. Only TCP performs error recovery and flow control.

2. a, f. URG and PSH are used to signal the receiver, during the middle of a connection, to pass data to the receiving socket application. SYN and ACK are used during connection establishment, and FIN and RST are used at connection termination.

3. a. TCP senders use the lower window setting of the two possibilities: the advertised window, as implied by the receiver in the Window field of the TCP segment, and the sender's calculated congestion window. The choice of using the lower of the two does not change as a result of packet loss. The URG code bit does not relate to error recovery, and is only set in retransmitted segments by coincidence. Finally, the Acknowledgement field identifies the next byte expected by the receiver, which is 1 greater than the last byte acknowledged by that receiver.

4. a, e

5. a, c. In active mode, the client allocates a dynamic port, and listens on that port. The client tells the server, via an FTP PORT command, the port number. The server then uses its well-known port 20, and initiates a TCP connection to the client's new listening port. In passive mode, the server allocates a dynamic port, which does not have to be port 20 (and is typically not port 20), and listens on that port. The server tells the client, via an FTP PORT command, the port number. The client then allocates a currently unused dynamic port, and initiates a TCP connection to the server's new listening port.

6. a, c. SNMPv1 and SNMPv2c call for the use of community strings for basic authentication, with those strings being passed as clear text. SNMPv3 allows for authentication using private keys and MD5 digests, and encryption, but the two security features do not have to be used concurrently.

7. c, e. Most of the details of the answers in this question are summarized in Tables 6-6 and 6-7. Briefly, SNMPv1 used communities, and several protocol messages (Get, GetNext, and Trap included). SNMPv2 dropped communities; however, RFC 1905, popularly called SNMPv2c, added SNMPv1 communities back to SNMPv2 as an option. SNMPv2 added the GetBulk and Inform protocol messages as well. SNMPv3 simply kept most of SNMPv2, adding authentication and encryption mechanisms.

Chapter 7

1. c

2. c, d, e. The CEF FIB is populated based on the contents of the IP routing table. The two incorrect answers refer to events that do not change the contents of the IP routing table; rather, they change adjacency information. The three correct answers cause a change to the IP routing table, which in turn changes the CEF FIB.

3. d. Routers send InARP messages in response to learning that a new PVC is up. Routers learn that a PVC is up when they receive LMI messages stating that a previously inactive DLCI is now Active. While an InARP from another router may be received around the same time, it is the LMI notification of the now-active DLCI that drives the process.

4. a, c, d. InARP is enabled by default on all three types of interfaces stated in the question. As a result, with the full mesh, all three routers will have the necessary mapping between the IP address and DLCI, and pinging will work between each pair of routers.

5. a, c, e. InARP is enabled on all three types of interfaces by default. However, InARPs only flow across a PVC, and are not forwarded—so R2 and R3 have no way to inform each other of the correct mapping information. Additionally, R3's point-to-point subinterface usage actually changes R3's logic, whereby R3 does not rely on the received InARP information. Rather, R3 uses logic like "when forwarding to any address in 10.1.1.0/24, send it over the one PVC on that subinterface." R2, with a multipoint subinterface, does not use such logic, and simply lacks the correct mapping information. As a result, when R2 pings R3, R2 does

not know what DLCI to use, and cannot send the Echo. R3, however, does what DLCI to use, so when R3 pings R2, R3 actually sends the packet to R1, which then forwards it to R2. R2 can't return the Echo Reply, so the ping fails.

6. c. The **ip classless** global configuration command tells the router to use classless IP forwarding/routing. This means that if a packet's destination address does not match any specific subnet in the routing table, and a default route exists, the router will use the default route.

7. e. Of the incorrect answers, only the **ip address** command is a valid command; that command sets the IP address associated with the interface.

8. e

9. c. The **set interface default** command tells policy routing to use the listed interface as if it were a default route, using it only if the IP routing table is not matched.

10. a, c. The **default** option tells policy routing to first try to match the routing table, and then to use the directions in the route map if no route exists in the routing table.

Chapter 8

1. a, c. RIPv2 added VLSM support to RIPv1 by including the subnet mask with each route. RIP does not send Hellos. It defines infinity as 16 hops, and uses either clear-text passwords or MD5 for authentication.

2. a. RIP sends full updates every 30 seconds, with those updates including all routes from the routing table, except for any routes omitted due to split horizon rules. The router actually adds 1 to the metrics shown in its routing table to the routes included in a routing update.

3. b, c. R1's metric 16 route advertisement was a poisoned route. R1 would suspend split horizon rules for that route upon receipt of the metric 16 route, sending back a poison reverse route, metric 16. R1 would have sent back the metric 16 route whether split horizon was enabled or disabled on s0/0. Also, if the last received metric was 3, and then 16, the failed route would not have been caused by a counting-to-infinity problem.

4. a, b. The Invalid timer is set per route, counting up from 0, and reset to 0 each time the same route is received in an update coming in the same interface as before. The timer is kept by a router, and is not advertised. The **debug** commands show information about advertised and received updates, but because the Invalid timer is not transmitted in the network, these **debug** commands do not display the timer.

5. f. The **clear ip route** command is not complete, and would be rejected by Cisco IOS. To delete all routes, the **clear ip route *** command would be used.

6. b, d. RIPv2 supports the advertisement of a next-hop IP address that is different from the sender's IP address. The **show ip route** and **show ip rip database** commands list the advertised next hop only, and not the IP address of the sender of the update.

7. a, b, e. The **network** command tells RIP to do three things on each interface in that classful network: Advertise the connected subnet, send updates, and receive updates. RIP does not have a Hello message. The **passive-interface** command does make RIP stop sending updates on an interface, but the command is a **router rip** subcommand, not an interface subcommand.

8. d. Cisco IOS disables split horizon by default on physical interfaces configured for Frame Relay, but it is enabled by default on Frame Relay multipoint interfaces. The default RIP authentication mode is simple text. RIP sends triggered updates when a route changes, and this feature cannot be disabled.

Chapter 9

1. c, d. EIGRP uses IP protocol 88, with no transport header following the IP header. It supports MD5 authentication, but not simple text. It first sends update messages to 224.0.0.10, then sends them as unicasts if RTP requires retransmission.

2. a, c, d. R2's K values differ from the others, so it will not become a neighbor. The Hello timer and Hold timer differences do not prevent EIGRP neighbor relationships from forming. Also, the different masks do not prevent neighborship, as long as each router believes that its neighbors are in the same primary subnet.

3. c

4. d. EIGRP Updates can be either full or partial. While Reply messages do include routing information, they often trigger the receiving router to use partial update messages to inform other neighbors about the change in a route.

5. b. The **show ip eigrp topology** command lists successor and feasible successor routes, omitting routes that are neither. Because two of the routes are successors, only one of the three is a feasible successor.

6. b. R11's route through 10.1.11.2 is considered invalid, because the neighbor at that IP address failed. The topology table holds one feasible successor route to the subnet, so it can be used immediately, without requiring active Querying of the route.

7. b. An EIGRP router must wait on all Reply messages to be received before acting on the new information. The Active timer dictates how long the router should wait for all the Reply messages to come back.

8. c. The **network** command wildcard mask matches any interface with an address that starts with 10.1, with 0, 1, 2, or 3 in the third octet, and anything in the fourth octet. So, it will not advertise 10.1.4.0/24 or 10.1.5.0/24, because those interfaces are not matched with a **network** command. Because it is passive, no Hellos are sent out fa0/0, meaning no neighbor relationships are formed, and no routes exchanged either in or out fa0/0.

9. a, d. The **passive-interface** command, by definition with EIGRP, tells EIGRP to not send EIGRP Hellos out the interface. A receive-only EIGRP stub router only receives routing updates, but to do so, it must form neighbor relationships with any routers, so it does send

Hellos. Of course, if no **network** command matches an interface, EIGRP is not enabled for the interface at all.

Chapter 10

1. b, c. OSPF uses IP protocol 89, and does not use TCP. LSUs can be acknowledged by simply repeating the LSU or by using the LSAck packet.

2. a, c. Multipoint interfaces default to use network type nonbroadcast, so the **ip ospf network non-broadcast** command would not show up in the configuration. This type defaults to 30-second Hello and 120-second Dead timers. **Neighbor** commands are required, but only one of the neighbors on either end of a PVC needs to configure the **neighbor** command. Network type nonbroadcast does require a DR. So for all routers to be able to communicate with the DR, router R-core needs to be the DR.

3. c. The **ip ospf network non-broadcast** command, which is also the default on multipoint interfaces, requires a DR, as well as requiring **neighbor** commands.

4. d. Unlike EIGRP, OSPF's hello intervals on LANs must match, so R2 will not form any neighbor relationships. R1, R3, and R4 will not expect Hellos from R2, because it will not be a neighbor. R2 will not become a DR, because a router must form a neighbor relationship before performing DR election. R3 ties on highest priority amongst R1, R3, and R4, and it beats R1 with a higher RID, so it becomes DR, not BDR. R4's dead interval will default to four times hello, or 40 seconds.

5. a. They must be in the same area, and must have the exact same stubby area type. The LSRefresh setting is not checked during the Hello process. Finally, the hello and dead intervals must match before two routers will become neighbors.

6. c, d. Routers in NSSAs inject type 7 LSAs; R1, being in the backbone area, cannot be in an NSSA, so it will inject a type 5 LSA for 200.1.1.0/24. R3 will indeed learn a route to 200.1.1.0/24, as area 0 is not a stub area. R2, an ABR, will forward the type 5 LSA created by R1 into area 1. Finally, every DR creates a type 2 LSA for the subnet and floods it throughout the area.

7. a, b, d. Routers in NSSA areas can inject type 7 LSAs into the NSSA area when redistributing into OSPF from external sources. However, because the area is totally NSSA, R2 will not forward type 5 or type 3 LSAs into totally NSSA area 1. Instead, R2 will inject a default route via a type 3 LSA into area 1. Finally, every DR creates a type 2 LSA for the subnet and floods it throughout the area.

8. b. The command configures area 55 as a totally stubby area, which means that no external type 5 (E1 or E2) LSA can be sent into the area by the ABR. Also, the ABR does not create type 3 summary LSAs for the subnets in other areas. The ABR does create and inject a default route into the area due to the **no-summary** option.

9. c. OSPF E1 routes include internal OSPF costs in the metric calculation; for E2 routes, the internal OSPF cost is not considered. R3's cost to reach 10.1.1.0/24 includes R3's cost to reach the ABR plus the cost stated in the type 3 LSA. When route summarization occurs, the summary uses the least cost of all the constituents subnets; however, this design does not use any route summarization.

10. a, e. The **network** command's mask works like an ACL wildcard, so the **network 10.0.0.0 0.0.0.255** command matches addresses beginning 10.0.0—so it does not match the LAN interface on R2. OSPF routers can use different process IDs and still become neighbors. OSPF costs can be asymmetric, meaning that routers can become neighbors without having the same OSPF costs. The cost value is not part of the DR election decision. Finally, with a reference bandwidth of 1000, R1 calculates the cost as 1000/100 = 10.

11. c, f. The **ip ospf dead-interval minimal** command sets the dead interval to 1 second, and the hello interval to (1/multiplier) seconds. The hello interval defaults to 10 seconds on some network types, notably point-to-point and broadcast networks, and defaults to 30 seconds on other network types. The **ip ospf hello-multiplier** command is not a valid command.

12. a. The **ip ospf authentication** interface subcommand takes precedence over the **area 0 authentication message-digest** command, causing R1 to attempt OSPF type 1 (simple text) authentication, with the key being configured with the **ip ospf authentication-key** command.

Chapter 11

1. a, d. A **route-map** clause acts on items that match the parameters on the **match** command. For routes that do not match a clause's **match** command, the route map moves on to the logic in the next **route-map** clause. Route 10.1.1.0/24 was matched, and because the **route-map** clause had a **permit** action, the route was redistributed. 10.1.2.0/24 was not matched by the first **route-map** clause, so it would fall through to be considered in the next **route-map** clause. However, the question did not supply the rest of the information, so you cannot tell whether 10.1.2.0/24 was redistributed or not.

2. c, d. The 10.128.0.0/9 defines the matching parameters on the prefix (subnet number), which matches subnets beginning with 10, and with 128–255 in the second octet. The **ge 20** implies that the routes must have a prefix length between 20 and 32, inclusive. Only 10.200.200.192 and 10.128.0.0 match both criteria.

3. a. EIGRP requires that a metric be defined for any route redistributed into EIGRP, and no metrics have been defined. So, the **redistribute ospf 2** command does not cause any routes to be redistributed. OSPF defaults to use metric 20, with redistributed routes as type E2.

4. a, b. The **redistribute eigrp 1 subnets** command looks for EIGRP routes, and connected routes that match any EIGRP **network** commands.

5. c. Because EIGRP treats external routes as AD 170 by default, R1 will not have any suboptimal routes as described in the question. For example, if subnet 1 was in the OSPF domain, and R2 injected it into EIGRP, the route would have administrative distance 170. R1,

upon learning the EIGRP route to subnet 1, would prefer the OSPF (default administrative distance 110) route over the administrative distance 170 route.

6. a. Table 11-7 in Chapter 11 summarizes the defaults for metric types and metric values when performing redistribution. OSPF defaults to external type 2, EIGRP to external (but that is the only option), and RIP has no concept of route type—or has a single route type, depending on your perspective. OSPF defaults to external type 2, regardless of whether the route is redistributed via an ASBR inside a normal area or from one inside an NSSA area.

7. d. The three incorrect answers are classic descriptions of what route summarization does do. However, summary routes remove some details of the topology from the routing table, which in itself increases the possibility of suboptimal routes. It does nothing to help the generalized problem of suboptimal routing caused by redistribution.

8. a. EIGRP configures summarization using the **ip summary-address** interface subcommand. RIPv2 does not support summaries that are supernets, and it does not support multiple summaries of the same classful network on the same interface—both features that EIGRP supports. RIPv1 cannot support route summarization if it does not support VLSM.

9. a, c. Without the **always** keyword, OSPF requires that a route to 0.0.0.0/0 exist, but that route can be a dynamic or static route. EIGRP does not support the command.

Chapter 12

1. d. BGP neighbors must reach the established state, a steady state in which Update messages can be sent and received as needed.

2. c. While eBGP neighbors often share a common link, there is no requirement that neighbors must be connected to the same subnet.

3. a, d. BGP sets TTL to 1 only for messages sent over eBGP connections, so the **ebgp-multihop** option is only required in that case. The BGP router ID can be set to any syntactically valid number, in the format of an IP address, using the **bgp router-id** *id* command; it does not have to match another router's **neighbor** command.

4. d. When **no auto-summary** is configured, the **network** command must be an exact match of the prefix/prefix length. If omitted, the prefix length is assumed based on the default classful network mask—in this case, **network 20.0.0.0** would imply a mask of 255.0.0.0.

5. a, b. The **redistribute** command, when redistributing from an IGP, takes routes actually in the routing table as added by that IGP, or connected routes on interfaces matched by that IGP's **network** commands.

6. c. The BGP **auto-summary** command only affects routes locally injected into a router's BGP table through redistribution or the **network** command. The **network** command, with **auto-summary** enabled, only needs to match one subnet of the classful network, which it does in this case. The **aggregate-address** command creates the 9.0.0.0/8 aggregate, but without the **summary-only** keyword, it also advertises the component subnet 9.1.0.0/16.

7. a, c, e. BGP routes must be considered to be **valid** and **best** before being advertised. With iBGP, the route also must not have been learned from another iBGP peer. Finally, the NEXT_HOP must be reachable, but the local router determines reachability by looking in its IP routing table for a matching route—not by pinging the NEXT_HOP IP address.

8. a, d, e. The **redistribute** command injects routes with an assigned ORIGIN of incomplete, whereas those injected with the **network** command are considered as IGP routes. The **aggregate-address** command, without the **as-set** option, always sets the ORIGIN code of the aggregate to IGP.

9. d. For an iBGP-learned route, BGP synchronization requires that the NLRI (prefix/prefix length) be in the IP routing table, as learned via an IGP, before considering that BGP route as a candidate to be BGP's best route to that prefix. The other answers simply do not meet the definition of BGP synchronization.

10. a, b. Confederations use eBGP rules for confederation eBGP peers regarding multihop and the advertisement of iBGP routes to eBGP (confederation) peers. Inside a confederation AS, a full mesh must be maintained.

11. a, c, e. NLRI 1.0.0.0/8 was learned via eBGP, so R1 advertises it to all iBGP peers—the route reflector logic has no impact on that logic. NLRI 3.0.0.0/8 shows normal route reflector operation for a route sent to the reflector by a client—it is reflected to all clients and nonclients. The NLRI 5.0.0.0/8 answer lists normal route reflector operation for routes received from a nonclient—it is only reflected to clients.

12. c. One of the challenges with migration to a confederation configuration is that the ASN is no longer configured on the **router bgp** command, but rather on the **bgp confederation identifier** command. Also, the **bgp confederation peers** command lists the confederation ASNs of routers in other confederation sub-autonomous systems, but it is only required on routers that have neighbor connections to routers in other confederation ASNs. As a result, R3 does not need the command.

Chapter 13

1. b, d. Standard **access-list 1** cannot match on the prefix length/mask, and, as stated, it would match any subnets that start with 20.128. **prefix-list 1** uses an invalid parameter (**eq**). **access-list 101** matches the prefix length based on the destination IP address, and **prefix-list 2** matches the prefix length of exactly 20.

2. c. Changes to routing policies are not implemented until the neighbor connection is cleared. Once cleared, the **show ip bgp advertised-routes** command will no longer list the filtered routes.

3. a. **.*333_** allows matching of 33333 as well as 333, because the wildcard before the 333 would match 33. For similar reasons, **.*_333.*$** would also match 33333, due to the **.*** right after **333**. Finally, **^.*_333_.*$** is close, but the **_.*$** on the end does not match correctly in a case for which 333 is the last ASN in the AS_PATH.

4. a, b. The **distribute-list** command will filter all prefixes in the range 11.8.0.0 through 11.11.255.255, permitting those but filtering all others due to the implicit **deny all** at the end of the ACL.

5. b. Well-known attributes must be supported by every BGP implementation, whereas optional attributes do not. However, well-known PAs do not have to be used all the time, so they may not be included in BGP Updates. Nontransitive attributes should not be forwarded by BGP implementations that do not understand the attribute. Finally, discretionary attributes are well known, but they do not have to be configured for use at all times.

6. a, c

7. b, c. The ORIGIN is not a numeric value. The MED and the IGP metric both act as metrics, considering the lower number to be better.

8. a. The WEIGHT is a Cisco-proprietary mechanism used by one router to influence its own BGP decision process; because it is not a path attribute, there is not even a place in the Update message in which to place the attribute.

9. e. The three routes tie on all steps up to the Neighbor Type check; only one route was learned via eBGP (the one to next-hop 10.1.2.3), so, as the only eBGP route, it was considered better than the two iBGP routes.

10. a. The second-listed route wins on several counts, but the first of those to be considered is the highest administrative weight (30).

11. c. The configuration is syntactically correct, and would add three 1s to the AS_PATH. Additionally, R1 would add its own AS_PATH before sending the Update to its eBGP peer 3.3.3.3, making a total of four consecutive 1s in the AS_PATH. The routes to subnets of 12.0.0.0/8 would be filtered, because the route map does not contain any **permit** clauses matching the subnets of network 12.0.0.0/8.

12. d. BGP can use the **maximum-paths** command to impact its logic to place multiple IP routes into the IP routing table; however, only a single BGP route in the BGP table, for each prefix, can be considered to be a best route.

13. a, d. BGP RFCs use the term NO_EXPORT_SUBCONFED, whereas Cisco documents and commands use the term LOCAL_AS, for the same COMMUNITY value. This value implies that a route can be advertised to confederation iBGP peers, but not to confederation eBGP or normal eBGP peers.

14. c. BGP communities enable a router to set the same value for a group of routes, then enable other routers to apply the same policy logic based on a match of that COMMUNITY value. The actual COMMUNITY value is not considered during the BGP decision process.

Chapter 14

1. c
2. a, d. DSCP is the high-order 6 bits of the DS field, formerly known and the ToS byte. IPP occupies the high-order 3 bits of that same byte.
3. a, b, e. CS3's first 3 bits purposefully match IPP 3. Also, with a value of binary 011000, CS3's decimal equivalent is 24.
4. b, d. AF31's binary value is 011010, so the first 3 bits, which comprise the same bits as the IPP field, are 011. Also, binary 011010 converts to decimal 26.
5. a. The **class-map** command defaults to use the **match-all** parameter, which means both **match** commands' conditions must be true in order to match the class.
6. a, b. The **match cos 3 4** command uses OR logic between its two parameters, matching CoS 3 or 4. **class-map c2** uses match-any logic, so either **match** command can be true to match **class-map c2**. Finally, with match-all logic, **class-map c3** fails to match, because the frame has a CoS of 3, and the **match cos 2** command fails to match. The IPP and DSCP fields do not impact the actions taken by the listed configuration.
7. d. Each class map has an optional parameter of **match-all** (default) or **match-any**. With the default of **match-all**, both **match** commands in the class map must match, and a packet can't have both DSCP EF and AF31. After creating a set of class maps, and referring to them with **class** commands inside **policy-map barney**, you used the **service-policy input barney** command under **interface fa 0/0**. However, the **show policy-map interface fa 0/0** command shows that no packets match class fred.
8. b, e. Because the policy works for outgoing packets, the policy map cannot classify based on the DE bit, although the DE bit can be set. CoS and CLP do not exist in Frame Relay, so those fields cannot be set.
9. a. CB Marking requires that CEF is enabled globally, regardless of whether NBAR is being used. NBAR is in use in this case because the **match protocol** command tells Cisco IOS to use NBAR to match the parameters on that command. NBAR and the **match protocol** command can be used as an input or output function.

Chapter 15

1. c
2. b, c. CQ reserves a minimum amount of bandwidth for each queue by reserving a byte count for each queue. Like all queuing tools, CQ provides a place to hold packets while waiting for the hardware queue to release a packet and open a slot. Because it does not have a concept of allocating bandwidth for an LLQ, it cannot have a concept of allocating the bandwidth that remains after giving some to an LLQ.

3. a, c. WFQ may discard the newly arriving packet, or it may discard a previously enqueued packet in another queue, depending on sequence numbers. It checks the hold-queue limit against the total number of enqueued packets among all WFQ queues on the interface, and also checks the CDT versus the individual queue's length.

4. a, b. Multiple classes can be configured as LLQs with the **priority** command. Also, only one style of **bandwidth** command is allowed in a single policy map, making the last two answers incorrect.

5. b. To find the answer, take interface bandwidth (100 kbps), subtract 25 percent of the bandwidth (based on the default **max-reserved-bandwidth** of 75 percent). That leaves 75 kbps. Subtract 20 percent of the interface bandwidth (20 percent of 100 kbps) for the LLQ, which leaves 55 kbps. The *bandwidth remaining percent* features then allocate percentages of the remaining bandwidth, which is 55 kbps in this case.

6. b. WRED increases the discard rate from 0 to 1/MPD as the average moves from the minimum threshold to the maximum threshold.

7. a, b. WRED defaults to using IPP, so **random-detect** enables it for IPP, as does the explicit version of the command (**random-detect precedence-based**).

8. a. The **priority-queue out** command enables the PQ (expedite queue) feature on a 3550 interface. The **wrr-queue bandwidth** command does not allow a 0 to be configured for any queue; 1 is the lowest allowed value. The queue for a frame is chosen based on CoS, not DSCP, and any CoS value(s) can be placed into the queue.

9. a. 3550 WRED does not include MPD. There are effectively two WRED traffic profiles, each with a single threshold. The threshold essentially sets the minimum queue depth. The WRED discard percentage grows as the queue depth climbs from that minimum setting to 100 percent full.

Chapter 16

1. c. CB Shaping adds Bc tokens to the bucket at the beginning of each shaping time interval (Tc). The presence of a non-0 Be means that the bucket is larger, as it is Bc + Be large, but it does not mean that more tokens are added at each Tc. (CB Policing adds tokens based on packet arrival.)

2. c. The formula is Tc = Bc/CIR. Shaping uses a unit of "bits" for Bc, so the units work out easily. In this case, Tc = 3200/128,000, or 1/40 of a second—25 ms.

3. a. **shape peak** actually shapes at a higher rate than the configured rate. CB Shaping defaults Be to be equal to Bc, so to make it 0, you must set it directly. Also, the shaping rate is configured in bps, not kbps.

4. b. The command could also have been used a point 2 in the configuration snippet—as long as the command was configured inside **class-default** inside **policy-map shape-question**. The answers beginning with **shape queue** are not valid commands.

5. e. The **show policy-map** command lists only the formatted configuration, with no calculated or statistical values. Tc = Bc/CIR = 5120/256,000 = 1/50th of a second, or 20 ms.

6. b. FRTS assigns settings based on the following order of precedence: the **class** command under the **frame-relay interface-dlci** command, the **frame-relay class** command under the subinterface, or the **frame-relay class** command under the interface.

7. d. The **frame-relay traffic-rate** command does not allow setting the Tc (the time interval), and FRTS normally defaults that value to 125 ms. To impact the Tc, the Bc must be set; to make Tc 62.5 ms, the rate and Bc must be chosen such that Bc/rate = 62.5 ms, or 1/16 second.

8. b, d. Shapers delay packets, and policers either discard packets or re-mark them.

9. c. CB Policing defaults its Be setting to 0 when it is configured as a two-color policer. That occurs when the **police** command does not have a **violate** action configured, or an explicit Be value set. Also, the policing rate uses a unit of bps; both commands beginning **police 128** police at 128 bps, not 128 kbps. The **police 128k** command is syntactically incorrect.

10. a, c. CAR is always a single-rate, two-color policer, meaning that it supports only the conform and exceed actions. It does not use MQC commands. However, it does allow for policing supersets and subsets of interface traffic, and can police packets going in either direction on an interface.

Chapter 17

1. e. The T1 framing options are SF (also known as D4) and ESF. A T1 has 24 DS0 channels available for customer use, plus 8 kbps of overhead. With ESF, 4 kbps of the 8 kbps of overhead is used for OAM functions. Bits are robbed or stolen from each DS0 channel for voice signaling in some cases. Finally, of the T1 encoding options of AMI and B8ZS, only B8ZS uses BPVs to ensure the T1's density.

2. a, b, d. The DCE raises DSR to indicate that it is ready to begin working, raises CTS to indicate that the DTE may send now, and raises DCD to indicate that the CSU/DSU currently has a working line.

3. b, c. BPVs occur when the encoder sends binary 1s as a 3-mV signal of alternating polarity, but the next transmitted binary 1 signal has the same polarity as the previous signal. AMI does not use BPVs to encode bits, so a BPV when using AMI means at least one bit error has occurred. B8ZS uses BPVs as part of an encoded set of eight consecutive 0s to ensure enough signal transitions when sending strings of eight binary 0s; the transitions are required to maintain clock synchronization.

4. b. RTP header compression is negotiated by IPCP, whereas the other answers are all LCP features. LCP must complete before an NCP (like IPCP) may begin negotiation.

5. d. MLP fragments packets and load balances the packets over the various links. The fragment delay can be set, implying the fragment length based on delay × bandwidth = fragment length. However, the fragment delay does not have to be set. In that case, the fragment size is based on the number of links, in this case creating fragments 1/3 the size of the original packet. Because the balancing is at Layer 2, Layer 3 mechanisms like CEF and fast switching are not involved in the decision.

6. c, d. CHAP requires PPP, which in turn requires the **encapsulation ppp** command. To cause CHAP negotiation, the **ppp authentication chap** command is required. The **username** command is a global configuration command that refers to the other router's host name, making both answers with the **username** command incorrect. R1 would need a **username R2 password** *samepassword* global config command.

7. a. FRF.9 is only supported on Frame Relay interfaces. MPPC and Predictor are only supported on PPP links.

Chapter 18

1. b. The ANSI T1.617 Annex D LMI and the ITU Q.933 Annex A LMI are equivalent, using DLCI 0 for LMI flows and supporting a maximum of 976 PVCs/DLCIs. The Cisco LMI uses DLCI 1023 and supports 992 PVCs/DLCIs. Both ANSI and ITU LMI types can be autosensed, but so can the Cisco LMI type; the question asks for answers that apply only to ANSI and ITU LMI types.

2. c. The FECN signifies congestion from R1 to R2, so R2 would not slow down its adaptive shaping; R2 would slow down on receipt of a BECN. The receiver on the end of the PVC does not react to congestion; rather, the intermediate Frame Relay switches could react. To signal the congestion back to R1, R2 could be configured to "reflect" the FECN back to R1 by sending its next frame to R1 with BECN set; R1 could then adaptively shape to a lower rate. Finally, Cisco IOS policers cannot set FECN, but they can set DE.

3. b, e. The **encapsulation frame-relay** command defaults to **ietf** encapsulation; the **encapsulation frame-relay ietf** interface subcommand would achieve the same result. The **encapsulation cisco** command is syntactically incorrect, but the **encapsulation frame-relay cisco** interface subcommand would enable Frame Relay, with Cisco encapsulation, on the interface. However, the command is only a physical interface subcommand, not a subinterface subcommand. Encapsulation types can differ for each VC. Those without the encapsulation type listed on the **frame-relay interface-dlci** *dlci encapsulation-type* command use the encapsulation configured on the **encapsulation** interface subcommand; those with the encapsulation type on the **frame-relay interface-dlci** command use the specified type.

4. b. Frame Relay access links work when LMI is disabled on the router. Of the four answers with commands, the only valid command is **no keepalive**, which does indeed disable Frame Relay LMI.

5. b, d. Of the three options, only packet-by-packet uses a per-packet compression dictionary; only FRF.9 is standardized by a public standards body (Frame Relay Forum); and all three use the LZS algorithm. Any of the three types can be configured with the **frame-relay payload-compress** command or the **frame-relay map** command.

6. a, e. The LMI messages do not contain any information about the router's subinterfaces, so the DLCI must be associated with each subinterface, with the **frame-relay interface-dlci** command being one method. The encapsulation type and the use of **frame-relay interface-dlci** and **frame-relay map** commands are unrelated. The FR network does not care about the headers past the LAPF header, and therefore does not know about nor care to characterize the encapsulation type. Because the FR network does not care about the encapsulation type, different types can be mixed over the same access link.

7. a. A router can correlate a DLCI to a subinterface using two methods: a **frame-relay interface-dlci** command or the **frame-relay map** command. On the six subinterfaces with neither command configured, R1 will not know how to associate any DLCIs with the subinterfaces, so it cannot send traffic out those subinterfaces. The four subinterfaces with **frame-relay interface-dlci** commands do not need a **frame-relay map** command, because the router will receive InARP messages, see the DLCI listed, correlate that DLCI to the subinterface with the **frame-relay interface-dlci** command, and remember the associated next-hop address.

Chapter 19

1. b, c, d. Multicast packets are sent once from a source to many destinations, which eliminates traffic redundancy; hence, multicast uses less bandwidth than unicast. Multicast applications use UDP at the transport layer, which provides connectionless service.

2. c. A multicast address can be permanently assigned by IANA or can be temporarily assigned and relinquished. The multicast address range is 224.0.0.0 to 239.255.255.255. A multicast address is unstructured and does not use any subnet mask; therefore, it cannot be entered as an IP address on an interface of a router.

3. a, d. IANA reserves all the addresses in the range 224.0.0.0 to 224.0.0.255. Multicast routers do not forward packets with a destination address from this range. The addresses 224.0.1.39 and 224.0.1.40 are also reserved but routers can forward packets with these destination addresses.

4. d. An Ethernet multicast MAC address of 48 bits is calculated from a Layer 3 multicast address by using 0x0100.5e as the multicast vendor code (OUI) for the first 24 bits, always binary 0 for the 25th bit, and copying the last 23 bits of the Layer 3 multicast address.

5. d. Only hosts originate IGMP Membership messages and only routers originate IGMP Query messages. Switches only forward these messages.

6. d. In IGMPv2, when a router receives a Leave message, it responds by sending a Group-Specific Query using the multicast address that was used in the Leave message as the destination address.

7. f. Only IGMPv1 uses a fixed Maximum Response Time of 10 seconds. This timer can be configured for IGMPv2 and IGMPv3 from 0 to 25.5 seconds, and 0 to 53 minutes, respectively. IGMPv1 does not have any querier election process. For querier election, IGMPv1 depends on the multicast routing protocol. IGMPv2 elects the router with the *lowest* IP address as the querier. IGMPv3 is backward compatible with IGMPv2 and IGMPv1. Both IGMPv2 and IGMPv3 are capable of using the Leave messages.

8. e

Chapter 20

1. c. When a multicast router receives a multicast packet, it first performs the reverse-path forwarding (RPF) check to determine whether the packet entered through the same interface it would use to go toward the source; if it did not, the router drops the packet.

2. d. When multiple PIM routers are connected to a LAN subnet, they send Assert messages to determine which router will be the forwarder of the multicast traffic on the LAN. Both PIM-DM and PIM-SM routing protocols use Assert messages.

3. d. When a PIM-DM router receives a Graft message after it has sent a Prune message, it will send a Graft message to the upstream router. It does not send a Prune message to the downstream router and it does not have to re-establish adjacency with the upstream router.

4. c, e. A PIM-DM router sends Prune and Graft messages based on the demand for multicast group traffic. If nobody wants the group traffic, the PIM-DM router sends a Prune message to its upstream router. If somebody requests group traffic and the router is not receiving the traffic from its upstream router, it sends a Graft message to its upstream router.

5. a, c. The RP sends a Register-Stop message only when it does not need to receive the traffic, or when it does need to receive the traffic but the first-hop DR is now sending the multicast to the RP via the shortest-path tree to the RP.

6. c. R1 is not switching over from SPT to RPT. R1's upstream router on the shared tree will show the R flag only for its (S, G) entry.

7. b. The PIM Auto-RP messages will not reach all the PIM-SM routers if the **ip pim sparse-mode** command is configured on the interfaces of all the routers. Congestion is not a problem because all the routers show all the PIM neighbors, which means they are receiving multicast PIM Hello messages. The static RP configuration with an **override** option would

show at least some RP mapping on the leaf routers. In order for an interface to be considered for use by PIM-SM, the **ip pim sparse-mode** command must be configured under the subinterface.

8. c, d. PIM-SM routers can maintain the forwarding state on a link only by periodically (default every 60 seconds) sending PIM Join messages. PIM-SM routers choose to send the periodic Joins for two reasons. First, Joins are sent to the RP if a host on a connected network is sending IGMP Reports, claiming to want traffic sent to that multicast group. Second, Joins are sent to the RP if a router is receiving PIM Joins from a downstream router.

Chapter 21

1. d. AAA authentication for enable mode (privileged exec mode) only uses the default set of authentication modes listed in the **aaa authentication enable default group radius local** command. The **enable authentication wilma** and **aaa authentication enable wilma group fred local** commands are not valid commands; the **enable** command can only use a default set of AAA authentication methods. Also, with the **aaa new-model** and **aaa authentication enable** commands as listed, the **enable secret** password is not used.

2. a. The **aaa authentication login fred line group radius none** command defines a set of methods beginning with **line**, which means using the **password** command listed in line configuration mode. The **login authentication fred** command refers to group fred. As a result, the router begins by just asking for a password, and using the password listed in the **password cisco** command.

3. c. Because there is no **login authentication** subcommand under **line vty 0 4**, Telnet attempts to use the default methods defined in the **aaa authentication login default** command. The methods are tried in order: the servers in the default group of RADIUS servers, and then the local set of usernames and passwords. Because barney/betty is the only defined username/password, if neither RADIUS server replied, then barney/betty would be the required username/password.

4. a, b, e. Several reference documents regarding security best practices are available at http://www.cisco.com/go/safe includes. The core SAFE document lists Dynamic ARP Inspection (DAI) as one best practice; it watches ARP messages to prevent many ARP-based attacks. Also, shutting down unused ports and enabling port security are also recommended. SAFE further recommends not using VLAN 1 for any traffic, and to disable DTP completely; automatic mode would allow another switch, or a device masquerading as a switch, to use DTP to dynamically create a trunk.

5. d. Port security requires that each enabled port be statically configured as an access port or a trunking port. By default, port security allows a single MAC. Sticky-learned MACs are added to the running configuration only; dynamic-learned (nonsticky) MACs are only used until the next reload of the switch.

6. b. With 802.1X, the user device is the supplicant, the switch is the authenticator, and a RADIUS server is the authentication server. EAPoL is used between the supplicant and the authenticator. Until a port is authenticated, the switch will forward only 802.1X traffic (typically EAPoL), plus CDP and STP.

7. d. Mask 0.0.1.255 matches 10.44.38.0 through 10.44.39.255. Mask 0.0.7.255 matches 10.44.32.0 through 10.44.39.255; in fact, if used as suggested, a **show run** command would have listed it as **access-list 1 permit 10.44.32.0 0.0.7.255**. Mask 0.0.15.255 would imply a range of 10.44.32.0 through 10.44.47.255; the resulting command output of **show run** would list **access-list 1 permit 10.44.32.0 0.0.15.255**. Finally, mask 0.0.5.255 uses discontiguous 0s and 1s, which is valid; however, it would not match IP address 10.44.40.18.

8. a, b. Routers will forward packets sent to subnet (directed) broadcast addresses, except for the router connected to the subnet; its action is predicated on the setting of the **ip directed-broadcast** command. An RPF check would also filter the packets, because a router's route to reach 9.1.1.0/24 would point into the enterprise, not toward the Internet, which is from where the packet arrived. Finally, a packet filter for all IP addresses in the subnet would filter both legitimate traffic and the attack.

9. b. TCP intercept can either watch the connections, monitoring them, or inject itself into the process. It injects itself by responding to TCP connection requests, and then forming another TCP connection to the server—but only if the client-side connection completes. It is enabled globally, but it uses an ACL to define the scope of connections it processes. It is not specifically associated with an interface.

Chapter 22

1. a. 802.11a offers the highest capacity because it has 12 nonoverlapping RF channels, as compared to only 3 for 802.11b and 802.11g.

2. b. Wireless devices in an infrastructure mode configuration connect to an access point, which provides a wireless link between the user devices and the access point. All data frames must travel through the access point, even though the data traffic is between two wireless users. Data frames travel directly between each other only when ad hoc mode is implemented.

3. c. Repeaters merely receive and then retransmit 802.11 frames that they receive. This provides a duplicate data frame for every data frame received, which doubles the number of data frames sent over the network.

4. a. The 802.11 requirement for a receiving station to send an acknowledgment for every data frame introduces significant overhead on the wireless network. This lowers throughput. Only 802.11 data frames sent with protection mechanisms enabled use RTS/CTS. 802.11 doesn't implement forward error correction, so there is no redundancy in data frames.

5. d. All 802.11 wireless LANs must implement beacons in order to announce the presence of the network and facilitate association and power-save functions. The sharing of the transmission of beacons among all ad hoc stations is necessary to ensure that beacons are still sent if a particular station becomes unavailable.

6. a. In infrastructure mode configurations, the access point operates on a particular RF channel. Radio cards periodically search for beacons by scanning all RF channels. The radio card automatically tunes the RF channel of the access point prior to connecting with the access point.

7. b. The DTIM interval defines the number of beacons that are sent before multicast frames are sent by the access point. A sleeping station then knows to stay awake longer to receive the multicast frames.

8. c. With the distributed coordination function, an 802.11 station independently decides to access the medium if the medium is idle (no carrier detected) and the network allocation vector (NAV) value is zero. The NAV is set after the radio card receives a frame, which includes a duration ID indicating the value of time that the radio card must wait prior to attempting to access the medium. This allows 802.11 functions, such as RTS/CTS, to complete before a radio card accesses and ties up the medium.

9. c. Public wireless LANs do not implement Layer 2 security, such as 802.11i, WPA, and WEP. As a result, users should make use of VPNs, which offers end-to-end encryption.

10. a. A 6-dBi antenna has 3-dB gain as compared to a 3-dBi antenna, which effectively doubles the transmit power. This stronger signal travels farther.

Chapter 23

1. b. WDS is a current Cisco feature that eases the deployment, usage, and support of wireless LANs. WDS is not hardware.

2. d. SWAN includes all the basic building blocks necessary to effectively implement and support a wireless LAN.

3. a. The Cisco Aironet 1000, 1100, and 1200 Series access points are designed for indoor use. The Cisco Aironet 1300 is the only SWAN access point designed for outdoor use.

4. b. The Cisco Aironet 350 is an older access point model that was released prior to the ratification of 802.11g and 802.11e.

5. d

6. b. To support voice services, the SNR must be at least 25 dB. At this SNR, the data rate will be optimal, which provides adequate performance. The coverage overlap must be at least 20 percent to avoid dropped calls as users roam through the facility. An SNR of 20 dB would provide good enough signal coverage for data-only applications, but it is not good enough for voice services.

7. c. Most RF interference is in the 2.4-GHz band, which is the band that 802.11b/g uses. As a result, you can avoid RF interference by using 802.11a, which operates in the 5-GHz band. Using additional Wi-Fi phones or decreasing the access point transmit power will not help counter RF interference.

8. b. If you disable the broadcasting of SSIDs with public wireless LANs, then Microsoft Windows on the user devices will not display the network to users. Therefore, it is best to broadcast the beacons.

9. a. With public wireless LANs, a billing component is necessary to charge users for access. The security of a public wireless LAN, however, is generally kept to a minimum to increase the openness of the network. Enterprise wireless LANs, though, include maximum security features and no billing functions. The performance, access point hardware, and management mechanisms of public and enterprise wireless LANs are generally the same.

10. d. With small office and home wireless LANs, a Wi-Fi router is necessary to enable the sharing of a common official IP address (supplied by the ISP) among multiple computers. Thus, DHCP and NAT are needed. Both Wi-Fi routers and access points can operate at the same speed, provide the same level of security, and have the same longevity.

Chapter 24

1. a, d. By asking IP to send data for it, TCP is using the features of an adjacent lower layer—hence it is an example of adjacent-layer interaction. TCP's features most closely relate to OSI Layer 4.

2. d. The layered structure does not necessarily mean that the software written to those specifications will run faster; in fact, the protocol details required for interoperability between layers and protocols may actually result in additional requirements and additional code.

3. e. The OSI layers (1 through 7) are physical, data link, network, transport, session, presentation, and application.

4. a. With a boot field of 0x0, the router will load ROMMON code. The two **boot** commands are syntactically invalid.

5. b

APPENDIX B

CCIE Routing and Switching Exam Updates: Version 1.0

Over time, reader feedback allows Cisco Press to gauge which topics give our readers the most problems when taking the exams. Additionally, Cisco may make small changes to the CCIE Routing and Switching exam blueprint. To assist readers with those topics, the authors create new materials clarifying and expanding upon those troublesome exam topics. As mentioned in the introduction to the *CCIE Routing and Switching Exam Certification Guide*, Second Edition, the additional content about the exam is contained in a PDF document on this book's companion website, at http://www.ciscopress.com/title/1587201410.

This appendix presents all the latest update information available at the time of this book's printing. To make sure you have the latest version of this document, visit the companion website to see if any more recent versions have been posted since this book went to press.

This appendix attempts to fill the void that occurs with any print book. In particular, this appendix does the following:

- Mentions technical items that might not have been mentioned elsewhere in the book
- Covers new topics when Cisco adds topics to the CCIE Routing and Switching written exam blueprint
- Provides a way to get up-to-the-minute current information about content for the exam

Always Get the Latest at the Companion Website

You are reading the version of this appendix that was available when your book was printed. However, given that the main purpose of this appendix is to be a living, changing document, it is very important that you look for the latest version online at the book's companion website. To do so:

1. Browse to http://www.ciscopress.com/title/1587201410.
2. Select the **Downloads** option under the **More Information** box.

3. Download the latest "Appendix B" document.

> **NOTE** Note that the downloaded document has a version number. Comparing the version of this print Appendix B (Version 1.0) with the latest online version of this appendix, you should do the following:
>
> - **Same version**—Ignore the PDF that you downloaded from the companion website.
> - **Website has a later version**—Ignore this Appendix B in your book, and just read the latest version that you downloaded from the companion website.

Technical Content

The current version of this appendix does not contain any additional technical coverage. However, a completely separate appendix (Appendix C) covers MPLS. At the time of printing, MPLS was not on the published Cisco CCIE Routing and Switching written exam blueprint. However, because it was being considered as a new topic, we added Appendix C to the book. Please check http://www.cisco.com/go/ccie for the latest blueprint, and study Appendix C if MPLS has been added.

APPENDIX C

MPLS

Cisco continually reviews the content of the various CCIE programs. On occasion, these reviews result in the addition or removal of topics on the written and/or practical exams. As of the time that this book was going to press, Cisco was considering the addition of MPLS basics back into the CCIE R/S written exam. This appendix is included for reference in case Cisco does indeed add the topic into the exam.

To know if you should read this appendix, or a more thorough separate book on MPLS, please refer to www.cisco.com/go/ccie, and look for the CCIE R/S Written Blueprint. It is also helpful to look at their "News and Announcements" section, which is typically where such changes are first announced.

This appendix is an extract from the very popular, well-written, and thorough book *MPLS and VPN Architectures*, by Ivan Pepelnjak and Jim Guichard. If you are looking for a more thorough reference on MPLS, this book would be my best suggestion on a good starting place at which to dig in to the topic. This appendix is an extract of all or parts of chapters 1, 2, and 8 from this book.

Architecture Overview

Traditional IP packet forwarding analyzes the destination IP address contained in the network layer header of each packet as the packet travels from its source to its final destination. A router analyzes the destination IP address independently at each hop in the network. Dynamic routing protocols or static configuration builds the database needed to analyze the destination IP address (the routing table). The process of implementing traditional IP routing also is called *hop-by-hop destination-based unicast routing.* Although successful, and obviously widely deployed, certain restrictions, which have been realized for some time, exist for this method of packet forwarding that diminish its flexibility. New techniques are therefore required to address and expand the functionality of an IP-based network infrastructure.

This first section concentrates on identifying these restrictions and presents a new architecture, known as *Multiprotocol Label Switching (MPLS),* that provides solutions to some of these restrictions. The following section focuses first on the details of the MPLS architecture in a pure router environment, and then on the use of MPLS to create VPNs.

Scalability and Flexibility of IP-Based Forwarding

To understand all the issues that affect the scalability and the flexibility of traditional IP packet forwarding networks, you must start with a review of some of the basic IP forwarding mechanisms and their interaction with the underlying infrastructure (local- or wide-area networks). With this information, you can identify any drawbacks to the existing approach and perhaps provide alternative ideas on how this could be improved.

Network Layer Routing Paradigm

Traditional network layer packet forwarding (for example, forwarding of IP packets across the Internet) relies on the information provided by network layer routing protocols (for example, Open Shortest Path First [OSPF] or Border Gateway Protocol [BGP]), or static routing, to make an independent forwarding decision at each hop (router) within the network. The forwarding decision is based solely on the destination unicast IP address. All packets for the same destination follow the same path across the network if no other equal-cost paths exist. Whenever a router has two equal-cost paths toward a destination, the packets toward the destination might take one or both of them, resulting in some degree of load sharing.

NOTE Enhanced Interior Gateway Routing Protocol (EIGRP) also supports non–equal-cost load sharing although the default behavior of this protocol is equal-cost. You must configure EIGRP *variance* for non–equal-cost load balancing. Please see *EIGRP Network Design Solutions* (ISBN 1-57870-165-1), from Cisco Press for more details on EIGRP.

Load sharing in Cisco IOS can be performed on a packet-by-packet or source-destination-pair basis (with Cisco Express Forwarding [CEF] switching) or on a destination basis (most of the other switching methods).

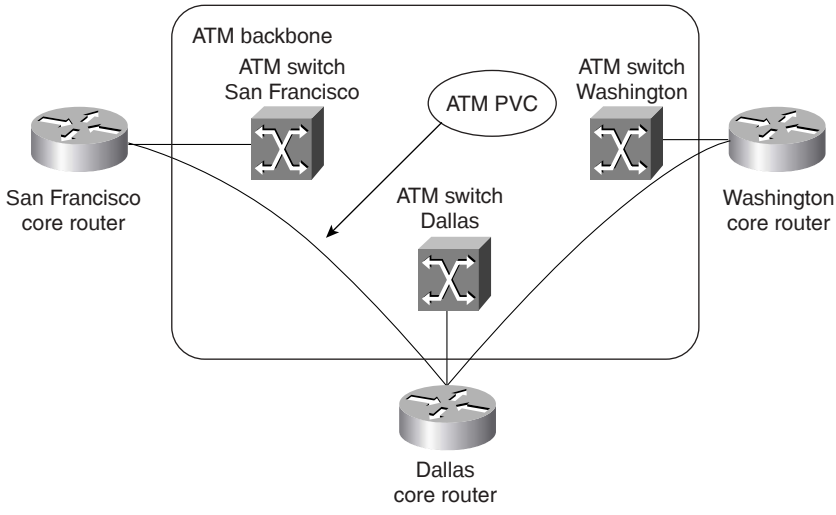
Routers perform the decision process that selects what path a packet takes. These network layer devices participate in the collection and distribution of network-layer information, and perform Layer 3 switching based on the contents of the network layer header of each packet. You can connect the routers directly by point-to-point links or local-area networks (for example, shared hub or MAU), or you can connect them by LAN or WAN switches (for example, Frame Relay or ATM switches). These Layer 2 (LAN or WAN) switches unfortunately do not have the capability to hold Layer 3 routing information or to select the path taken by a packet through analysis of its Layer 3 destination address. Thus, Layer 2 (LAN or WAN) switches cannot be involved in the Layer 3 packet forwarding decision process. In the case of the WAN environment, the network designer has to establish Layer 2 paths manually across the WAN network. These paths then forward Layer 3 packets between the routers that are connected physically to the Layer 2 network.

LAN Layer 2 paths are simple to establish—all LAN switches are transparent to the devices connected to them. The WAN Layer 2 path establishment is more complex. WAN Layer 2 paths usually are based on a point-to-point paradigm (for example, virtual circuits in most WAN

networks) and are established only on request through manual configuration. Any routing device (ingress router) at the edge of the Layer 2 network that wants to forward Layer 3 packets to any other routing device (egress router) therefore needs to either establish a direct connection across the network to the egress device or send its data to a different device for transmission to the final destination.

Consider, for example, the network shown in Figure C-1.

Figure C-1 *Sample IP Network Based on ATM Core*



The network illustrated in Figure C-1 is based on an ATM core surrounded by routers that perform network layer forwarding. Assuming that the only connections between the routers are the ones shown in Figure C-1, all the packets sent from San Francisco to or via Washington must be sent to the Dallas router, where they are analyzed and sent back over the same ATM connection in Dallas to the Washington router. This extra step introduces delay in the network and unnecessarily loads the CPU of the Dallas router as well as the ATM link between the Dallas router and the adjacent ATM switch in Dallas.

To ensure optimal packet forwarding in the network, an ATM virtual circuit must exist between any two routers connected to the ATM core. Although this might be easy to achieve in small networks, such as the one in Figure C-1, you run into serious scalability problems in large networks where several tens or even hundreds of routers connect to the same WAN core.

The following facts illustrate the scalability problems you might encounter:

- Every time a new router is connected to the WAN core of the network, a virtual circuit must be established between this router and any other router, if optimal routing is required.

> **NOTE** In Frame Relay networks, the entire configuration could be done within the Layer 2 WAN core and the routers would find new neighbors and their Layer 3 protocol addresses through the use of LMI and Inverse ARP. This also is possible on an ATM network through the use of Inverse ARP, which is enabled by default when a new PVC is added to the configuration of the router, and ILMI, which can discover PVCs dynamically that are configured on the local ATM switch.

- With certain routing protocol configurations, every router attached to the Layer 2 WAN core (built with ATM or Frame Relay switches) needs a dedicated virtual circuit to every other router attached to the same core. To achieve the desired core redundancy, every router also must establish a routing protocol adjacency with every other router attached to the same core. The resulting full-mesh of router adjacencies results in every router having a large number of routing protocol neighbors, resulting in large amounts of routing traffic. For example, if the network runs OSPF or IS-IS as its routing protocol, every router propagates every change in the network topology to every other router connected to the same WAN backbone, resulting in routing traffic proportional to the *square* of the number of routers.

> **NOTE** Configuration tools exist in recent Cisco IOS implementations of IS-IS and OSPF routing protocols that allow you to reduce the routing protocol traffic in the network. Discussing the design and the configuration of these tools is beyond the scope of this book (any interested reader should refer to the relevant Cisco IOS configuration guides).

- Provisioning of the virtual circuits between the routers is complex, because it's very hard to predict the exact amount of traffic between any two routers in the network. To simplify the provisioning, some service providers just opt for lack of service guarantee in the network—zero Committed Information Rate (CIR) in a Frame Relay network or Unspecified Bit Rate (UBR) connections in an ATM network.

The lack of information exchange between the routers and the WAN switches was not an issue for traditional Internet service providers that used router-only backbones or for traditional service providers that provided just the WAN services (ATM or Frame Relay virtual circuits). There are, however, several drivers that push both groups toward mixed backbone designs:

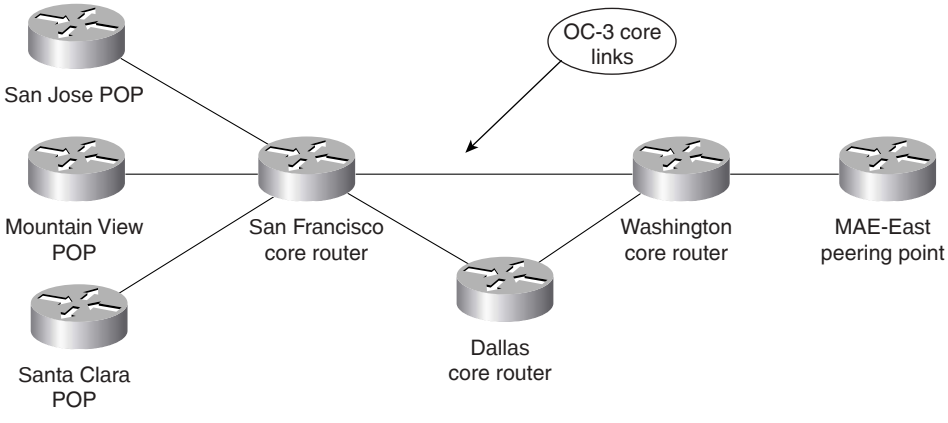
- Traditional service providers are asked to offer IP services. They want to leverage their investments and base these new services on their existing WAN infrastructure.
- Internet service providers are asked to provide tighter quality of service (QoS) guarantees that are easier to meet with ATM switches than with traditional routers.
- The rapid increase in bandwidth requirements prior to the introduction of optical router interfaces forced some large service providers to start relying on ATM technology because the router interfaces at that time did not provide the speeds offered by the ATM switches.

It is clear, therefore, that a different mechanism must be used to enable the exchange of network layer information between the routers and the WAN switches and to allow the switches to participate in the decision process of forwarding packets so that direct connections between edge routers are no longer required.

Differentiated Packet Servicing

Conventional IP packet forwarding uses only the IP destination address contained within the Layer 3 header within a packet to make a forwarding decision. The hop-by-hop destination-only paradigm used today prevents a number of innovative approaches to network design and traffic-flow optimization. In Figure C-2, for example, the direct link between the San Francisco core router and the Washington core router forwards the traffic entering the network in any of the Bay Area Points-of-Presence (POPs), although that link might be congested and the links from San Francisco to Dallas and from Dallas to Washington might be only lightly loaded.

Figure C-2 *Sample Network that Would Benefit from Traffic Engineering*



Although certain techniques exist to affect the decision process, such as Policy Based Routing (PBR), no single scalable technique exists to decide on the full path a packet takes across the network to its final destination. In the network shown in Figure C-2, the policy- based routing must be deployed on the San Francisco core router to divert some of the Bay Area to Washington traffic toward Dallas. Deploying such features as PBR on core routers could severely reduce the performance of a core router and result in a rather unscalable network design. Ideally, the edge routers (for example, the Santa Clara POP in Figure C-2) can specify over which core links the packets should flow.

NOTE Several additional issues are associated with policy-based routing. PBR can lead easily to forwarding loops as a router configured with PBR deviates from the forwarding path learned from the routing protocols. PBR also is hard to deploy in large networks; if you configure PBR at the edge, you must be sure that *all* routers in the forwarding path can make the *same* route selection.

Because most major service providers deploy networks with redundant paths, a requirement clearly exists to allow the ingress routing device to be capable of deciding on packet forwarding, which affects the path a packet takes across the network, and of applying a *label* to that packet that indicates to other devices which path the packet should take.

This requirement also should allow packets that are destined for the same IP network to take separate paths instead of the path determined by the Layer 3 routing protocol. This decision also should be based on factors other than the destination IP address of the packet, such as from which port the packet was learned, what quality of service level the packet requires, and so on.

Independent Forwarding and Control

With conventional IP packet forwarding, any change in the information that controls the forwarding of packets is communicated to all devices within the routing domain. This change always involves a period of convergence within the forwarding algorithm.

A mechanism that can change how a packet is forwarded, without affecting other devices within the network, certainly is desirable. To implement such a mechanism, forwarding devices (routers) should not rely on IP header information to forward the packet; thus, an additional label must be attached to a forwarded packet to indicate its desired forwarding behavior. With the packet forwarding being performed based on labels attached to the original IP packets, any change within the decision process can be communicated to other devices through the distribution of new labels. Because these devices merely forward traffic based on the attached label, a change should be able to occur without any impact at all on any devices that perform packet forwarding.

External Routing Information Propagation

Conventional packet forwarding within the core of an IP network requires that external routing information be advertised to all transit routing devices. This is necessary so that packets can be routed based on the destination address that is contained within the network layer header of the packet. To continue the example from previous sections, the core routers in Figure C-2 would have to store all Internet routes so that they could propagate packets between Bay Area customers and a peering point in MAE-East.

NOTE You might argue that each major service provider also must have a peering point somewhere on the West coast. That fact, although true, is not relevant to this discussion because you can always find a scenario where a core router with no customers or peering partners connected to it needs complete routing information to be able to forward IP packets correctly.

This method has scalability implications in terms of route propagation, memory usage, and CPU utilization on the core routers, and is not really a required function if all you want to do is pass a packet from one edge of the network to another.

A mechanism that allows internal routing devices to *switch* the packets across the network from an ingress router toward an egress router without analyzing network layer destination addresses is an obvious requirement.

Multiprotocol Label Switching Introduction

Multiprotocol Label Switching (MPLS) is an emerging technology that aims to address many of the existing issues associated with packet forwarding in today's Internetworking environment. Members of the IETF community worked extensively to bring a set of standards to market and to evolve the ideas of several vendors and individuals in the area of *label switching.* The IETF document *draft-ietf-mpls-framework* contains the framework of this initiative and describes the primary goal as follows:

> The primary goal of the MPLS working group is to standardize a base technology that integrates the label swapping forwarding paradigm with network layer routing. This base technology (label swapping) is expected to improve the price/performance of network layer routing, improve the scalability of the network layer, and provide greater flexibility in the delivery of (new) routing services (by allowing new routing services to be added without a change to the forwarding paradigm).

NOTE You can download IETF working documents from the IETF home page (www.ietf.org). For MPLS working documents, start at the MPLS home page (www.ietf.org/html.charters/mpls-charter.html).

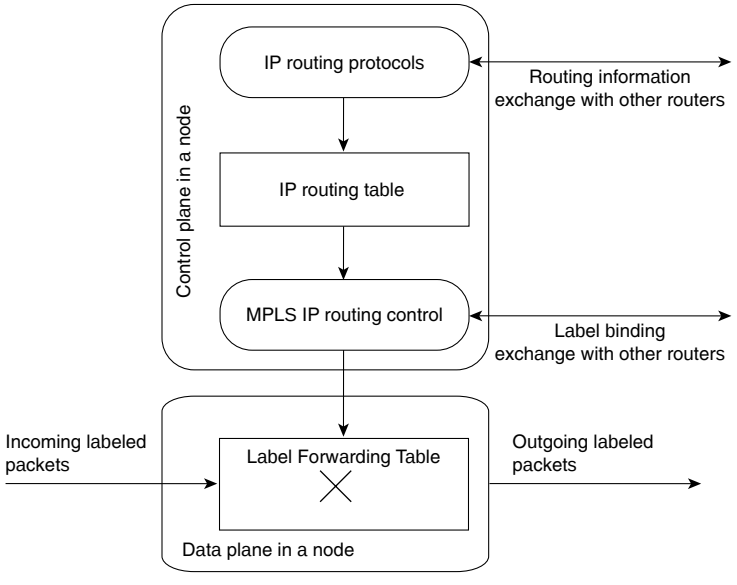
The MPLS architecture describes the mechanisms to perform label switching, which combines the benefits of packet forwarding based on Layer 2 switching with the benefits of Layer 3 routing. Similar to Layer 2 networks (for example, Frame Relay or ATM), MPLS assigns *labels* to packets for transport across packet- or cell-based networks. The forwarding mechanism throughout the network is *label swapping,* in which units of data (for example, a packet or a cell) carry a short, fixed-length label that tells switching nodes along the packets path how to process and forward the data.

The significant difference between MPLS and traditional WAN technologies is the way labels are assigned and the capability to carry a stack of labels attached to a packet. The concept of a label stack enables new applications, such as Traffic Engineering, Virtual Private Networks, fast rerouting around link and node failures, and so on.

Packet forwarding in MPLS is in stark contrast to today's connectionless network environment, where each packet is analyzed on a hop-by-hop basis, its Layer 3 header is checked, and an independent forwarding decision is made based on the information extracted from a network layer routing algorithm.

The architecture is split into two separate components: the *forwarding* component (also called the *data plane*) and the *control* component (also called the *control plane*). The forwarding component uses a label-forwarding database maintained by a label switch to perform the forwarding of data packets based on labels carried by packets. The control component is responsible for creating and maintaining label-forwarding information (referred to as *bindings*) among a group of interconnected label switches. Figure C-3 shows the basic architecture of an MPLS node performing IP routing.

Figure C-3 *Basic Architecture of an MPLS Node Performing IP Routing*



Every MPLS node must run one or more IP routing protocols (or rely on static routing) to exchange IP routing information with other MPLS nodes in the network. In this sense, every MPLS node (including ATM switches) is an IP router on the control plane.

Similar to traditional routers, the IP routing protocols populate the IP routing table. In traditional IP routers, the IP routing table is used to build the IP forwarding cache (fast switching cache in Cisco IOS) or the IP forwarding table (Forwarding Information Base [FIB] in Cisco IOS) used by Cisco Express Forwarding (CEF).

In an MPLS node, the IP routing table is used to determine the label binding exchange, where adjacent MPLS nodes exchange labels for individual subnets that are contained within the IP routing table. The label binding exchange for unicast destination-based IP routing is performed using the Cisco proprietary Tag Distribution Protocol (TDP) or the IETF-specified Label Distribution Protocol (LDP).

The MPLS IP Routing Control process uses labels exchanged with adjacent MPLS nodes to build the Label Forwarding Table, which is the forwarding plane database that is used to forward labeled packets through the MPLS network.

MPLS Architecture—The Building Blocks

As with any new technology, several new terms are introduced to describe the devices that make up the architecture. These new terms describe the functionality of each device and their roles within the MPLS domain structure.

The first device to be introduced is the *Label Switch Router (LSR)*. Any router or switch that implements label distribution procedures and can forward packets based on labels falls under this category. The basic function of label distribution procedures is to allow an LSR to distribute its label bindings to other LSRs within the MPLS network. (The "Frame-Mode MPLS Operation" section discusses label distribution procedures in detail.) Several different types of LSR exist that are differentiated by what functionality they provide within the network infrastructure. These different types of LSR are described within the architecture as *Edge-LSR, ATM-LSR,* and *ATM edge-LSR.* The distinction between various LSR types is purely architectural—a single box can serve several of the roles.

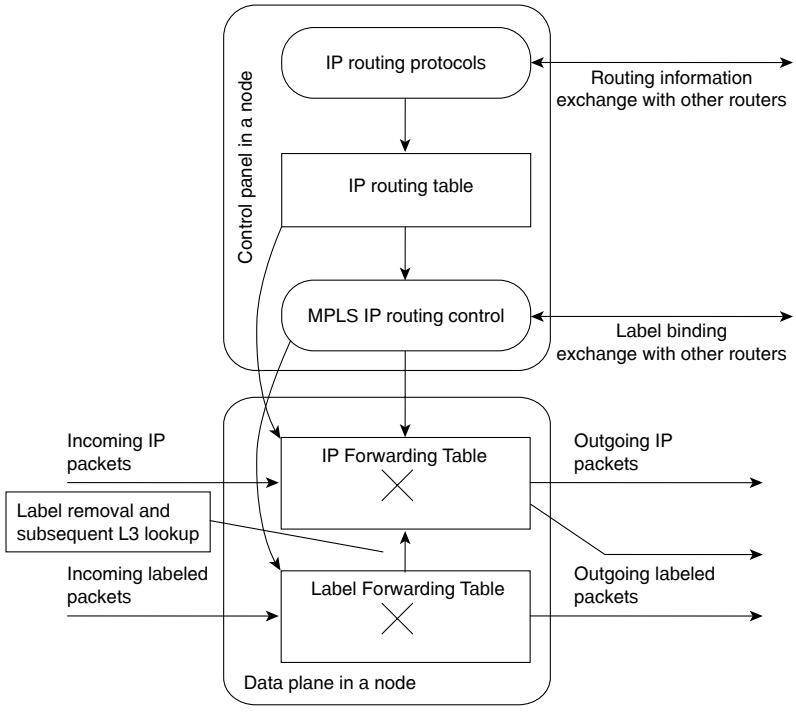
An Edge-LSR is a router that performs either label imposition (sometimes also referred to as *push* action) or label disposition (also called *pop* action) at the edge of the MPLS network. Label imposition is the act of prepending a label, or a stack of labels, to a packet in the ingress point (in respect of the traffic flow from source to destination) of the MPLS domain. Label disposition is the reverse of this and is the act of removing the last label from a packet at the egress point before it is forwarded to a neighbor that is outside the MPLS domain.

Any LSR that has any non-MPLS neighbors is considered an Edge-LSR. However, if that LSR has any interfaces that connect through MPLS to an ATM-LSR, then it also is considered to be an ATM edge-LSR. Edge-LSRs use a traditional IP forwarding table, augmented with labeling information, to label IP packets or to remove labels from labeled packets before sending them to non-MPLS nodes. Figure C-4 shows the architecture of an Edge-LSR.

An Edge-LSR extends the MPLS node architecture from Figure C-3 with additional components in the data plane. The standard IP forwarding table is built from the IP routing table and is

extended with labeling information. Incoming IP packets can be forwarded as pure IP packets to non-MPLS nodes or can be labeled and sent out as labeled packets to other MPLS nodes. The incoming labeled packets can be forwarded as labeled packets to other MPLS nodes. For labeled packets destined for non-MPLS nodes, the label is removed and a Layer 3 lookup (IP forwarding) is performed to find the non-MPLS destination.

Figure C-4 *Architecture of an Edge-LSR*



An ATM-LSR is an ATM switch that can act as an LSR. The Cisco Systems, Inc. LS1010 and BPX family of switches are examples of this type of LSR. The ATM-LSR performs IP routing and label assignment in the control plane and forwards the data packets using traditional ATM cell switching mechanisms on the data plane. In other words, the ATM switching matrix of an ATM switch is used as a Label Forwarding Table of an MPLS node. Traditional ATM switches, therefore, can be redeployed as ATM-LSRs through a software upgrade of their control component.

Table C-1 summarizes the functions performed by different LSR types. Please note that any individual device in the network can perform more than one function (for example, it can be Edge-LSR and ATM edge-LSR at the same time).

Table C-1 *Actions Performed by Various LSR Types*

LSR Type	Actions Performed by This LSR Type
LSR	Forwards labeled packets.
Edge-LSR	Can receive an IP packet, perform Layer 3 lookups, and impose a label stack before forwarding the packet into the LSR domain.
	Can receive a labeled packet, remove labels, perform Layer 3 lookups, and forward the IP packet toward its next-hop.
ATM-LSR	Runs MPLS protocols in the control plane to set up ATM virtual circuits. Forwards labeled packets as ATM cells.
ATM edge-LSR	Can receive a labeled or unlabeled packet, segment it into ATM cells, and forward the cells toward the next-hop ATM-LSR.
	Can receive ATM cells from an adjacent ATM-LSR, reassemble these cells into the original packet, and then forward the packet as a labeled or unlabeled packet.

Label Imposition at the Network Edge

Label imposition has been described already as the act of prepending a label to a packet as it enters the MPLS domain. This is an edge function, which means that packets are labeled before they are forwarded to the MPLS domain.

To perform this function, an Edge-LSR needs to understand where the packet is headed and which label, or stack of labels, it should assign to the packet. In conventional Layer 3 IP forwarding, each hop in the network performs a lookup in the IP forwarding table for the IP destination address contained in the Layer 3 header of the packet. It selects a next hop IP address for the packet at each iteration of the lookup and eventually sends the packet out of an interface toward its final destination.

NOTE Some forwarding mechanisms, such as CEF, allow the router to associate each destination prefix known in the routing table to the adjacent next-hop of the destination prefix, thus solving the recursive lookup problem. The whole recursion is resolved while the router populates the cache or the forwarding table and not when it has to forward packets.

Choosing the next hop for the IP packet is a combination of two functions. The first function partitions the entire set of possible packets into a set of IP destination prefixes. The second function

maps each IP destination prefix to an IP next hop address. This means that each destination in the network is reachable by one path in respect to traffic flow from one ingress device to the destination egress device (multiple paths might be available if load balancing is performed using equal-cost paths or unequal-cost paths as with some IGP protocols, such as Enhanced IGRP).

Within the MPLS architecture, the results of the first function are known as *Forwarding Equivalence Classes (FECs)*. These can be visualized as describing a group of IP packets that are forwarded in the same manner, over the same path, with the same forwarding treatment.

NOTE A Forwarding Equivalence Class might correspond to a destination IP subnet, but also might correspond to any traffic class that the Edge-LSR considers significant. For example, all interactive traffic toward a certain destination or all traffic with a certain value of IP precedence might constitute an FEC. As another example, an FEC can be a subset of the BGP table, including all destination prefixes reachable through the same exit point (egress BGP router).

With conventional IP forwarding, the previously described packet processing is performed at each hop in the network. However, when MPLS is introduced, a particular packet is assigned to a particular FEC just once, and this is at the edge device as the packet enters the network. The FEC to which the packet is assigned is then encoded as a short fixed-length identifier, known as a *label*.

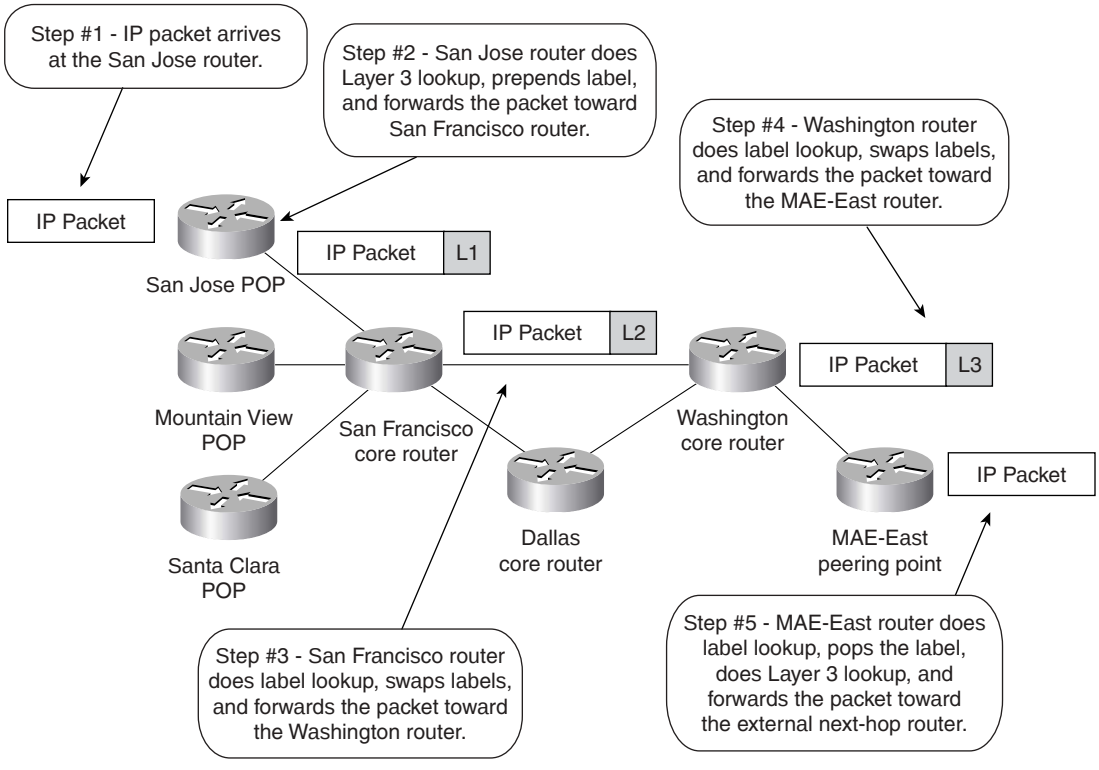
When a packet is forwarded to its next hop, the label is prepended already to the IP packet so that the next device in the path of the packet can forward it based on the encoded label rather than through the analysis of the Layer 3 header information. Figure C-5 illustrates the whole process of label imposition and forwarding.

NOTE The actual packet forwarding between the Washington and MAE-East routers might be slightly different from the one shown in Figure C-5 due to a mechanism called *penultimate hop popping (PHP)*. Penultimate hop popping arguably might improve the switching performance, but does not impact the logic of label switching. The next section covers this mechanism and its implications.

MPLS Packet Forwarding and Label Switched Paths

Each packet enters an MPLS network at an ingress LSR and exits the MPLS network at an egress LSR. This mechanism creates what is known as an *Label Switched Path (LSP)*, which essentially describes the set of LSRs through which a labeled packet must traverse to reach the egress LSR for a particular FEC. This LSP is unidirectional, which means that a different LSP is used for return traffic from a particular FEC.

Figure C-5 *MPLS Label Imposition and Forwarding*



The creation of the LSP is a connection-oriented scheme because the path is set up prior to any traffic flow. However, this connection setup is based on topology information rather than a requirement for traffic flow. This means that the path is created regardless of whether any traffic actually is required to flow along the path to a particular set of FECs.

As the packet traverses the MPLS network, each LSR swaps the incoming label with an outgoing label, much like the mechanism used today within ATM where the VPI/VCI is swapped to a different VPI/VCI pair when exiting the ATM switch. This continues until the last LSR, known as the egress LSR, is reached.

Each LSR keeps two tables, which hold information that is relevant to the MPLS forwarding component. The first, known in Cisco IOS as the *Tag Information Base (TIB)* or *Label Information Base (LIB)* in standard MPLS terms, holds all labels assigned by this LSR and the mappings of these labels to labels received from any neighbors. These label mappings are distributed through the use of label-distribution protocols, which The next section discusses in more detail.

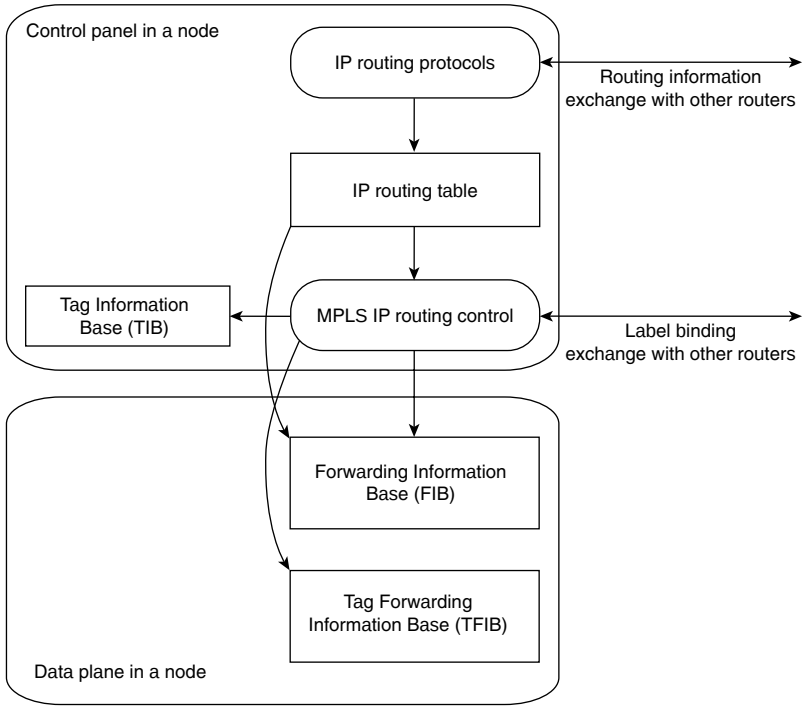
Just as multiple neighbors can send labels for the same IP prefix but might not be the actual IP next hop currently in use in the routing table for the destination, not all the labels within the TIB/LIB

need to be used for packet forwarding. The second table, known in Cisco IOS as the *Tag Forwarding Information Base (TFIB) or Label Forwarding Information Base (LFIB)* in MPLS terms, is used during the actual forwarding of packets and holds only labels that are in use currently by the forwarding component of MPLS.

NOTE Label Forwarding Information Base is the MPLS equivalent of the switching matrix of an ATM switch.

Using Cisco IOS terms and Cisco Express Forwarding (CEF) terminology, the Edge-LSR architecture in Figure C-4 can be redrawn as shown in Figure C-6 (Edge-LSR was chosen because its function is a superset of non–Edge-LSR).

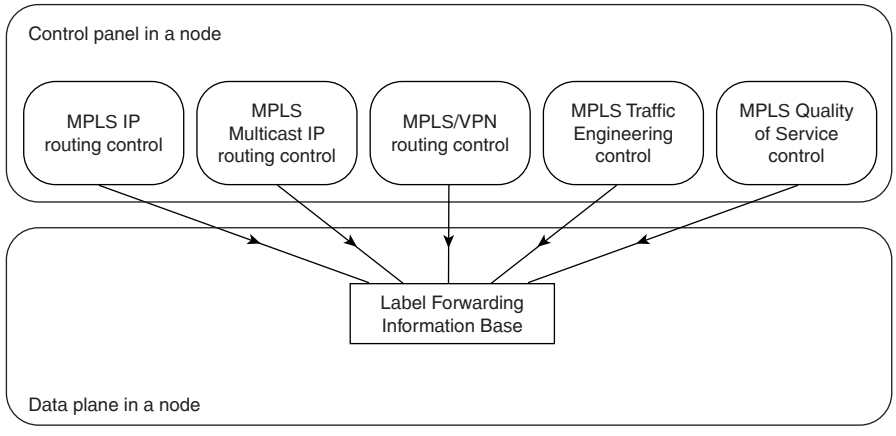
Figure C-6 *Edge-LSR Architecture Using Cisco IOS Terms*



Other MPLS Applications

The MPLS architecture, as discussed so far, enables the smooth integration of traditional routers and ATM switches in a unified IP backbone (IP+ATM architecture). The real power of MPLS, however, lies in other applications that were made possible, ranging from traffic engineering to peer-to-peer Virtual Private Networks. All MPLS applications use control-plane functionality similar to the IP routing control plane shown in Figure C-6 to set up the label switching database. Figure C-7 outlines the interaction between these applications and the label-switching matrix.

Figure C-7 *Various MPLS Applications and Their Interactions*



Every MPLS application has the same set of components as the IP routing application:

- A database defining the Forward Equivalence Classes (FECs) table for the application (the IP routing table in an IP routing application)
- Control protocols that exchange the contents of the FEC table between the LSRs (IP routing protocols or static routing in an IP routing application)
- Control process that performs label binding to FECs and a protocol to exchange label bindings between LSRs (TDP or LDP in an IP routing application)
- Optionally, an internal database of FEC-to-label mapping (Label Information Base in an IP routing application)

Each application uses its own set of protocols to exchange FEC table or FEC-to-label mapping between nodes. Table C-2 summarizes the protocols and the data structures.

Table C-2 *Control Protocols Used in Various MPLS Applications*

Application	FEC Table	Control Protocol Used to Build FEC Table	Control Protocol Used to Exchange FEC-to-Label Mapping
IP routing	IP routing table	Any IP routing protocol	Tag Distribution Protocol (TDP) or Label Distribution Protocol (LDP)
Multicast IP routing	Multicast routing table	PIM	PIM version 2 extensions

continues

Table C-2 *Control Protocols Used in Various MPLS Applications (Continued)*

Application	FEC Table	Control Protocol Used to Build FEC Table	Control Protocol Used to Exchange FEC-to-Label Mapping
VPN routing	Per-VPN routing table	Most IP routing protocols between service provider and customer, Multiprotocol BGP inside the service provider network	Multiprotocol BGP
Traffic engineering	MPLS tunnels definition	Manual interface definitions, extensions to IS-IS or OSPF	RSVP or CR-LDP
MPLS Quality of Service	IP routing table	IP routing protocols	Extensions to TDP LDP

Summary: Architecture Overview

Traditional IP routing has several well-known limitations, ranging from scalability issues to poor support of traffic engineering and poor integration with Layer 2 backbones already existing in large service provider networks. With the rapid growth of the Internet and the establishment of IP as the Layer 3 protocol of choice in most environments, the drawbacks of traditional IP routing became more and more obvious.

MPLS was created to combine the benefits of connectionless Layer 3 routing and forwarding with connection-oriented Layer 2 forwarding. MPLS clearly separates the control plane, where Layer 3 routing protocols establish the paths used for packet forwarding, and the data plane, where Layer 2 label switched paths forward data packets across the MPLS infrastructure. MPLS also simplifies per-hop data forwarding, where it replaces the Layer 3 lookup function performed in traditional routers with simpler label swapping. The simplicity of data plane packet forwarding and its similarity to existing Layer 2 technologies enable traditional WAN equipment (ATM or Frame Relay switches) to be redeployed as MPLS nodes (supporting IP routing in the control plane) just with software upgrades to their control plane.

The control component in the MPLS node uses its internal data structure to identify potential traffic classes (also called *Forward Equivalence Classes*). A protocol is used between control components in MPLS nodes to exchange the contents of the FEC database and the FEC-to-label mapping. The FEC table and FEC-to-label mapping is used in Edge- LSRs to label ingress packets and send them into the MPLS network. The Label Forwarding Information Base (LFIB) is built within each MPLS node based on the contents of the FEC tables and the FEC-to-label mapping exchanged between the nodes. The LFIB then is used to propagate labeled packets across the MPLS network, similar to the function performed by an ATM switching matrix in the ATM switches.

The MPLS architecture is generic enough to support other applications besides IP routing. The simplest additions to the architecture are the IP multicast routing and quality of service extensions. The MPLS connection-oriented forwarding mechanism together with Layer 2 label-based look ups in the network core also has enabled a range of novel applications, from Traffic Engineering to real peer-to-peer Virtual Private Networks.

Frame-Mode MPLS Operation

In the first section of this appendix you saw the overall MPLS architecture as well as the underlying concepts. This chapter focuses on one particular application: unicast destination-based IP routing in a pure router environment (also called Frame-mode MPLS because the labeled packets are exchanged as frames on Layer 2).

This section first focuses on the MPLS data plane, assuming that the labels were somehow agreed upon between the routers. The next section explains the exact mechanisms used to distribute the labels between the routers, and the last section covers the interaction between label distribution protocols, the Interior Gateway Protocol (IGP), and the Border Gateway Protocol (BGP) in a service provider network.

Throughout this section, we refer to the generic architecture of an MPLS Label Switch router (LSR), as shown in Figure C-8, and use the sample service provider network (called SuperNet) shown in Figure C-9 for any configuration or debugging printouts.

The SuperNet network uses unnumbered serial links based on loopback interfaces that have IP addresses from Table C-3.

Table C-3 *Loopback Addresses in the SuperNet Network*

Router	Loopback Interface
San Jose	172.16.1.1/32
Mountain View	172.16.1.2/32
Santa Clara	172.16.1.3/32
San Francisco	172.16.1.4/32
Dallas	172.16.2.1/32
Washington	172.16.3.1/32
New York	172.16.3.2/32
MAE-East	172.16.4.1/32

Figure C-8 *Edge-LSR Architecture*

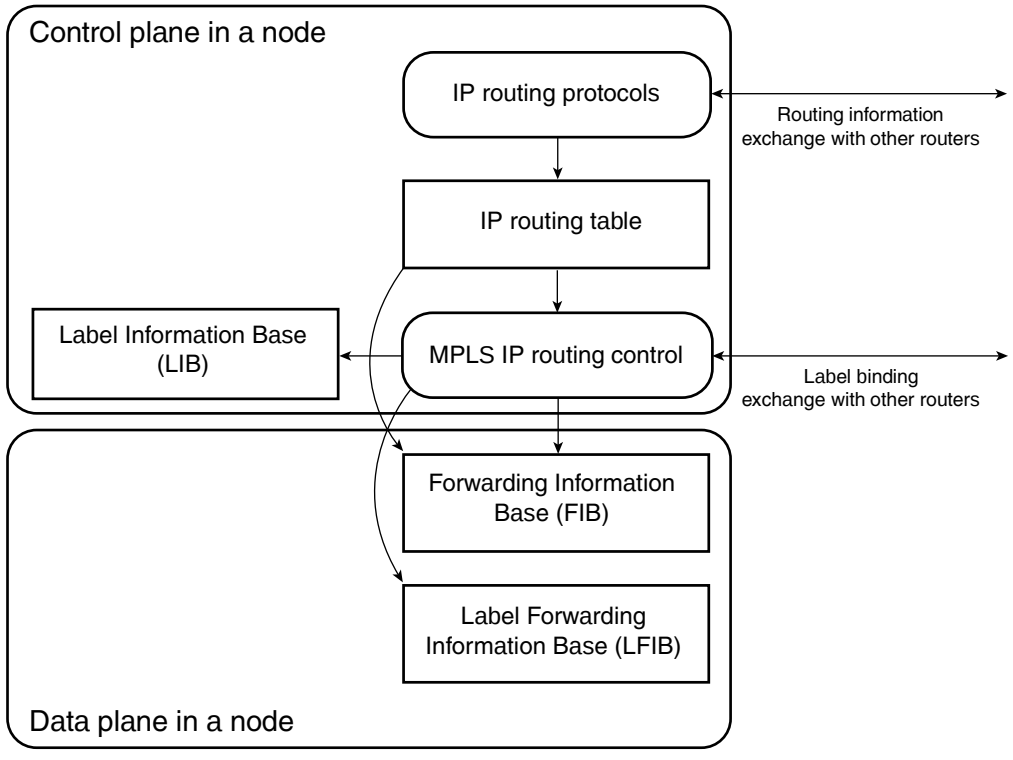
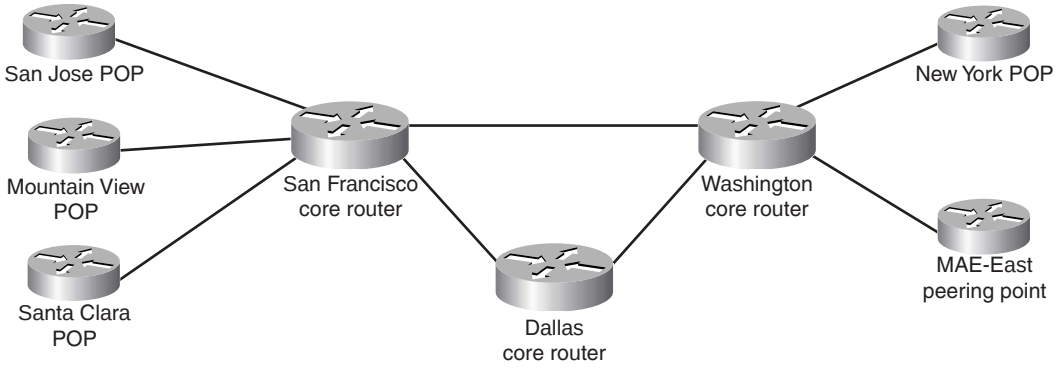


Figure C-9 *SuperNet Service Provider Network*



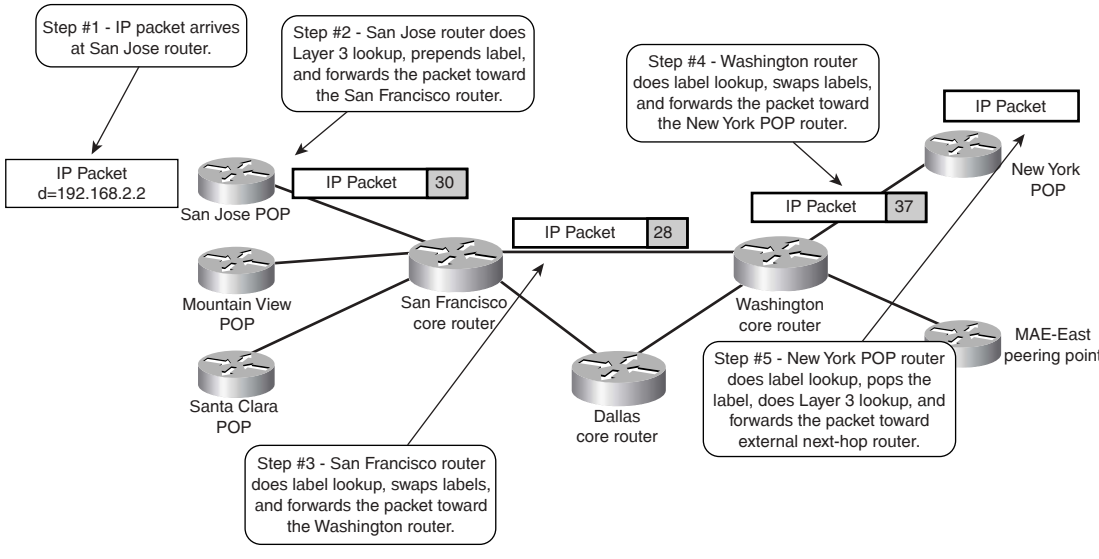
Frame-Mode MPLS Data Plane Operation

There are three major steps in the propagation of an IP packet across an MPLS backbone.

- The Ingress Edge-LSR receives an IP packet, classifies the packet into a forward equivalence class (FEC), and labels the packet with the outgoing label stack corresponding to the FEC. For unicast destination-based IP routing, the FEC corresponds to a destination subnet and the packet classification is a traditional Layer 3 lookup in the forwarding table.
- Core LSRs receive this labeled packet and use label forwarding tables to exchange the inbound label in the incoming packet with the outbound label corresponding to the same FEC (IP subnet, in this case).
- When the Egress Edge-LSR for this particular FEC receives the labeled packet, it removes the label and performs a traditional Layer 3 lookup on the resulting IP packet.

Figure C-10 shows these steps being performed in the SuperNet network for a packet traversing the network from the San Jose POP toward a customer attached to the NewYork POP.

Figure C-10 *Packet Forwarding Between San Jose POP and New York Customer*



The San Jose POP router receives an IP packet with the destination address of 192.168.2.2 and performs a traditional Layer 3 lookup through the IP forwarding table (also called *Forwarding Information Base [FIB]*).

NOTE Because Cisco Express Forwarding (CEF) is the only Layer 3 switching mechanism that uses the FIB table, CEF must be enabled in all the routers running MPLS *and* all the ingress interfaces receiving unlabeled IP packets that are propagated as labeled packets across an MPLS backbone must support CEF switching.

The core routers do not perform CEF switching—they just switch labeled packets—but they still must have CEF enabled globally for label allocation purposes.

The entry in the FIB (shown in Example C-1) indicates that the San Jose POP router should forward the IP packet it just received as a labeled packet. Thus, the San Jose router imposes the label "30" into the packet before it's forwarded to the San Francisco router, which brings up the first question: Where is the label imposed and how does the San Francisco router know that the packet it received is a labeled packet and not a pure IP packet?

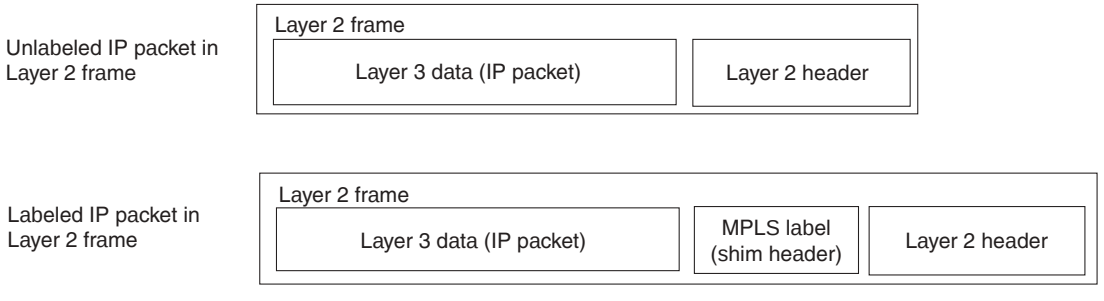
Example C-1 *CEF Entry in the San Jose POP Router*

```
SanJose#show ip cef 192.168.2.0
192.168.2.0/24, version 11, cached adjacency to Serial1/0/1
0 packets, 0 bytes
  tag information set
    local tag: 29
    fast tag rewrite with Se1/0/1, point2point, tags imposed: {30}
  via 172.16.1.4, Serial1/0/1, 0 dependencies
    next hop 172.16.1.4, Serial1/0/1
    valid cached adjacency
    tag rewrite with Se1/0/1, point2point, tags imposed: {30}
```

MPLS Label Stack Header

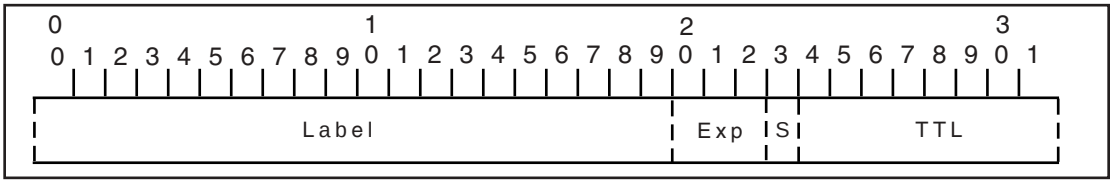
For various reasons, switching performance being one, the MPLS label must be inserted in front of the labeled data in a frame-mode implementation of the MPLS architecture. The MPLS label thus is inserted between the Layer 2 header and the Layer 3 contents of the Layer 2 frame, as displayed in Figure C-11.

Figure C-11 *Position of the MPLS Label in a Layer 2 Frame*



Due to the way an MPLS label is inserted between the Layer 3 packet and the Layer 2 header, the MPLS label header also is called the *shim header*. The MPLS label header (detailed in Figure C-12) contains the MPLS label (20 bits), the class-of-service information (three bits, also called *experimental bits*, in the IETF MPLS documentation), and the eight-bit Time-to-Live (TTL) field (which has the identical functions in loop detection as the IP TTL field) and one bit called the *Bottom-of-Stack* bit.

Figure C-12 *MPLS Label Stack Header*



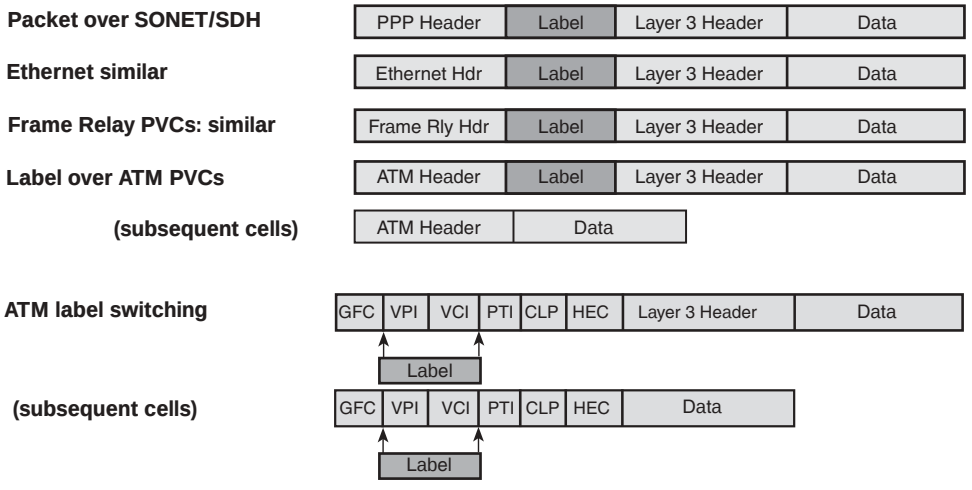
The Bottom-of-Stack bit implements an MPLS label stack, which is defined as a combination of two or more label headers attached to a single packet. Simple unicast IP routing does not use the label stack, but other MPLS applications, including MPLS-based Virtual Private Networks or MPLS Traffic Engineering, rely heavily on it.

With the MPLS label stack header being inserted between the Layer 2 header and the Layer 3 payload, the sending router must have some means to indicate to the receiving router that the packet being transmitted is not a pure IP datagram but a labeled packet (an MPLS datagram). To facilitate this, new protocol types were defined above Layer 2 as follows:

- In LAN environments, labeled packets carrying unicast and multicast Layer 3 packets use ethertype values 8847 hex and 8848 hex. These ethertype values can be used directly on Ethernet media (including Fast Ethernet and Gigabit Ethernet) as well as part of the SNAP header on other LAN media (including Token Ring and FDDI).
- On point-to-point links using PPP encapsulation, a new Network Control Protocol (NCP) called MPLS Control Protocol (MPLSCP) was introduced. MPLS packets are marked with PPP Protocol field value 8281 hex.
- MPLS packets transmitted across a Frame Relay DLCI between a pair of routers are marked with Frame Relay SNAP Network Layer Protocol ID (NLPID), followed by a SNAP header with type ethertype value 8847 hex.
- MPLS packets transmitted between a pair of routers over an ATM Forum virtual circuit are encapsulated with a SNAP header that uses ethertype values equal to those used in the LAN environment.

Figure C-13 shows the summary of all the MPLS encapsulation techniques.

Figure C-13 *Summary of MPLS Encapsulation Techniques*



The San Jose router in the example shown in Figure C-10 inserts the MPLS label in front of the IP packet just received, encapsulates the labeled packet in a PPP frame with a PPP Protocol field value of 8281 hex, and forwards the Layer 2 frame toward the San Francisco router.

Label Switching in Frame-Mode MPLS

After receiving the Layer 2 PPP frame from the San Jose router, the San Francisco router immediately identifies the received packet as a labeled packet based on its PPP Protocol field value and performs a label lookup in its Label Forwarding Information Base (LFIB).

> **NOTE** LFIB also is called Tag Forwarding Information Base (TFIB) in older Cisco documentation.

The LFIB entry corresponding to inbound label 30 (and displayed in Example C-2) directs the San Francisco router to replace the label 30 with an outbound label 28 and to propagate the packet toward the Washington router.

Example C-2 *LFIB Entry for Label 30 in the San Francisco Router*

```
SanFrancisco#show tag forwarding-table tags 30 detail
Local  Outgoing    Prefix           Bytes tag  Outgoing   Next Hop tag    tag or VC   or
  Tunnel Id      switched   interface
30     28          192.168.2.0/24    0          Se0/0/1    172.16.3.1
        MAC/Encaps=14/18, MTU=1504, Tag Stack{28}
        00107BB59E2000107BEC6B008847 0001C000
    Per-packet load-sharing
```

The labeled packet is propagated in a similar fashion across the SuperNet backbone until it reaches the NewYork POP, where the LFIB entry tells the NewYork router to pop the label and forward the unlabeled packet (see Example C-3).

Example C-3 *LFIB Entry in the New York Router*

```
NewYork#show tag forwarding-table tags 37 detail
Local  Outgoing    Prefix           Bytes tag  Outgoing   Next Hop tag    tag or VC   or
  Tunnel Id      switched   interface
37     untagged       192.168.2.0/24 0          Se2/1/3    192.168.2.1
        MAC/Encaps=0/0, MTU=1504, Tag Stack{}
    Per-packet load-sharing
```

A Cisco router running Cisco IOS software and operating as an MPLS LSR in Frame-mode MPLS can perform a number of actions on a labeled packet:

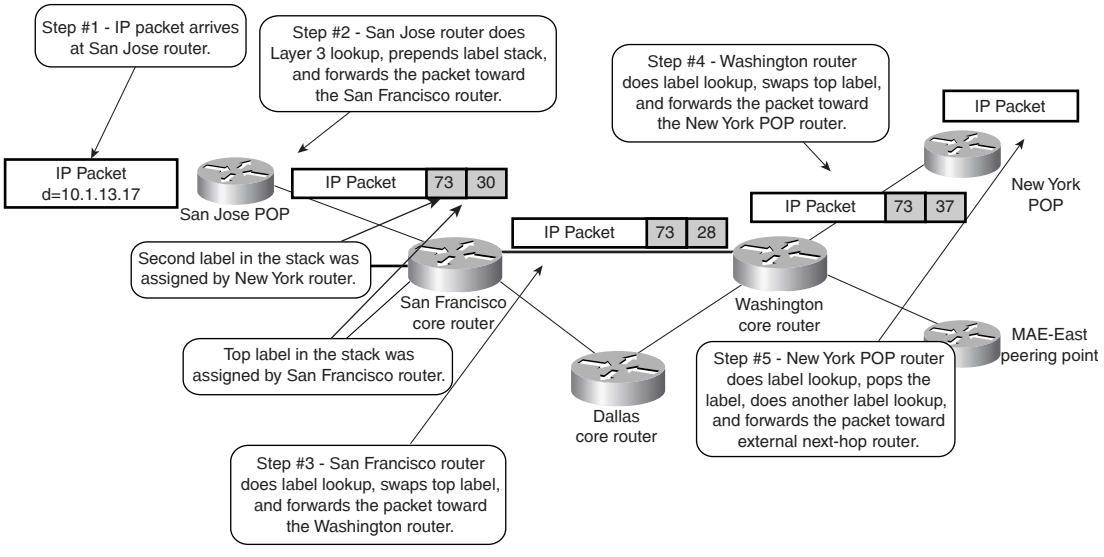
- **Pop tag—**Removes the top label in the MPLS label stack and propagates the remaining payload as either a labeled packet (if the Bottom-of-Stack bit is zero) or as an unlabeled IP packet (the Tag Stack field in the LFIB is empty).
- **Swap tag—**Replaces the top label in the MPLS label stack with another value (the Tag Stack field in the LFIB is one label long)
- **Push tag—**Replaces the top label in the MPLS label stack with a set of labels (the Tag Stack field in the LFIB contains several labels).
- **Aggregate—**Removes the top label in the MPLS label stack and does a Layer 3 lookup on the underlying IP packet. The removed label is the bottom label in the MPLS label stack; otherwise, the datagram is discarded.
- **Untag—**Removes the top label in the MPLS label stack and forwards the underlying IP packet to the specified IP next-hop. The removed label is the bottom label in the MPLS label stack; otherwise, the datagram is discarded.

MPLS Label Switching with Label Stack

The label switching operation is performed in the same way regardless of whether the labeled packet contains only one label or a label stack several labels deep. In both cases, the LSR switching the packet acts only on the top label in the stack, ignoring the other labels. This function enables a variety of MPLS applications where the edge routers can agree on packet classification rules and associated labels without knowledge of the core routers. For example, assume that the San Jose router and the New York router in the SuperNet network support MPLS-based Virtual Private Networks and that they have agreed that network 10.1.0.0/16, which is reachable through the New York router, is assigned a label value of 73. The core routers in the SuperNet network (San Francisco and Washington) are not aware of this.

To send a packet to a destination host in network 10.1.0.0/16, the San Jose router builds a label stack. The bottom label in the stack is the label agreed upon with the NewYork router, and the top label in the stack is the label assigned to the IP address of the NewYork router by the San Francisco router. When the network propagates the packet (as displayed in Figure C-14), the top label is switched exactly like in the example where a pure IP packet was propagated across the backbone and the second label in the stack reaches the New York router intact.

Figure C-14 *Label Switching with the MPLS Label Stack*



Label Bindings and Propagation in Frame-Mode MPLS

The previous section identifies the mechanisms necessary to forward labeled packets between the LSRs using framed interfaces (LAN, point-to-point links, or WAN virtual circuits). This section focuses on FEC-to-label bindings and their propagation between LSRs over framed interfaces.

Cisco IOS software implements two label binding protocols that can be used to associate IP subnets with MPLS labels for the purpose of unicast destination-based routing:

- **Tag Distribution Protocol (TDP)—**Cisco's proprietary protocol available in IOS software release 11.1CT, as well as 12.0 and all subsequent IOS releases
- **Label Distribution Protocol (LDP)—**IETF standard label binding protocol available in 12.2T release

TDP and LDP functionally are equivalent and can be used concurrently within the network, even on different interfaces of the same LSR. Due to their functional equivalence, this section shows only TDP debugging and monitoring commands.

To start MPLS packet labeling for unicast IP packets and associated protocols on an interface, use the commands in Table C-4.

Table C-4 *IOS Configuration Commands Used to Start MPLS on an Interface*

Task	IOS Command
Start MPLS packet labeling and run TDP on the specified interface.	**tag-switching ip**
Start MPLS packet labeling on the specified interface. TDP is used as the default label distribution protocol. Note: This command is equivalent to the tag-switching **ip** command.	**mpls ip**
Select the label distribution protocol on the specified interface.	**mpls label-distribution [ldp | tdp |both]**

LDP/TDP Session Establishment

When you start MPLS on the first interface in a router, the TDP/LDP process is started and the Label Information Base (LIB) structure is created. The router also tries to discover other LSRs on the interfaces running MPLS through TDP hello packets. The TDP hello packets are sent as broadcast or multicast UDP packets, making LSR neighbor discovery automatic. The **debug tag tdp transport** command can monitor the TDP hellos. Example C-4 shows the TDP process startup and Example C-5 illustrates the successful establishment of a TDP adjacency.

> **NOTE** The **debug mpls** commands replace the **debug tag** commands in IOS images with LDP support.

Example C-4 *TDP Startup After the First Interface Is Configured for MPLS*

```
SanFrancisco#debug tag tdp transport TDP
transport   events   debugging   is   on
SanFrancisco#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
SanFrancisco(config)#interface serial 1/0/1
SanFrancisco(config-subif)#tag-switching ip

1d20h: enabling tdp on Serial1/0/1
1d20h: tdp: 1<tdp_start: tdp_process_ptr = 0x80B7826C
1d20h: tdp: tdp_set_intf_id: intf 0x80E49B74, Serial1/0/1, not tc-atm, intf_id 0
1d20h: enabling tdp on Serial1/0/1
1d20h: tdp: Got TDP Id
1d20h: tdp: Got TDP TCP Listen socket
1d20h: tdp: tdp_hello_process tdp inited
1d20h: tdp: tdp_hello_process start hello for Serial1/0/1
1d20h: tdp: Got TDP UDP socket
```

Example C-5 *TDP Neighbor Discovery*

```
1d20h: tdp: Send hello; Serial1/0/1, src/dst 172.16.1.4/255.255.255.255, inst_id 0
1d20h: tdp: Rcvd hello; Serial1/0/1, from 172.16.1.1 (172.16.1.1:0), intf_id 0, opt
  0x4
1d20h: tdp: Hello from 172.16.1.1 (172.16.1.1:0) to 255.255.255.255, opt 0x4
```

There also might be cases where an adjacent LSR wants to establish an LDP or TDP session with the LSR under consideration, but the interface connecting the two is not configured for MPLS due to security or other administrative reasons. In such a case, the debugging printout similar to the printout shown in Example C-6 indicates ignored hello packets being received through interfaces on which MPLS is not configured.

Example C-6 *Ignored TDP Hello*

```
1d20h: tdp: Ignore Hello from 172.16.3.1, Serial0/0/1; no intf
```

After the TDP hello process discovers a TDP neighbor, a TDP session is established with the neighbor. TDP sessions are run on the well-known TCP port 711; LDP uses TCP port 646. TCP is used as the transport protocol (similar to BGP) to ensure reliable information delivery. Using TCP as the underlying transport protocol also results in excellent flow control properties and good adjustments to interface congestion conditions. Example C-7 shows the TDP session establishment.

Example C-7 *TDP Session Establishment*

```
1d20h: tdp: New adj 0x80EA92D4 from 172.16.1.1 (172.16.1.1:0), Serial1/0/1
1d20h: tdp: Opening conn; adj 0x80EA92D4, 172.16.1.4 <-> 172.16.1.1
1d20h: tdp: Conn is up; adj 0x80EA92D4, 172.16.1.4:11000 <-> 172.16.1.1:711
1d20h: tdp: Sent open PIE to 172.16.1.1 (pp 0x0)
1d20h: tdp: Rcvd open PIE from 172.16.1.1 (pp 0x0)
```

After a TDP session is established, it's monitored constantly with TDP keepalive packets to ensure that it's still operational. Example C-8 shows the TDP keepalive packets.

Example C-8 *TDP Keepalives*

```
1d20h: tdp: Sent keep_alive PIE to 172.16.1.1:0 (pp 0x0)
1d20h: tdp: Rcvd keep_alive PIE from 172.16.1.1:0 (pp 0x0)
```

The TDP neighbors and the status of individual TDP sessions also can be monitored with **show tag tdp neighbor** command, as shown in Example C-9. This printout was taken at the moment when the San Jose router was the only TDP neighbor of the San Francisco router.

Example C-9 *Show Tag TDP Neighbor Printout*

```
SanFrancisco#show tag-switching tdp neighbor
  Peer TDP Ident: 172.16.1.1:0; Local TDP Ident 172.16.1.4:0
  TCP connection: 172.16.1.1.711 - 172.16.1.4.11000
                  State: Oper; PIEs sent/rcvd: 4/4; ; Downstream
                  Up time: 00:01:05
                  TDP discovery sources:
                   Serial1/0.1
    Addresses bound to peer TDP Ident:
    172.16.1.1
```

The command displays the TDP identifiers of the local and remote routers, the IP addresses and the TCP port numbers between which the TDP connection is established, the connection uptime and the interfaces through which the TDP neighbor was discovered, as well as all the interface IP addresses used by the TDP neighbor.

NOTE The TDP identifier is determined in the same way as the OSPF or BGP identifier (unless controlled by the **tag tdp router-id** command)—the highest IP address of all loopback interfaces is used. If no loopback interfaces are configured on the router, the TDP identifier becomes the highest IP address of any interface that was operational at the TDP process startup time.

NOTE The IP address used as the TDP identifier *must be reachable* by adjacent LSRs; otherwise, the TDP/LDP session cannot be established.

Label Binding and Distribution

As soon as the Label Information Base (LIB) is created in a router, a label is assigned to every Forward Equivalence Class known to the router. For unicast destination-based routing, the FEC is equivalent to an IGP prefix in the IP routing table. Thus, a label is assigned to every prefix in the IP routing table and the mapping between the two is stored in the LIB.

NOTE Labels are not assigned to BGP routes in the IP routing table. The BGP routes use the same label as the interior route toward the BGP next hop. For more information on MPLS/BGP integration, see the section, "MPLS Interaction with the Border Gateway Protocol," later in this section.

The Label Information Base is always kept synchronized to the IP routing table—as soon as a new non-BGP route appears in the IP routing table, a new label is allocated and bound to the new route. The **debug tag tdp bindings** printouts show the subnet-to-label binding. Example C-10 shows a sample printout.

Example C-10 *Sample Label-to-prefix Bindings*

```
SanFrancisco#debug tag-switching tdp bindings
TDP Tag Information Base (TIB) changes debugging is on
1d20h: tagcon: tibent(172.16.1.4/32): created; find route tags request
1d20h: tagcon: tibent(172.16.1.4/32): lcl tag 1 (#2) assigned
1d20h: tagcon: tibent(172.16.1.1/32): created; find route tags request
1d20h: tagcon: tibent(172.16.1.1/32): lcl tag 26 (#4) assigned
1d20h: tagcon: tibent(172.16.1.3/32): created; find route tags request
1d20h: tagcon: tibent(172.16.1.3/32): lcl tag 27 (#6) assigned
1d20h: tagcon: tibent(172.16.1.2/32): created; find route tags request
1d20h: tagcon: tibent(172.16.1.2/32): lcl tag 28 (#8) assigned
1d20h: tagcon: tibent(192.168.1.0/24): created; find route tags request
1d20h: tagcon: tibent(192.168.1.0/24): lcl tag 1 (#10) assigned
1d20h: tagcon: tibent(192.168.2.0/24): created; find route tags request
1d20h: tagcon: tibent(192.168.2.0/24): lcl tag 29 (#12) assigned
```

Because the LSR assigns a label to each IP prefix in its routing table as soon as the prefix appears in the routing table, and the label is meant to be used by other LSRs to send the labeled packets toward the assigning LSR, this method of label allocation and label distribution is called *independent control* label assignment, with *unsolicited downstream* label distribution:

- The label allocation in routers is done regardless of whether the router has received a label for the same prefix already from its next-hop router or not. Thus, label allocation in routers is called *independent control*.
- The distribution method is unsolicited because the LSR assigns the label and advertises the mapping to upstream neighbors regardless of whether other LSRs need the label. The on-demand distribution method is the other possibility. An LSR assigns only a label to an IP prefix and distributes it to upstream neighbors when asked to do so.
- The distribution method is downstream when the LSR assigns a label that other LSRs (upstream LSRs) can use to forward labeled packets and advertises these label mappings to its neighbors. Initial tag switching architecture also contains provisions for upstream label distribution, but neither the current tag switching implementation nor the MPLS architecture needs this type of distribution method.

All label bindings are advertised immediately to all other routers through the TDP sessions. The advertisements also can be examined by means of debugging commands, as shown in Example C-11. The printout was taken on the San Francisco router after the route toward 192.168.2.0/24 was propagated from New York to San Francisco via the IGP and entered into the San Francisco LSR's routing table.

As you can see from the printout, the San Francisco router announces its IP prefix-to-label binding to all TDP neighbors, regardless of whether they are upstream or downstream. Even more, the binding also is sent to the next-hop router, so there is no split-horizon processing in TDP or LDP.

Example C-11 *IP Prefix-to-label Binding Propagation Through TDP*

```
1d20h: tagcon: adj 172.16.1.1:0 (pp 0x80EA98E4): advertise 192.168.2.0/24, tag 29
  (#12)
1d20h: tagcon: adj 172.16.3.1:0 (pp 0x80EA98E4): advertise 192.168.2.0/24, tag 29
  (#12)
1d20h: tagcon: adj 172.16.2.1:0 (pp 0x80EA98E4): advertise 192.168.2.0/24, tag 29
  (#12)
1d20h: tagcon: adj 172.16.1.2:0 (pp 0x80EA98E4): advertise 192.168.2.0/24, tag 29
  (#12)
1d20h: tagcon: adj 172.16.1.3:0 (pp 0x80EA98E4): advertise 192.168.2.0/24, tag 29
  (#12)
1d20h: tdp: Sent bind PIE to 172.16.1.1:0 (pp 0x80EA98E4)
```

The adjacent LSRs receive prefix-to-label mappings, store them in their LIB, and use them in their FIB or LFIB if the mapping has been received from their downstream neighbor, which is the next-hop for the particular FEC in question. This storage method is called *liberal retention mode* as opposed to *conservative retention mode*, where an LSR retains only the labels assigned to a prefix by its current downstream routers.

NOTE There are a number of possible combinations between the three label allocation parameters (unsolicited versus on-demand distribution, independent versus ordered control, and liberal versus conservative retention), but the routers running Cisco IOS software always use unsolicited distribution, independent control, and liberal retention over Frame-mode MPLS interfaces. The fixed set of parameters should not prevent the router from interoperating through LDP with other devices that use a different default. For more details on which combinations work and which ones don't, please refer to the IETF LDP documentation.

The **show tag-switching tdp bindings** command can display all the label mappings generated by a router or received from its TDP neighbors. Example C-12 displays the result of that command for IP prefix 192.168.2.0/24 on the San Francisco router.

Example C-12 *Label Information Base Entry on San Francisco Router*

```
SanFrancisco#show tag-switching tdp bindings 192.168.2.0
  tib entry: 192.168.2.0/24, rev 7
                  local binding:  tag: 30
                  remote binding: tsr: 172.16.1.1:0, tag: 33
                  remote binding: tsr: 172.16.1.2:0, tag: 35
                  remote binding: tsr: 172.16.1.3:0, tag: 23
                  remote binding: tsr: 172.16.2.1:0, tag: 59
                  remote binding: tsr: 172.16.3.1:0, tag: 28
SanFrancisco#
```

A router might receive TDP bindings from a number of neighbors, but uses only a few of them in the forwarding tables as follows:

- The label binding from the next-hop router is entered in the corresponding FIB entry. If the router doesn't receive the label binding from the next-hop router, the FIB entry specifies that the packets for that destination should be sent unlabeled.
- If the router receives a label binding from the next-hop router, the local label and the next-hop label are entered in the LFIB. If the next-hop router didn't assign a label to the corresponding prefix, the outgoing action in LFIB is unlabeled. Example C-13 shows both cases.

NOTE A router that has no label for a specific IP prefix from the next-hop router marks the prefix as unlabeled if it is not a directly connected interface or is not a summary route. If the route is connected directly or is a summary route, an additional Layer 3 lookup is needed and a router assigns a null label to that prefix due to a mechanism called *penultimate hop popping*, which is covered in the next section.

Example C-13 *Label Forwarding Information Base on San Francisco Router*

```
SanFrancisco#show tag forwarding-table tags 30-31
Local  Outgoing    Prefix            Bytes tag  Outgoing   Next Hop tag    tag or VC   or
  Tunnel Id      switched   interface
30     28          192.168.2.0/32    0          Se0/0/1    172.16.3.1
31     untagged    192.168.100.4/32  0          Se1/0/3    172.16.1.3
```

Convergence in a Frame-Mode MPLS Network

An important aspect in MPLS network design is the convergence time of the network. Some MPLS applications (for example, an MPLS/VPN or BGP design based on MPLS) do not work correctly unless a labeled packet can be sent all the way through from the ingress Edge-LSR to the egress Edge-LSR. In these applications, the convergence time needed by an Interior Gateway Protocol (IGP) to converge around a failure in the core network could be increased by the label propagation delay.

In a Frame-mode MPLS network, using liberal retention mode in combination with independent label control and unsolicited downstream label distribution minimizes the TDP/LDP convergence delay. Every router using liberal retention mode usually has label assignments for a given prefix from all its TDP/LDP neighbors, so it can always find a proper outgoing label following the routing table convergence without asking its new next-hop router for the label assignment.

NOTE Unfortunately the immediate TDP/LDP convergence happens only when a link fails. When a link is reestablished, the IGP adjacency and convergence usually happens before the TDP adjacency is set up and the labels are exchanged, resulting in the temporary incapability to forward labeled packets until the labels are exchanged.

The next set of examples, based on a failure scenario (the link between Washington and San Francisco fails) in the SuperNet network, illustrate the immediate convergence. The examples observe only the route toward network 192.168.100.2/32, which is attached to the New York router.

The **show** command printouts (see Example C-14) in the initial state indicate that the target route is reachable through interface Serial0/0/1 through next-hop 172.16.3.1.

Example C-14 *TDP, LFIB, and FIB Entries Prior to Link failure*

```
SanFrancisco#show tag-switching tdp binding 192.168.100.2 32
  tib entry: 192.168.100.2/32, rev 10
        local binding:  tag: 28
        remote binding: tsr: 172.16.2.1:0, tag: 28
        remote binding: tsr: 172.16.3.1:0, tag: 32

SanFrancisco#show tag-switching forwarding 192.168.100.2
Local  Outgoing    Prefix           Bytes tag  Outgoing   Next Hop tag    tag or VC   or
  Tunnel Id      switched   interface
28     32          192.168.100.2/32  0          Se0/0/1    point2point
SanFrancisco#show ip cef 192.168.100.2
192.168.100.2/32, version 76, attached
0 packets, 0 bytes
  tag information set, shared, unshareable
    local tag: 28
  via Serial0/0/1, 9 dependencies
    valid adjacency
    tag rewrite with Se0/0/1, point2point, tags imposed: {32}
```

Immediately following the link failure, the LFIB is scanned to clean up any entries that used the failed interface as the outgoing interface (see Example C-15).

Example C-15 *LFIB Scan Following a Link Failure*

```
SanFrancisco#sh debug
IP routing:
  IP routing debugging is on
Tag Switching:
  TDP Tag Information Base (TIB) changes debugging is on
  TDP tag and address advertisements debugging is on
  Cisco Express Forwarding related TFIB services debugging is on

SanFrancisco#
3d03h: %LINK-5-CHANGED: Interface Serial0/0/1, changed state to down
3d03h: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/1, changed state to down
3d03h: TFIB: fib scan start:needed:1,unres:0,mac:0,mtu:0,loadinfo:0,scans aborted 0
```

continues

Example C-15 *LFIB Scan Following a Link Failure (Continued)*

```
3d03h: TFIB: fib check cleanup for 192.168.100.2/32,index=0,return_value=0
3d03h: TFIB: fib_scanner_walk,reslve path 0 of 192.168.100.2/32
3d03h: TFIB: resolve tag rew,prefix=192.168.100.2/32,has tag_info,no parent
3d03h: TFIB: finish fib res 192.168.100.2/32:index 0,parent outg tag no parent
3d03h: TFIB: set fib rew: pfx 192.168.100.2/32,index=0,add=1,tag_rew->adj=Serial 0/
0/1
3d03h: TFIB: Update TFIB for 192.168.100.2/32, fib no loadinfo, tfib no loadinfo,
per_pkt,resolved=1
3d03h: TFIB: fib_scanner_end
```

The failed interface then is removed from the routing table and the associated routes are removed from the IP routing table. Because no alternative equal-cost route toward 192.168.100.2/32 currently exists, the route is removed completely from the routing table and the associated entry is deleted from the LFIB (see Example C-16).

Example C-16 *Routing Table and LFIB Cleanup*

```
3d03h: RT: interface Serial0/0/1 removed from routing table
3d03h: RT: delete route to 192.168.100.2 via 0.0.0.0, Serial0/0/1
3d03h: RT: no routes to 192.168.100.2, flushing
3d03h: TFIB: tfib_fib_delete,192.168.100.2/32,fib->count=1
3d03h: TFIB: fib complete delete: prefix=192.168.100.2/32,inc tag=28,del info=1
3d03h: TFIB: deactivate tag rew for 192.168.100.2/32,index=0
3d03h: TFIB: Update TFIB for 192.168.100.2/32, fib no loadinfo, tfib no loadinfo,
  per_pkt,resolved=0
3d03h: TFIB: set fib rew: pfx 192.168.100.2/32,index=0,add=0,tag_rew->adj=Serial 0/0/1
```

An alternate route to 192.168.100.2 goes through the Denver router. The OSPF process immediately installs the alternate route in the routing table. Corresponding CEF and LFIB entries are created and the LFIB entry gets the label assigned by 172.16.2.1 (the Denver router) as its outgoing label. The new LFIB entry is installed without any TDP/LDP interaction with any TDP/LDP neighbors (see Example C-17).

Example C-17 *Alternate Route Is Installed in the Routing Table*

```
3d03h: RT: add 192.168.100.2/32 via 172.16.2.1, ospf metric [110/21]
3d03h: TFIB: post table chg,ROUTE_UP 192.168.100.2/32,loadinfo ct=1
3d03h: TFIB: find_rt_tgs,192.168.100.2/32,meth 1,res_next_hop=172.16.2.1, Se0/0/2,
  next_hop 172.16.2.1
3d03h: TFIB: route tag chg 192.168.100.2/32,idx=0,inc=28,outg=28,enabled=0x1
3d03h: TFIB: create tag info 192.168.100.2/32,inc tag=28,has no info
3d03h: TFIB: resolve tag rew,prefix=192.168.100.2/32,has tag_info,no parent
3d03h: TFIB: finish fib res 192.168.100.2/32:index 0,parent outg tag no parent
3d03h: TFIB: set fib rew: pfx 192.168.100.2/32,index=0,add=1,tag_rew->adj=FastEt hernet0/0
3d03h: TFIB: Update TFIB for 192.168.100.2/32, fib no loadinfo, tfib no loadinfo,
  per_pkt,resolved=1
```

As the last step, all entries from the TDP neighbor 172.16.3.1 (the Washington router), which is no longer reachable, are removed from the Label Information Base (see Example C-18).

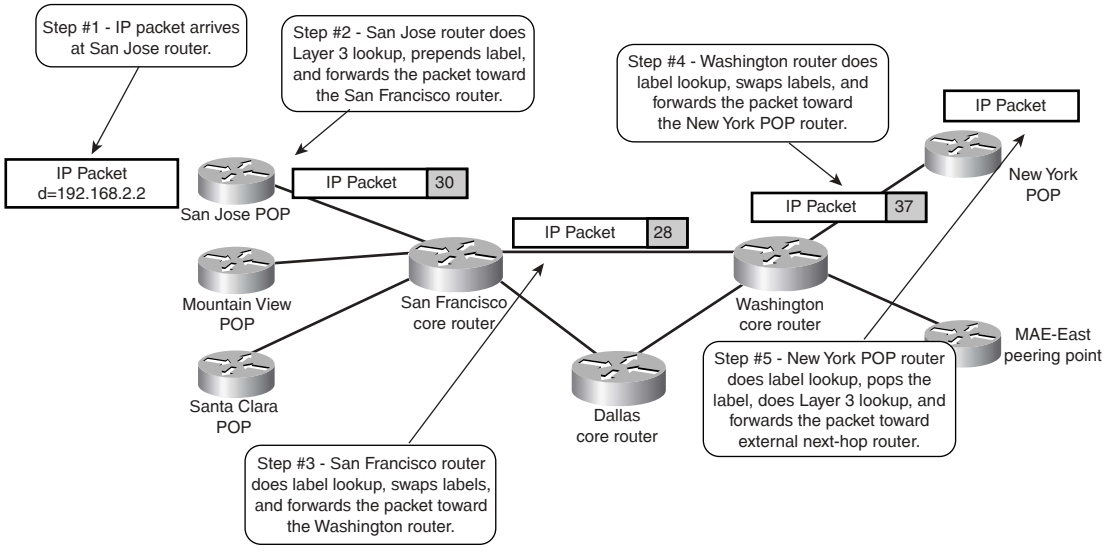
Example C-18 *LIB Entries Received from Washington Router Are Removed*

```
3d03h: tagcon: tibent(192.168.100.2/32): rem tag 1 from 172.16.3.1:0 removed
3d03h: tagcon: no route_tag_change for: 192.168.100.2/32
        for tsr 172.16.3.1:0: tsr is not next hop
3d03h: TFIB: resolve recursive: share rewrite of parent 192.168.100.2/32
```

Penultimate Hop Popping

An egress Edge-LSR in an MPLS network might have to perform two lookups on a packet received from an MPLS neighbor and destined for a subnet outside the MPLS domain. It must inspect the label in the label stack header, and it must perform the label lookup just to realize that the label has to be popped and the underlying IP packet inspected. An additional Layer 3 lookup must be performed on the IP packet before it can be forwarded to its final destination. Figure C-15 shows the corresponding process in the SuperNet network.

Figure C-15 *Double Lookup in New York POP Router*

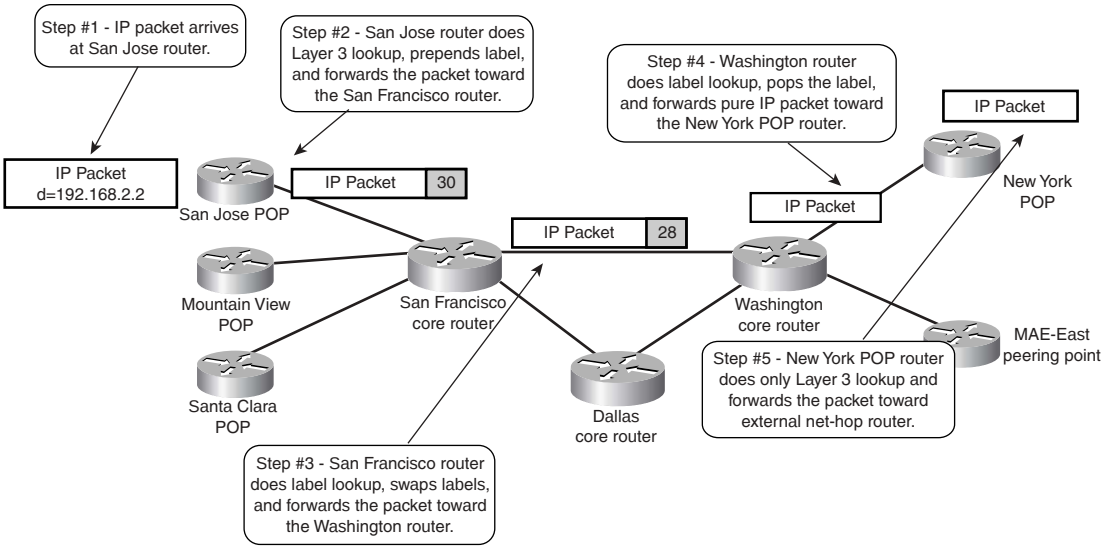


The double lookup in the NewYork POP router might reduce the performance of that node. Furthermore, in environments where MPLS and IP switching is realized in hardware, the fact that a double lookup might need to be performed can increase the complexity of the hardware implementation significantly. To address both issues, *penultimate hop popping (PHP)* was introduced into the MPLS architecture.

NOTE Penultimate hop popping is used only for directly connected subnets or aggregate routes. In the case of a directly connected interface, Layer 3 lookup is necessary to obtain the correct next-hop information for a packet that is sent toward a directly connected destination. If the prefix is an aggregate, a Layer 3 lookup also is necessary to find a more specific route that then is used to route the packet toward its correct destination. In all other cases, the Layer 2 outbound packet information is available within the LFIB and, therefore, a Layer 3 lookup is not necessary and the packet can be label switched.

With penultimate hop popping, the Edge-LSR can request a label pop operation from its upstream neighbors. In the SuperNet network, the Washington router pops the label from the packet (Step 4 in Figure C-16) and sends a pure IP packet to the New York router. Then the New York router does a simple Layer 3 lookup and forwards the packet to its final destination (Step 5 in Figure C-16).

Figure C-16 *Penultimate Hop Popping in the SuperNet Network*



Penultimate hop popping is requested through TDP or LDP by using a special label value (1 for TDP, 3 for LDP) that also is called the *implicit-null* value.

When the egress LSR requests penultimate hop popping for an IP prefix, the local LIB entry in the egress LSR and the remote LIB entry in the upstream LSRs indicate the **imp-null** value (see Example C-19) and the LFIB entry in the penultimate LSR indicates a tag pop operation (see Example C-20).

Example C-19 *LIB Entries in Edge LSR and Penultimate LSR*

```
NewYork#show tag tdp binding 192.168.2.0 24
  tib entry: 192.168.2.0/24, rev 10
       local binding:  tag: imp-null(1)
       remote binding: tsr: 172.16.3.1:0, tag: 28

Washington#show tag tdp binding 192.168.2.0 24
   tib entry: 192.168.2.0/24, rev 10
       local binding:  tag: 28
       remote binding: tsr: 172.16.3.2:0, tag: imp-null(1)
       remote binding: tsr: 172.16.1.4:0, tag: 30
       remote binding: tsr: 172.16.2.1:0, tag: 37
```

Example C-20 *LFIB Entry in Washington Router*

```
Washington#show tag forwarding tags 28
Local  Outgoing    Prefix           Bytes tag  Outgoing   Next Hop tag    tag or VC   or
  Tunnel Id      switched   interface
26     Pop tag     192.168.2.0/24    0          Se0/0/2    point2point
```

MPLS Interaction with the Border Gateway Protocol

In the section "Label Binding and Distribution," earlier in this appendix, you saw that a label is assigned to every IP prefix in the IP routing table of a router acting as LSR, the only exception being routes learned through the Border Gateway Protocol (BGP). No labels are assigned to these routes and the ingress Edge-LSR uses the label assigned to the BGP next hop to label the packets forwarded toward BGP destinations.

To illustrate this phenomenon, assume that the MAE-East router in the SuperNet network receives a route for network 192.168.3.0 from a router in Autonomous System 4635. The route is propagated throughout the SuperNet network with the MAE-East router from AS4635 being the BGP next-hop. When looking in the BGP table on the San Jose router and in the corresponding FIB table entries, you can see that the same label (28) is used to label the packets for the BGP destination and for the BGP next-hop (see Example C-21).

Example C-21 *BGP and FIB Entries on the San Jose Router*

```
SanJose#show ip bgp 192.168.3.0
BGP routing table entry for 192.168.3.0/24, version 2
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  4635
    192.168.100.2 (metric 21) from 172.16.4.1 (172.16.4.1)
      Origin IGP, metric 0, localpref 100, valid, internal, best
SanJose#show ip cef 192.168.3.0
192.168.3.0/24, version 52, cached adjacency 172.16.1.4
```

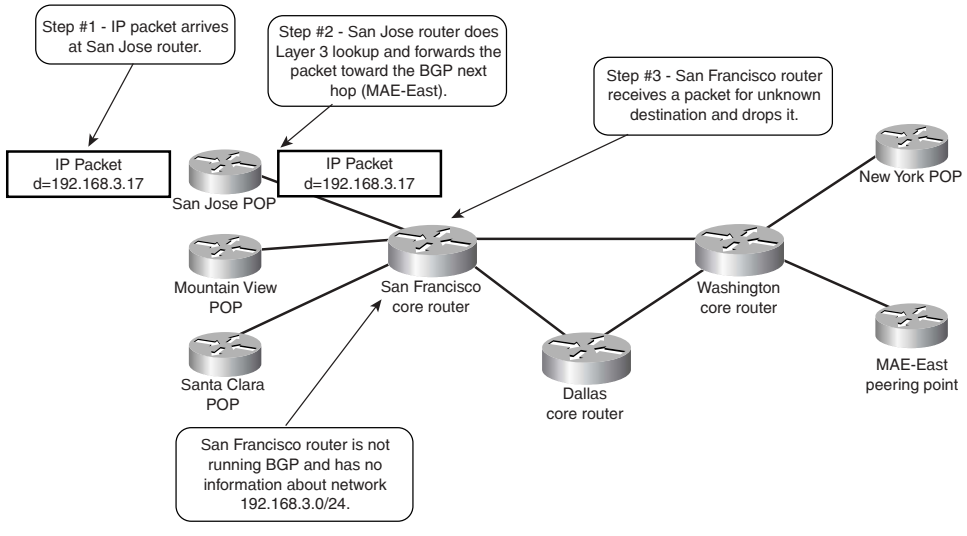
continues

Example C-21 *BGP and FIB Entries on the San Jose Router (Continued)*

```
0 packets, 0 bytes
tag information from 172.16.1.4/32, shared
    local tag: 39
    fast tag rewrite with Se1/0/1, 172.16.1.4, tags imposed: {28}
  via 192.168.100.2, 0 dependencies, recursive
    next hop 172.16.1.4, Serial1/0/1 via 172.16.1.4/32
    valid cached adjacency
  tag rewrite with Se1/0/1, 172.16.1.4, tags imposed: {28}  SanJose#show ip cef 192.168.100.2
192.168.100.2/32, version 26, cached adjacency 172.16.1.4
0 packets, 0 bytes
  tag information set, shared
    local tag: 39
    fast tag rewrite with Se1/0/1, 172.16.1.4, tags imposed: {28}
  via 192.168.100.2, 0 dependencies, recursive
    next hop 172.16.1.4, Serial1/0/1 via 172.16.1.4/32
    valid cached adjacency
    tag rewrite with Se1/0/1, 172.16.1.4, tags imposed: {28}
```

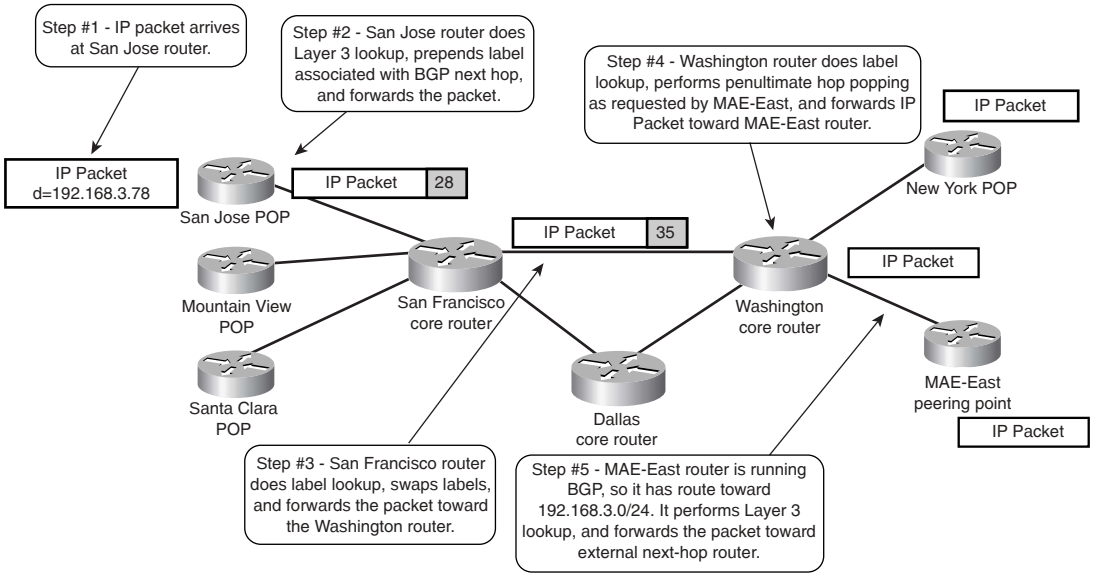
The interaction between MPLS, IGP, and BGP gives a network designer a completely new approach to network design. Traditionally, BGP had to be run on every router in the core of a service provider network to enable proper packet forwarding. For example, BGP information from MAE-East had to be propagated to every core router in the SuperNet network (Washington, Denver, and San Francisco). If that were not the case, the core routers could not route the packets toward the BGP destination, as illustrated in Figure C-17.

Figure C-17 *Connectivity Loss in Network with No BGP on Core Routers*



If, however, the SuperNet network runs MPLS, the San Jose router propagates the packet toward a BGP destination as a labeled packet with the label associated with the BGP next hop. Because the BGP next hop should always be announced in the IGP the network is running, all intermediate routers must have an incoming-to-outgoing label mapping for that destination in their LFIB already and must propagate the labeled packet toward the egress LSR (MAE-East) but need not run BGP. Figure C-18 displays the whole process.

Figure C-18 *Packet Propagation Toward BGP Destination in MPLS-enabled Network*



The removal of the BGP process from the core routers in a service provider network has a number of benefits:

- The routing tables in the core routers become much more stable because the core routers do not process route flaps in the Internet.
- Memory requirements for the core routers are reduced because they do not have to store the Internet routes (around 70,000 to 80,000 routes, consuming between 20 to 40 MB of memory in the router).
- The route processor CPU utilization on the core routers is reduced greatly because they do not have to process BGP updates.

MPLS deployment, even in a pure router-based service provider IP backbone, is therefore highly recommended.

Summary: Frame-Mode MPLS Operations

This section discusses MPLS operation over interfaces where labeled packets are sent encapsulated in Layer 2 frames (Frame-mode MPLS operation).

Label Switch Routers (LSRs) use Label Distribution Protocol (LDP, an IETF standard) or Tag Distribution Protocol (TDP, a Cisco pre-standard) to exchange IP prefix-to-label bindings. A Label Information Base (LIB, also called a Tag Information Base [TIB]) stores these bindings, which are used to build the Forwarding Information Base (FIB) entries in ingress Edge-LSRs as well as Label Forwarding Information Base (LFIB, also called Tag Forwarding Information Base [TFIB]) in all MPLS nodes. Cisco IOS supports both label distribution protocols, and you can use both in the same network, even on separate interfaces of the same LSR.

The **tag-switching ip** or **mpls ip** interface configuration command enables MPLS on a Frame-mode interface. In IOS releases supporting LDP, the desired label distribution protocol must be selected using the **mpls label-distribution** command. These commands start TDP or LDP on the specified interface. TDP/LDP finds other LSRs attached to the same subnet through TDP/LDP hello packets sent as UDP packets to broadcast or multicast IP addresses. When the neighboring LSRs are discovered, a TDP/LDP session is established using TCP as the transport protocol to ensure the reliable delivery of label mappings.

The IOS implementation of LSR on Frame-mode interfaces assigns labels to IP prefixes as soon as they appear in the routing table, even though the LSR hasn't received a corresponding label from its downstream neighbor, because it can always perform a Layer 3 lookup if needed. The router is thus working in independent control allocation mode, as opposed to ordered control allocation, where a device assigns only labels to those prefixes where a downstream label already exists in the LIB.

When running MPLS over Frame-mode interfaces, a Cisco router immediately propagates allocated labels to its TDP/LDP neighbors. This distribution method is called unsolicited downstream distribution, as opposed to downstream on demand distribution, where the upstream routers explicitly ask the downstream routers for specific labels.

A Cisco router acting as an LSR stores all label mappings received from its TDP/LDP neighbors. This storage method is called liberal retention mode as opposed to conservative retention mode where the LSR stores only labels received from its next hop downstream routers. The liberal retention mode uses more memory but enables instantaneous TDP/LDP convergence following the routing protocol convergence after a failure in the network. After the LSRs in an MPLS network have exchanged label mappings, the ingress LSR can label the incoming data packets. The ingress LSR inserts a label stack header between the Layer 2 header and the IP header. For unicast destination-only IP routing, the label stack header usually contains only one label, but the MPLS architecture also supports stacked labels used by other MPLS applications, such as traffic

engineering or Virtual Private Networks. The labeled packets are distinguished from the unlabeled IP packets by using different ethertype codes on LAN media and a different PPP Protocol field value.

Network designers usually consider MPLS only as a technology that allows seamless integration of IP routers and ATM switches or enables additional applications, such as MPLS Traffic Engineering or MPLS/VPN. They usually don't realize they can gain significant simplifications by deploying MPLS in any network that runs BGP as its exterior routing protocol. Deploying MPLS in a network running BGP allows you to remove BGP routing from core routers (non–Edge-LSRs), resulting in a network design that is more stable, requires less memory on the core routers, and prevents high CPU utilization due to BGP update processing on the core routers.

MPLS/VPN Architecture Overview

Virtual private networks (VPNs) have evolved using two major VPN models: overlay VPN and peer-to-peer VPN. The overlay VPN model, most commonly used in a service provider network, dictates that the design and provisioning of virtual circuits across the backbone must be complete prior to any traffic flow. In the case of an IP network, this means that even though the underlying technology is connectionless, it requires a connection-oriented approach to provision the service.

From a service provider's point of view, the scaling issues of an overlay VPN model are felt most when having to manage and provision a large number of circuits/tunnels between customer devices. From a customer's point of view, the Interior Gateway Protocol design is typically extremely complex and also difficult to manage.

On the other hand, the peer-to-peer VPN model suffers from lack of isolation between the customers and the need for coordinated IP address space between them.

With the introduction of Multiprotocol Label Switching (MPLS), which combines the benefits of Layer 2 switching with Layer 3 routing and switching, it became possible to construct a technology that combines the benefits of an overlay VPN (such as security and isolation among customers) with the benefits of simplified routing that a peer-to-peer VPN implementation brings. The new technology, called MPLS/VPN, results in simpler customer routing and somewhat simpler service provider provisioning, and makes possible a number of topologies that are hard to implement in either the overlay or peer-to-peer VPN models. MPLS also adds the benefits of a connection-oriented approach to the IP routing paradigm, through the establishment of label-switched paths, which are created based on topology information rather than traffic flow.

NOTE This introduction might lead you to believe that any overlay VPN implementation can be replaced with an MPLS/VPN implementation. Unfortunately, that is not true. MPLS/VPN currently supports only IP as the Layer 3 protocol. Other protocols, such as IPX and AppleTalk, still must be tunneled across an IP backbone.

The MPLS/VPN architecture provides the capability to commission an IP network infrastructure that delivers *private* network services over a *public* infrastructure. The MPLS/VPN technology is quite complex in itself and will not be fully covered in this appendix. In this section, you'll see the basic MPLS/VPN concepts without going into too many details that would clutter the overall picture.

Case Study: Virtual Private Networks in SuperCom Service Provider Network

As with all complex topics, the MPLS/VPN concepts are best explained through use of a case study. Imagine a service provider (let's call it SuperCom) that is offering VPN services based on MPLS/VPN technologies. The service provider has two points of presence (POP), a U.S. POP in the San Jose area and a French POP in the Paris area. The POPs are linked through a core router located in Washington, D.C.

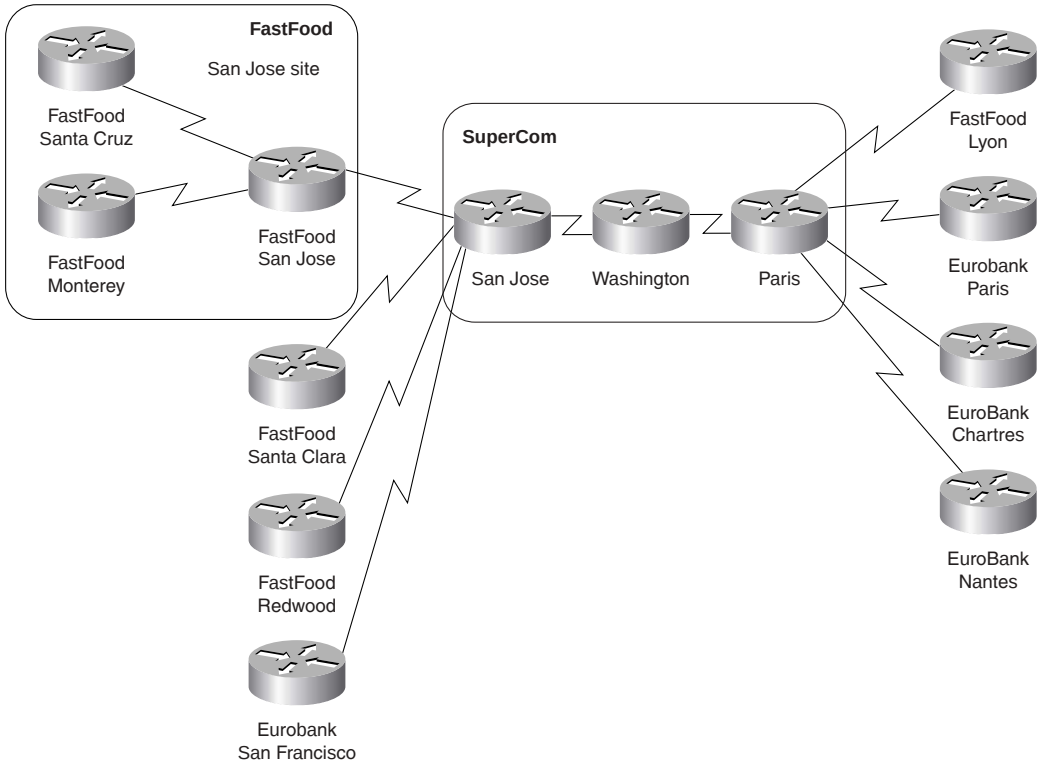
The service provider has two customers: FastFood, with headquarters in San Jose and branch offices in Santa Clara and Lyon; and EuroBank, with headquarters in Paris and branch offices in Chartres and San Francisco. The FastFood company has a number of other branch offices (for example, in Santa Cruz and Monterey) that are linked directly with the FastFood central site. The whole network is shown in Figure C-19.

The routers in Figure C-19 have the following roles:

- San Jose and Paris routers link the SuperCom network with its customers; they are thus provider edge (PE) routers.
- The Washington router does not have any customer connection; therefore, it's a provider (P) router.
- Customer routers connected to the SuperCom network—FastFood routers in San Jose, Santa Clara, and Lyon, as well as EuroBank routers in San Francisco, Paris, and Chartres—are customer edge (CE) routers.
- The FastFood routers in Santa Cruz and Monterey have no connection to the SuperCom network; they are customer (C) routers. All the networks connected directly to the FastFood San Jose site (Santa Cruz and Monterey networks) form a customer network (C-network) and

represent a single site to the SuperCom network. The service provider does not care (and does not need to know) about the internal structure of that site.

Figure C-19 *SuperCom Network and Its Customers*



Let's assume that both companies, FastFood and EuroBank, follow the same addressing convention—the central sites use public IP addresses, whereas all the remote sites use private IP address space (network 10.0.0.0).

NOTE The addressing scheme used by these corporations is seen more often in real customer networks, more so in cases in which the customer didn't acquire a significant portion of public IP address space several years ago.

The IP addresses used by these two companies are summarized in Table C-5.

Table C-5 *Address Space of FastFood and EuroBank*

Company	Site	Subnet
FastFood	San Jose	195.12.2.0/24
	Santa Clara	10.1.1.0/24
	Redwood	10.1.2.0/24
	Santa Cruz	10.1.3.0/24
	Monterey	10.1.4.0/24
	Lyon	10.2.1.0/24
EuroBank	Paris	196.7.25.0/24
	Chartres	10.2.1.0/24
	Nantes	10.2.2.0/24
	San Francisco	10.1.1.0/24

The SuperCom service provider would like to offer IP-based VPN service based on the peer-to-peer model (not a number of IP-over-IP tunnels), but it cannot do so easily because the address space of sites connected to the same router overlap.

NOTE The service provider would encounter a similar (but not so obvious) problem if the address space overlap occurred between customers connected to different POPs. The traditional peer-to-peer model requires strict uniqueness of IP address space.

SuperCom can traditionally solve the overlapping addresses issue in three ways:

- It can persuade the customers to renumber their networks. Most customers would not be willing to do that and would rather find another service provider.
- It can implement the VPN service with IP-over-IP tunnels, where the customer IP addresses are hidden from the service provider routers.
- It can implement a complex network address translation (NAT) scheme that would translate customer addresses into a different (but unique) set of addresses at the provider edge router and then translate those addresses back to the customer addresses before the packet would be sent from the egress PE-router to the CE router. Although such a solution is technically feasible, the administrative overhead is prohibitively large.

VPN Routing and Forwarding Tables

The overlapping addresses, usually resulting from usage of private IP addresses in customer networks, are one of the major obstacles to successful deployment of peer-to-peer VPN implementations. The MPLS/VPN technology provides an elegant solution to the dilemma: Each VPN has its own routing and forwarding table in the router, so any customer or site that belongs to that VPN is provided access only to the set of routes contained within that table. Any PE-router in an MPLS/VPN network thus contains a number of per-VPN routing tables and a global routing table that is used to reach other routers in the provider network, as well as external globally reachable destinations (for example, the rest of the Internet). Effectively, a number of virtual routers are created in a single physical router, as displayed in Figure C-20 for the case of San Jose router of SuperCom network.

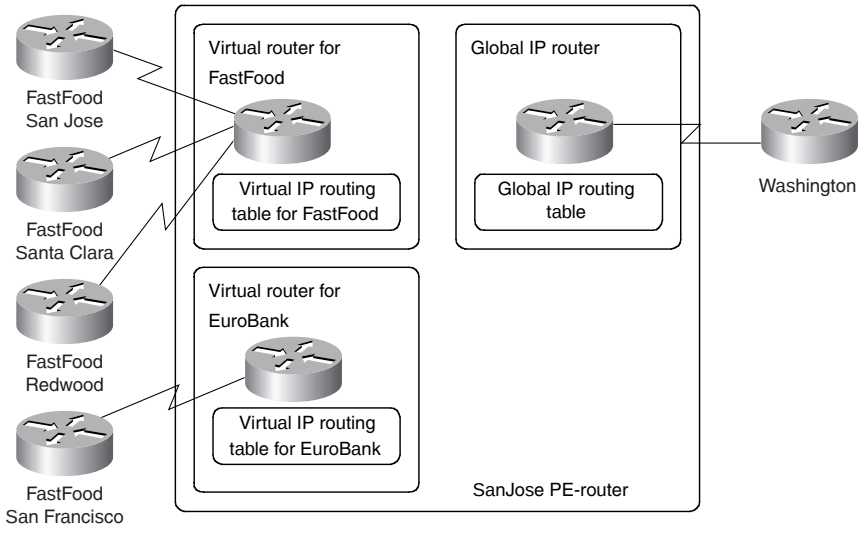
NOTE The relationship between virtual private networks and VPN routing and forwarding tables as explained in the previous paragraph is a slight simplification of the actual relationship between these two concepts. Nevertheless, it is true in cases where each site (or customer) belongs only to one VPN. The additional complexity introduced by overlapping VPNs or sites belonging to more than one VPN is explained in the section "Overlapping Virtual Private Networks," later in this section.

The concept of virtual routers allows the customers to use either global or private IP address space in each VPN. Each customer site belongs to a particular VPN, so the only requirement is that the address space be unique within that VPN. Uniqueness of addresses is not required among VPNs except where two VPNs that share the same private address space want to communicate.

More structures are associated with each virtual router than just the virtual IP routing table:

- A forwarding table that is derived from the routing table and is based on CEF technology.
- A set of interfaces that use the derived forwarding table.
- Rules that control the import and export of routes from and into the VPN routing table. These rules were introduced to support overlapping VPNs and are explained later in this section.
- A set of routing protocols/peers, which inject information into the VPN routing table. This includes static routing.
- Router variables associated with the routing protocol that is used to populate the VPN routing table.

Figure C-20 *Virtual Routers Created in a PE-router*



The usage of these structures is explained in the rest of this section.

The combination of the VPN IP routing table and associated VPN IP forwarding table is called VPN routing and forwarding instance (VRF).

NOTE You might think that there is no difference between an IP routing table and an IP forwarding table—and usually that's true. In an MPLS environment, the only minor difference between them is the fact that the IP forwarding table also contains MPLS encapsulation information. A major difference between the two tables arises in cases where an IP route refers to a next hop that is not directly connected. In that case, the routing table will contain the next-hop information, but not the outgoing interface or the IP address of the downstream router. The forwarding table will contain all the information needed to forward the packet toward the destination. For example, with the configuration in Example C-22, the routing table lists the next hop for network 10.0.0.0/8 as 1.0.0.1 (as shown in Example C-23), while the forwarding table contains the real next hop (the IP address of the downstream router), as shown in Example C-24.

Example C-22 *Sample Configuration with Recursive IP Routing*

```
ip route 10.0.0.0 255.0.0.0 1.0.0.1
ip route 1.0.0.1 255.255.255.255 2.0.0.2
!
interface serial 0
ip address 2.0.0.1 255.0.0.0
```

Example C-23 *IP Routing Table for the Recursive IP Routing Example*

```
mpls router#  show ip route
...
     1.0.0.0/32 is subnetted, 1 subnets
S       1.0.0.1 [1/0] via 2.0.0.2
C    2.0.0.0/8 is directly connected, Serial0
S    10.0.0.0/8 [1/0] via 1.0.0.1
...
```

Example C-24 *CEF Forwarding Table Entry for Recursive IP Routing Example*

```
mpls router#  show ip cef 10.0.0.0

10.0.0.0/8, version 87
0 packets, 0 bytes
  via 1.0.0.1, 0 dependencies, recursive
  next hop  2.0.0.2, Serial0 via 1.0.0.1/32
```

In the SuperCom case, the San Jose router contains three IP routing and forwarding tables—one table per customer and a global table used to forward non-VPN IP packets and to route VPN packets between PE-routers.

Overlapping Virtual Private Networks

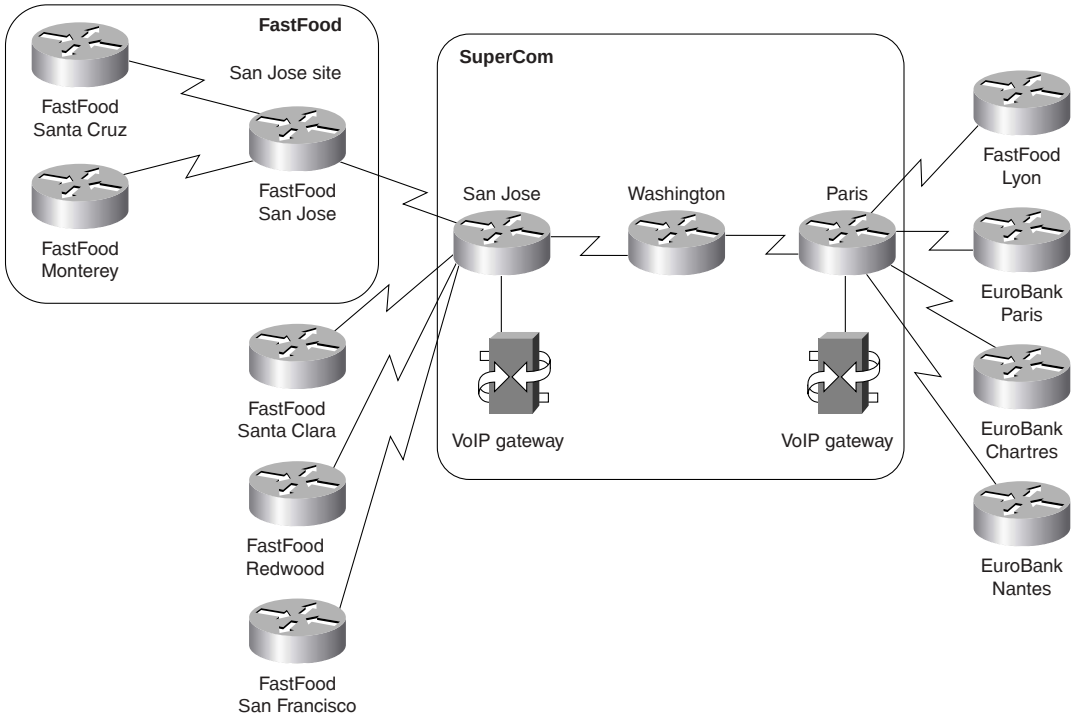
The SuperCom example might lead you to believe that a VPN is associated with a single VRF in a PE-router. Although that would be true in the case where the VPN customer needs no connectivity with other VPN customers, the situation might become more complex and require more than one VRF per VPN customer.

Imagine that SuperCom wants to extend its service offering with a Voice over IP (VoIP) service with gateways to the public voice network located in San Jose and Paris, as shown in Figure C-21. The VoIP gateways were placed in a separate VPN to enhance the security of the newly created service. The IP addresses of these gateways are shown in Table C-6.

Table C-6 *IP Addresses of VoIP Gateways in SuperCom Network*

VoIP Gateway Location	VoIP Gateway IP Address
San Jose	212.15.23.12
Paris	212.15.27.35

Figure C-21 *VoIP Gateways in SuperCom Network*

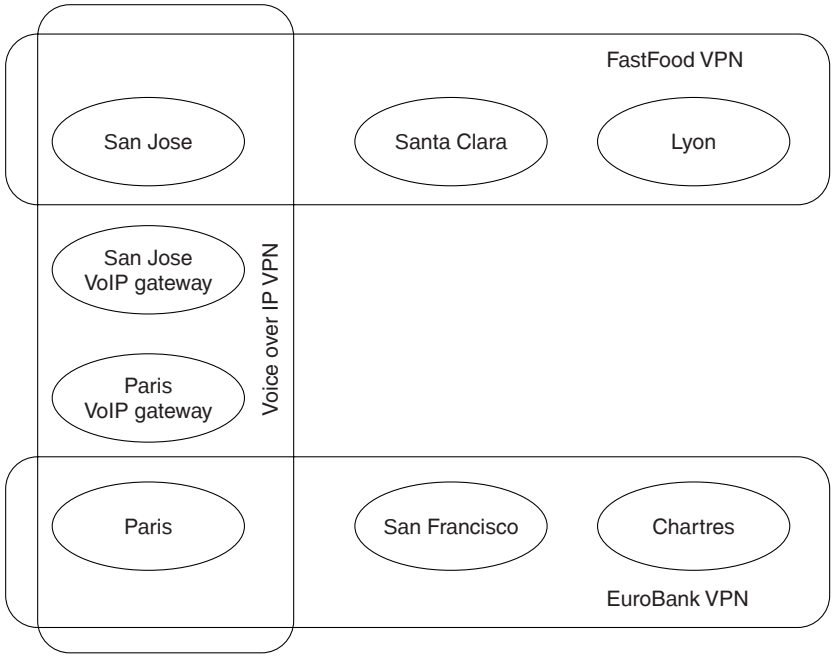


Both EuroBank and FastFood decided to use the service, but only from their central sites— the branch offices have no need for international voice connectivity. This requirement leads to an interesting problem: The central sites of both organizations need to be in two VPNs: the corporate VPN to reach their remote sites and the VoIP VPN to reach the VoIP gateways. The connectivity requirements are illustrated in Figure C-22.

NOTE The connectivity requirements in Figure C-22 are a simplification of the requirements that you would encounter in a real service provider network. Most often, for security reasons, the customers using a common service (for example, VoIP gateways) will not see each other, but only the gateways or servers providing the service that they are using.

To support connectivity requirements similar to those in Figure C-22, the MPLS/VPN architecture supports the concept of *sites*, where a VPN is made up of one or multiple sites. A VPN is essentially a collection of sites sharing common routing information, which means that a site may belong to more than one VPN if it holds routes from separate VPNs. This provides the capability to build intranets and extranets. A VPN in the MPLS/VPN architecture can therefore be pictured as a community of interest or a closed user group, which is dictated by the routing visibility that the site will have.

Figure C-22 *VPN Connectivity Requirements in SuperCom Network*



The VRF concept introduced in the previous section must be modified to support the concept of sites that can reside in more than one VPN. For example, the central site of FastFood and EuroBank cannot use the same VRF as all other FastFood or EuroBank sites connected to the same PE-router. The central site of EuroBank, for example, needs to access the VoIP gateways, so the routes toward these gateways must be in the VRF for that site, whereas the same routes will not be in the Chartres' site VRF. Therefore, the MPLS/VPN architecture unbundles the concept of VRF from the concept of VPN. The VRF is simply a collection of routes that should be available to a particular site (or set of sites) connected to a PE-router. These routes can belong to more than one VPN.

NOTE You might be inclined at this moment to jump from a one-VPN-one-VRF model to the other extreme: one-site-one-VRF model. Although that model is theoretically correct and supports any VPN topology, it leads to more complex configurations of the PE-routers that are harder to maintain and that also use more memory. Therefore, it is recommended to keep the number of VRFs to a minimum (for example, one VRF for the customer's central site and another VRF for all remote offices connected to the same PE-router).

The relationship between the VPNs, sites, and VRFs can be summarized in the following rule, which should be used as the basis for any VRF definition in an MPLS/VPN network.

NOTE All sites that share the same routing information (usually this means that they belong to the same set of VPNs), that are allowed to communicate directly with each other, and that are connected to the same PE-router can be placed in a common VRF.

Using this rule, the minimum set of VRFs in the SuperCom network is the one outlined in Table C-7.

Table C-7 *VRFs in the PE-routers in the SuperCom Network*

PE-router	VRF	Sites in the VRF	VRF Belongs to VPNs
San Jose	FastFood_Central	FastFood SanJose site	FastFood
	VoIP FastFood	FastFood Santa Clara site	FastFood
		FastFood Redwood site	
	EuroBank	EuroBank San Francisco site	EuroBank
	VoIP	San Jose VoIP gateway	VoIP
Paris	FastFood	FastFood Lyon site	EuroBank
	EuroBank_Central	EuroBank Paris site	EuroBank, VoIP
	EuroBank	EuroBank Chartres site	EuroBank
		EuroBank Nantes site	
	VoIP	Paris VoIP gateway	VoIP

Route Targets

A careful reader might start asking an interesting question: If there is no one-to-one mapping between VPN and VRF, how does the router know which routes need to be inserted into which VRF? This dilemma is solved by the introduction of another concept in the MPLS/VPN architecture: the *route target*. Every VPN route is tagged with one or more route targets when it is exported from a VRF (to be offered to other VRFs). You can also associate a set of route targets with a VRF, and all routes tagged with at least one of those route targets will be inserted into the VRF.

NOTE The *route target* is the closest approximation to a *VPN identifier* in the MPLS/VPN architecture. In most VPN topologies, you can equate them, but in other topologies (usually a central services topology), a single VPN might need more than one route target for successful implementation.

NOTE The route target is a 64-bit quantity. For simplicity reasons, we will use names for route targets in this appendix.

The SuperCom network contains three VPNs and thus requires three route targets. The association between route targets and VRFs in the SuperCom network is outlined in Table C-8.

Table C-8 *Correspondence Between VRFs and Route Targets in SuperCom Network*

PE-router	VRF	Sites in the VRF	Route Target Attached to Exported Routes	Import Route Targets
San Jose	FastFood_Central	FastFood SanJose site	FastFood, VoIP	FastFood, VoIP
	FastFood	FastFood Santa Clara site	FastFood	FastFood
		FastFood Redwood site		
	EuroBank	EuroBank San Francisco site	EuroBank	EuroBank
	VoIP	San Jose VoIP gateway	VoIP	VoIP
Paris	FastFood	FastFood Lyon site	FastFood	FastFood
	EuroBank_Central	EuroBank Paris site	EuroBank, VoIP	EuroBank, VoIP
	EuroBank	EuroBank Chartres site	EuroBank	EuroBank
		EuroBank Nantes site		
	VoIP	Paris VoIP gateway	VoIP	VoIP

NOTE Based on Table C-8, you might assume that the route targets attached to routes exported from a VRF always match the set of import route targets of a VRF. Although that's certainly true in simpler VPN topologies, there are widespread VPN topologies (for example, central services VPN) in which this assumption is not true.

Propagation of VPN Routing Information in the Provider Network

The previous sections have explained MPLS/VPN architecture from a single PE-router standpoint. Two issues have yet to be addressed:

- How will the PE-routers exchange information about VPN customers and VPN routes between themselves?
- How will the PE-routers forward packets originated in customer VPNs?

This section addresses inter-PE routing; the next section briefly describes the forwarding mechanism.

Two fundamentally different ways exist for approaching the VPN route exchange between PE-routers:

- The PE-routers could run a different routing algorithm for each VPN. For example, a copy of OSPF or EIGRP could be run for each VPN. This solution would face serious scalability problems in service provider networks with a large number of VPNs. It would also face interesting design challenges when asked to provide support for overlapping VPNs.
- The PE-routers run a single routing protocol to exchange all VPN routes. To support overlapping address spaces of VPN customers, the IP addresses used by the VPN customers must be augmented with additional information to make them unique.

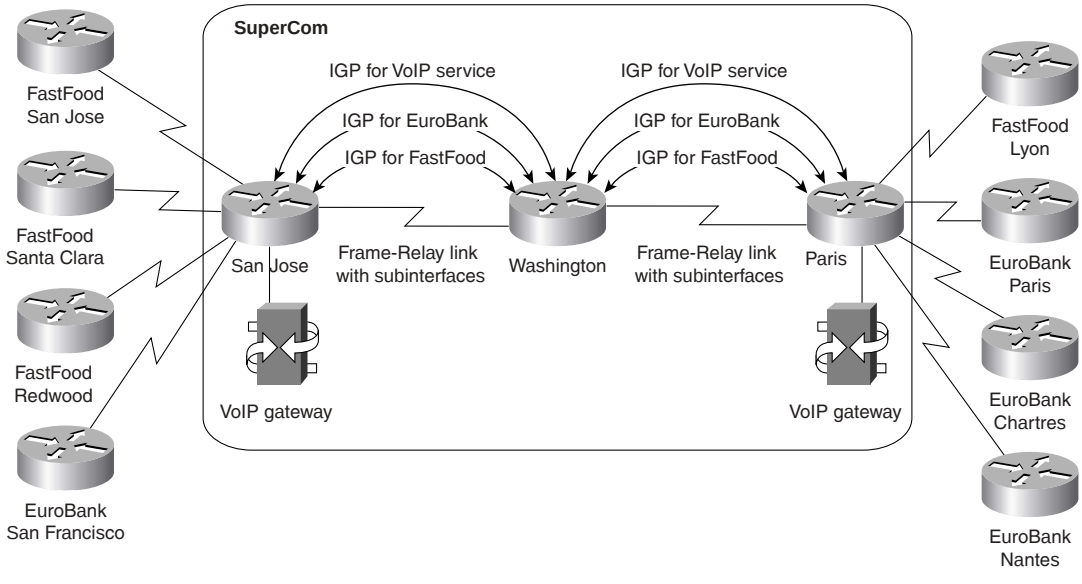
NOTE To illustrate the scalability issues that might arise from deploying one routing algorithm per VPN, consider the case where the SuperCom network would have to support more than 100 VPN customers connected to the San Jose and Paris routers with OSPF as the routing protocol. The PE-routers in the SuperCom network would run more than 100 independent copies of OSPF routing process (if that were technically possible), with each copy sending hello packets and periodic refreshments over the network. Because you cannot run more than one copy of OSPF over the same link, you would have to configure per-VPN subinterfaces (for example, using Frame Relay encapsulation) on the link between San Jose (or Paris) and Washington, resulting in an extremely complex network similar to the one shown in Figure C-23. You would also have to run 100 different SPF algorithms and maintain 100 separate topology databases in the service provider routers.

The second approach was chosen as the building block of MPLS/VPN technology. IP subnets advertised by the CE-routers to the PE-routers are augmented with a 64-bit prefix called a *route distinguisher* to make them unique. The resulting 96-bit addresses are then exchanged between the PE-routers using a special address family of Multiprotocol BGP (hereby referred to as MP-BGP).

There were several reasons for choosing BGP as the routing protocol used to transport VPN routes:

- The number of VPN routes in a network can become very large. BGP is the only routing protocol that can support a very large number of routes.
- BGP, EIGRP, and IS-IS are the only routing protocols that are multiprotocol by design (all of them can carry routing information for a number of different address families). IS-IS and EIGRP, however, do not scale to the same number of routes as BGP. BGP is also designed to exchange information between routers that are not directly connected. This BGP feature supports keeping VPN routing information out of the provider core routers (P-routers).
- GP can carry any information attached to a route as an optional BGP attribute. What's more, you can define additional attributes that will be transparently forwarded by any BGP router that does not understand them. This property of BGP makes propagation of route targets between PE-routers extremely simple.

Figure C-23 *SuperCom Network with One IGP per VPN*



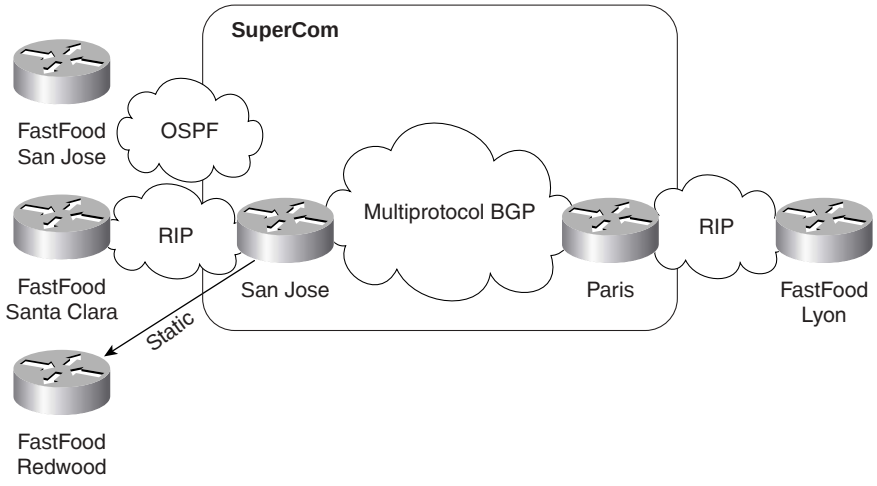
Multiprotocol BGP in the SuperCom Network

To illustrate the interaction of per-VPN routing protocols with the MP-BGP used in the service provider network core, consider the case of the FastFood customer in the SuperCom network. Let's assume that the San Jose site is using OSPF to interact with the SuperCom backbone, the Lyon and Santa Clara sites are using RIP, and the Redwood site is using no routing protocol—there

is a static route configured on the San Jose PE-router and the default route configured on the Redwood router. The routing protocols used in FastFood VPN are shown in Figure C-24.

NOTE The Washington router (the P-router in the SuperCom network) is not involved in the MP- BGP. As you'll see in the next section, the forwarding model used in MPLS/VPN does not require the P-routers to make any routing decisions based on VPN addresses; they just forward packets based on the label value attached to the packet. The P-routers, therefore, do not need to carry the VPN routes, resulting in even better scalability.

Figure C-24 *Routing Protocols Used in FastFood VPN*



The San Jose PE-router collects routing information from the San Jose site using a per-VPN OSPF process. Similarly, the information from the Santa Clara site is collected using a per-VPN RIP process. This process is marked as Step 1 in Figure 8-7.

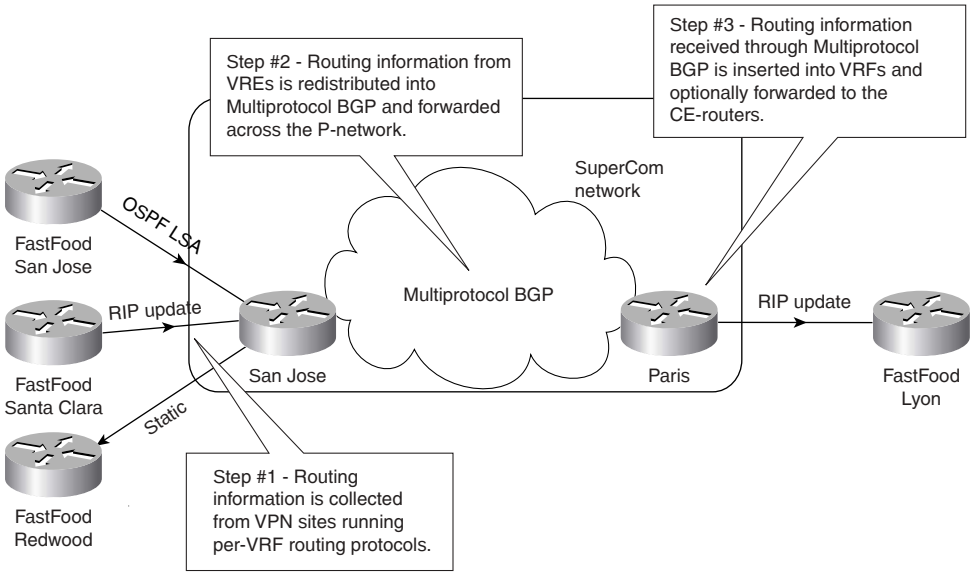
NOTE The routing protocol used within a VPN network must be limited to the VPN in question. If the same routing protocol would be used in different VPNs, the possibility of using overlapping IP addresses between VPNs would be lost, and there would be potential route leakage between VPNs.

To support overlapping VPNs, the routing protocol must be limited to a single VPN routing and forwarding (VRF) table. Each PE-router must be configured so that any routing information learnt from an interface can be associated with a particular VRF. This is done through the standard routing protocol process and is known as the *routing context*. A separate routing context is used per VRF.

Some routing protocols (for example, RIP) support several instances (or routing contexts) of the same protocol, with each instance running in a different VRF. Other protocols (for example, OSPF) require a separate copy of the routing protocol process for each VRF.

The information gathered by various routing protocols in the San Jose PE-router, as well as the static routes configured on the San Jose router, is redistributed into MP-BGP. VPN addresses are augmented with the route distinguishers at the moment of redistribution. The route export route target specified in the originating VRF is also attached to the route. The resulting 96-bit routing information is propagated by MP-BGP to the Paris router (Step 2 in Figure C-25).

Figure C-25 *Routing Protocol Operation in SuperCom Network*



WARNING The redistribution of the per-VPN routing information into MP-BGP is not automatic and must be manually configured on the router for each VRF, unless this information was learned from the customer via BGP. The omission of manual redistribution into MP-BGP is one of the most common configuration errors in MPLS/VPN deployment.

The Paris router, after receiving MP-BGP routes, inserts the received routes into various VRF tables based on the route target attribute attached to each individual route. The route distinguisher is dropped from the 96-bit route when the route is inserted into the VRF, resulting yet again in a traditional IP route. Finally, the routing information received through BGP is redistributed into the RIP process and is passed on to the Lyon site through RIP updates (Step 3 in Figure C-25).

WARNING Similar to the redistribution of VRF routes into MP-BGP, the redistribution of routes received over the service provider backbone back into the per-VRF routing process is not automatic, unless this process is BGP; it must be manually configured if the redistribution is required by the routing design.

Contrary to the traditional BGP operation in which the internal BGP routes are not allowed to be redistributed into other routing protocols, this restriction is lifted in the MPLS/VPN environment. The VPN routes received by a PE-router through an internal MP-BGP session from another PE-router can be redistributed into other routing protocols.

VPN Packet Forwarding

In the previous section, you saw that the IP addresses used within a VPN must be prepended with a 64-bit prefix called a route distinguisher (RD) to make them unique.

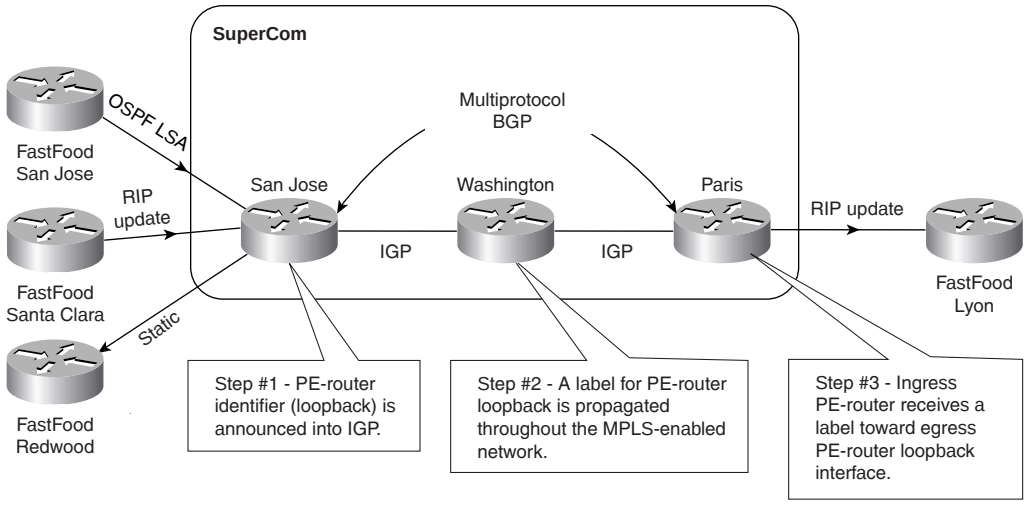
Similarly, when the VPN-originated IP packets are forwarded across the service provider backbone (the P-network), they must be augmented to make them uniquely recognizable. Yet again, several technology options are possible:

- The IP packet is rewritten to include 96-bit addresses in the packet header. This operation would be slow and complex.
- The IP packet is tunneled across the network in VPN-over-IP tunnels. This choice would make MPLS/VPN as complex as traditional IP-over-IP VPN solutions using the overlay VPN model.

With the introduction of MPLS, a third technology option was made possible: Each VPN packet is labeled by the ingress PE-router with a label uniquely identifying the egress PE-router, and is sent across the network. All the routers in the network subsequently switch labels without having to look into the packet itself. The preparatory steps for this process are illustrated in Figure C-26.

Each PE-router needs a unique identifier (a host route—usually the loopback IP address is used), which is then propagated throughout the P-network using the usual IGP (Step 1). This IP address is also used as the BGP next-hop attribute of all VPN routes announced by the PE-router. A label is assigned in each P-router for that host route and is propagated to each of its neighbors (Step 2). Finally, all other PE-routers receive a label associated with the egress PE-router through an MPLS label distribution process (Step 3). After the label for the egress PE-router is received by the ingress PE-router, the VPN packet exchange can start.

Figure C-26 *PN Packet Forwarding—Preparatory Steps*



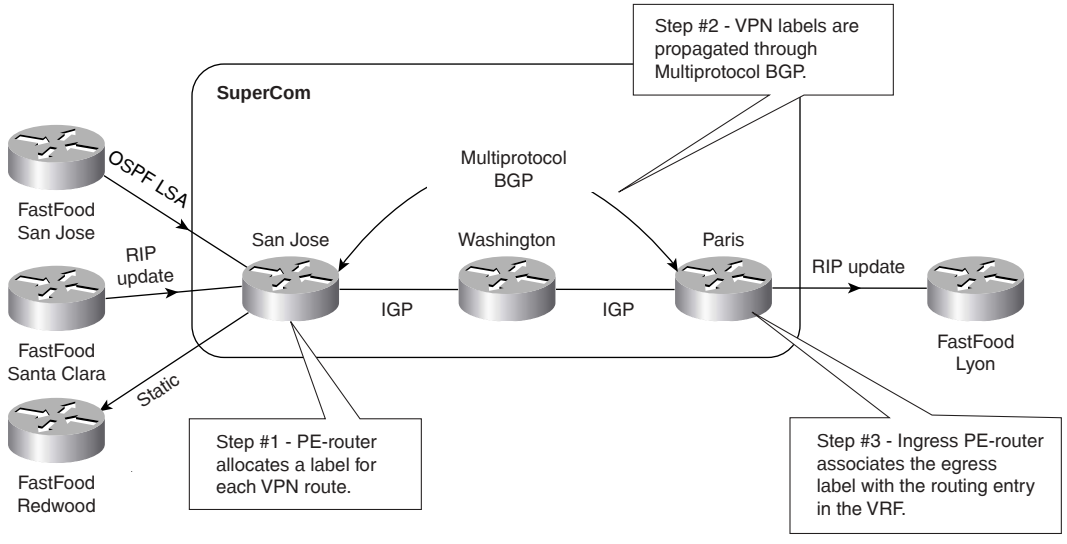
However, when the egress PE-router receives the VPN packet, it has no information to tell it which VPN the packet is destined for. To make the communication between VPN sites unique, a second set of labels is introduced, as illustrated in Figure C-27.

Each PE-router allocates a unique label for each route in each VPN routing and forwarding (VRF) instance (Step 1). These labels are propagated together with the corresponding routes through MP-BGP to all other PE-routers (Step 2). The PE-routers receiving the MP- BGP update and installing the received routes in their VRF tables (see Figure C-25 for additional details) also install the label assigned by the egress router in their VRF tables. The MPLS/VPN network is now ready to forward VPN packets.

When a VPN packet is received by the ingress PE-router, the corresponding VRF is examined, and the label associated with the destination address by the egress PE-router is fetched. Another label, pointing toward the egress PE-router, is obtained from the global forwarding table. Both labels are combined into an MPLS label stack, are attached in front of the VPN packet, and are sent toward the egress PE-router.

All the P-routers in the network switch the VPN packet based only on the top label in the stack, which points toward the egress PE-router. Because of the normal MPLS forwarding rules, the P-routers never look beyond the first label and are thus completely unaware of the second label or the VPN packet carried across the network.

Figure C-27 *VPN Label Allocation*



The egress PE-router receives the labeled packet, drops the first label, and performs a lookup on the second label, which uniquely identifies the target VRF and sometimes even the outgoing interface on the PE-router. A lookup is performed in the target VRF (if needed), and the packet is sent toward the proper CE-router.

NOTE The egress PE-router assigns labels to VPN routes in such a way that the need for additional Layer 3 lookup in the target VRF is minimized. The additional Layer 3 lookup is needed only for summary VPN routes advertised between the PE-routers.

The router just before the egress PE-router might also remove the first label in the label stack through *penultimate hop popping*.

In the best case (no summary VPN routes and network topology that supports penultimate hop popping), the egress PE-router would perform only a single label lookup, resulting in maximum forwarding performance.

Summary: MPLS/VPN Architecture Overview

Virtual Private Networks (VPN) based on Multiprotocol Label Switching (MPLS) combine the benefits of the overlay VPN model, such as isolation and security, with the benefits of the peer-to-peer VPN model, such as simplified routing, easier provisioning, and better scalability. A number of mechanisms are needed to successfully meet all these goals:

- Each VPN needs a separate VPN routing and forwarding instance (VRF) in each PE-router to guarantee isolation and enable usage of uncoordinated private IP addresses.

- To support overlapping VPN topologies, the VRFs can be more granular than the VPNs and can participate in more than one VPN at a time. An attribute called a route target is needed to identify the set of VPNs in which a particular VRF participates. For maximum flexibility, a set of route targets can be associated with a VRF or attached to a VPN route.
- VPN IP addresses are prepended with 64-bit route distinguishers to make VPN addresses globally unique. These 96-bit addresses are exchanged between the PE-routers through MP-BGP, which also carries additional route attributes (for example, the route target) by means of optional BGP route attributes, called extended communities.
- Each PE-router needs a unique router ID (host route—usually the loopback address) that is used to allocate a label and enable VPN packet forwarding across the backbone.
- Each PE-router allocates a unique label to each route in each VRF (even if they have the same next hop) and propagates these labels together with 96-bit VPN addresses through MP-BGP.
- Ingress PE-routers use a two-level MPLS label stack to label the VPN packets with a VPN label assigned by the egress PE-router and an IGP label identifying the PE-router assigned through the regular MPLS label distribution mechanisms. The label stack is prepended to the VPN packet, and the resulting MPLS packet is forwarded across the P-network.

APPENDIX D

Decimal to Binary Conversion Table

Decimal Value	Binary Value	Decimal Value	Binary Value
0	0000 0000	23	0001 0111
1	0000 0001	24	0001 1000
2	0000 0010	25	0001 1001
3	0000 0011	26	0001 1010
4	0000 0100	27	0001 1011
5	0000 0101	28	0001 1100
6	0000 0110	29	0001 1101
7	0000 0111	30	0001 1110
8	0000 1000	31	0001 1111
9	0000 1001	32	0010 0000
10	0000 1010	33	0010 0001
11	0000 1011	34	0010 0010
12	0000 1100	35	0010 0011
13	0000 1101	36	0010 0100
14	0000 1110	37	0010 0101
15	0000 1111	38	0010 0110
16	0001 0000	39	0010 0111
17	0001 0001	40	0010 1000
18	0001 0010	41	0010 1001
19	0001 0011	42	0010 1010
20	0001 0100	43	0010 1011
21	0001 0101	44	0010 1100
22	0001 0110	45	0010 1101

continues

(Continued)

Decimal Value	Binary Value	Decimal Value	Binary Value
46	0010 1110	76	0100 1100
47	0010 1111	77	0100 1101
48	0011 0000	78	0100 1110
49	0011 0001	79	0100 1111
50	0011 0010	80	0101 0000
51	0011 0011	81	0101 0001
52	0011 0100	82	0101 0010
53	0011 0101	83	0101 0011
54	0011 0110	84	0101 0100
55	0011 0111	85	0101 0101
56	0011 1000	86	0101 0110
57	0011 1001	87	0101 0111
58	0011 1010	88	0101 1000
59	0011 1011	89	0101 1001
60	0011 1100	90	0101 1010
61	0011 1101	91	0101 1011
62	0011 1110	92	0101 1100
63	0011 1111	93	0101 1101
64	0100 0000	94	0101 1110
65	0100 0001	95	0101 1111
66	0100 0010	96	0110 0000
67	0100 0011	97	0110 0001
68	0100 0100	98	0110 0010
69	0100 0101	99	0110 0011
70	0100 0110	100	0110 0100
71	0100 0111	101	0110 0101
72	0100 1000	102	0110 0110
73	0100 1001	103	0110 0111
74	0100 1010	104	0110 1000
75	0100 1011	105	0110 1001

(Continued)

Decimal Value	Binary Value	Decimal Value	Binary Value
106	0110 1010	136	1000 1000
107	0110 1011	137	1000 1001
108	0110 1100	138	1000 1010
109	0110 1101	139	1000 1011
110	0110 1110	140	1000 1100
111	0110 1111	141	1000 1101
112	0111 0000	142	1000 1110
113	0111 0001	143	1000 1111
114	0111 0010	144	1001 0000
115	0111 0011	145	1001 0001
116	0111 0100	146	1001 0010
117	0111 0101	147	1001 0011
118	0111 0110	148	1001 0100
119	0111 0111	149	1001 0101
120	0111 1000	150	1001 0110
121	0111 1001	151	1001 0111
122	0111 1010	152	1001 1000
123	0111 1011	153	1001 1001
124	0111 1100	154	1001 1010
125	0111 1101	155	1001 1011
126	0111 1110	156	1001 1100
127	0111 1111	157	1001 1101
128	1000 0000	158	1001 1110
129	1000 0001	159	1001 1111
130	1000 0010	160	1010 0000
131	1000 0011	161	1010 0001
132	1000 0100	162	1010 0010
133	1000 0101	163	1010 0011
134	1000 0110	164	1010 0100
135	1000 0111	165	1010 0101

continues

(Continued)

Decimal Value	Binary Value	Decimal Value	Binary Value
166	1010 0110	196	1100 0100
167	1010 0111	197	1100 0101
168	1010 1000	198	1100 0110
169	1010 1001	199	1100 0111
170	1010 1010	200	1100 1000
171	1010 1011	201	1100 1001
172	1010 1100	202	1100 1010
173	1010 1101	203	1100 1011
174	1010 1110	204	1100 1100
175	1010 1111	205	1100 1101
176	1011 0000	206	1100 1110
177	1011 0001	207	1100 1111
178	1011 0010	208	1101 0000
179	1011 0011	209	1101 0001
180	1011 0100	210	1101 0010
181	1011 0101	211	1101 0011
182	1011 0110	212	1101 0100
183	1011 0111	213	1101 0101
184	1011 1000	214	1101 0110
185	1011 1001	215	1101 0111
186	1011 1010	216	1101 1000
187	1011 1011	217	1101 1001
188	1011 1100	218	1101 1010
189	1011 1101	219	1101 1011
190	1011 1110	220	1101 1100
191	1011 1111	221	1101 1101
192	1100 0000	222	1101 1110
193	1100 0001	223	1101 1111
194	1100 0010	224	1110 0000
195	1100 0011	225	1110 0001

(Continued)

Decimal Value	Binary Value	Decimal Value	Binary Value
226	1110 0010		
227	1110 0011		
228	1110 0100		
229	1110 0101		
230	1110 0110		
231	1110 0111		
232	1110 1000		
233	1110 1001		
234	1110 1010		
235	1110 1011		
236	1110 1100		
237	1110 1101		
238	1110 1110		
239	1110 1111		
240	1111 0000		
241	1111 0001		
242	1111 0010		
243	1111 0011		
244	1111 0100		
245	1111 0101		
246	1111 0110		
247	1111 0111		
248	1111 1000		
249	1111 1001		
250	1111 1010		
251	1111 1011		
252	1111 1100		
253	1111 1101		
254	1111 1110		
255	1111 1111		

GLOSSARY

224.0.0.5 The All OSPF Routers multicast IP address, listened for by all OSPF routers.

224.0.0.6 The All OSPF DR Routers multicast IP address, listened for by DR and BDR routers.

2Way (OSPF) A neighbor state that signifies the other router has reached neighbor status, having passed the parameter check.

802.11a A wireless LAN physical layer that operates at up to 54-Mbps data rates using OFDM in the 5-GHz band.

802.11b A wireless LAN physical layer that operates at up to 11-Mbps data rates using DSSS in the 2.4-GHz band.

802.11g A wireless LAN physical layer that is backward compatible with 802.11b and operates at up to 54-Mbps data rates using OFDM in the 2.4-GHz band.

802.11n A wireless LAN physical layer that is currently (as of publication) under development that will offer data rates in the hundreds of megabits per second.

802.1Q The IEEE standardized protocol for VLAN trunking.

802.1Q-in-Q A mechanism in which VLAN information can extend over another set of 802.1Q trunks by tunneling the original 802.1Q traffic with another 802.1Q tag. It allows a service provider to support transparent VLAN services with multiple customers, even if the customers use overlapping VLAN numbers.

AAA *See* Authentication, authorization, and accounting.

ABR *See* Area Border Router.

Access Control Entry An individual line in an ACL.

Access Control Server A term referring generically to a server that performs many AAA functions. It also refers to the software product Cisco Secure Access Control Server.

access link In Frame Relay, a link between a router and a Frame Relay switch.

access rate (AR) The speed at which the access link is clocked. This choice affects the price of the connection and many aspects of traffic shaping and policing, compression, quality of service, and other configuration options.

ACE *See* Access Control Entry.

Ack (EIGRP) An EIGRP message that is used to acknowledge reliable EIGRP messages, namely Update, Query, and Reply messages. Acks do not require an Ack.

ACS *See* Access Control Server.

active (EIGRP) A state for a route in an EIGRP topology table that indicates that the router is actively sending Query messages for this route, attempting to validate and learn the current best route to that subnet.

active mode FTP Defines a particular behavior for FTP regarding the establishment of data TCP connections. In active mode, the FTP client uses the FTP PORT command, over the FTP control connection, to tell the FTP server the port on which the client should be listening for a new data connection. The server uses well-known port 20, and initiates a TCP connection to the FTP client's earlier-declared port.

active scanning Each 802.11 station periodically sends a probe request frame on each RF channel and monitors probe response frames that all access points within range send back. Stations use the signal strength of the probe response frames to determine which access point or ad hoc network to associate with.

actual queue depth The actual number of packets in a queue at a particular time.

ad hoc mode A wireless LAN that only includes wireless users and no access points. 802.11 data frames in an ad hoc network travel directly between wireless users.

adaptive shaping A Frame Relay Traffic Shaping feature during which the shaping rate is reduced when the shaper notices congestion through the receipt of BECN or ForeSight messages.

adjacency (EIGRP) Often used synonymously with neighbor, but with emphasis on the fact that all required parameters match, allowing routing updates to be exchanged between the routers.

adjacency table A table used by CEF that holds information about adjacent IP hosts to which packets can be forwarded.

adjacent (OSPF) Any OSPF neighbor for which the database flooding process has completed.

adjacent-layer interaction On a single computer, one layer provides a service to a higher layer. The software or hardware that implements the higher layer requests that the next lower layer perform the needed function.

administrative scoping Controls the distribution of multicast traffic for the private multicast address range 239.0.0.0 to 239.255.255.255 by configuring a filter and applying it on the interfaces.

administrative weight A Cisco-proprietary BGP feature. The administrative weight can be assigned to each NLRI and path locally on a router, impacting the local router's choice of the best BGP routes. The value cannot be communicated to another router.

administratively scoped addresses The range 239.0.0.0 through 239.255.255.255 that IANA has assigned for use in private multicast domains.

advertised window *See* receiver's advertised window.

AES Advanced Encryption Standard. A superior encryption mechanism that is part of the 802.11i standard and has much stronger security than TKIP.

AF *See* Assured Forwarding.

aggregatable global unicast address An IPv6 address format used for publicly registered IPv6 addresses.

aggregate route Another term for summary route.

AGGREGATOR An optional transitive BGP path attribute that, for a summary route, lists the BGP RID and ASN of the router that created the summary.

AIS Alarm Indication Signal. With T1s, the practice of sending all binary 1s on the line in reaction to problems, to provide signal transitions and allow recovery of synchronization and framing.

All OSPF DR Routers The multicast IP address 224.0.0.6, listened for by DR and BDR routers.

All OSPF Routers The multicast IP address 224.0.0.5, listened for by all OSPF routers.

Alternate state An 802.1w RSTP port state in which the port is not the Root Port but is available to become the Root Port if the current Root Port goes down.

AMI Alternate Mark Inversion. A serial-line encoding standard that sends alternating positive and negative 3-volt signals for binary 1, and no signal (0 V) for binary 0.

area (OSPF) A contiguous group of data links that share the same OSPF area number.

Area Border Router An OSPF router that connects to the backbone area and to one or more non-backbone area.

ARP Address Resolution Protocol. Defined in RFC 826, a protocol used on LANs so that an IP host can discover the MAC address of another device that is using a particular IP address.

AS number (ASN) A number between 1 and 64,511 (public) and 64,512 and 65,535 (private) assigned to an AS for the purpose of proper BGP operation.

AS_CONFED_SEQ A type of AS_PATH segment consisting of an ordered list of confederation sub-ASNs through which a route has been advertised.

AS_CONFED_SET A type of AS_PATH segment consisting of an unordered list of confederation sub-ASNs consolidated from component subnets of a summary BGP route created inside a confederation.

AS_PATH A BGP path attribute that lists ASNs through which the route has been advertised. The AS_PATH includes four types of segments: AS_SEQ, AS_SET, AS_CONFED_SEQ, and AS_CONFED_SET. Often, this term is used synonymously with AS_SEQ.

AS_PATH access list A Cisco IOS configuration tool, using the **ip as-path access-list** command, that defines a list of statements that match the AS_PATH BGP path attribute using regular expressions.

AS_PATH length A calculation of the length of the AS_PATH PA, which includes 1 for each number in the AS_SEQ, 1 for an entire AS_SET segment, and possibly other considerations.

AS_PATH prepending This term has two BGP-related definitions. First, it is the normal process in which a router, before sending an Update to an eBGP peer, adds its local ASN to the beginning of the AS_PATH path attribute. Second, it is the routing policy of purposefully adding one or more ASNs to the beginning of a route's AS_PATH path attribute, typically to lengthen the AS_PATH and make the route less desirable in the BGP decision process.

AS_SEQUENCE A type of AS_PATH segment consisting of an ordered list of ASNs through which the route has been advertised.

AS_SET A type of AS_PATH segment consisting of an unordered list of ASNs consolidated from component subnets of a summary BGP route.

ASBR Autonomous System Boundary Router. An OSPF router that redistributes routes from some other source into OSPF.

Assert message Sent by a PIM-DM or PIM-SM router when it receives a multicast packet for a group on a LAN interface that is in the outgoing interface list for the group; includes the administrative distance of the unicast routing protocol used to learn the network of the source with its metric value.

association ID When a wireless station connects to an access point, the access point assigns an association ID (AID) to the station. Various protocols, such as power-save mode, make use of the association ID.

Assured Forwarding A set of DiffServ PHBs that defines 12 DSCP values, with four queuing classes and three drop probabilities within each queuing class.

ATOMIC_AGGREGATE A well-known discretionary BGP path attribute that flags a route as being a summary route.

authentication With routing protocols, the process by which the router receiving a routing update determines if the routing update came from a trusted router.

Authentication, authorization, and accounting Three core security functions.

authentication method A term referring generically to ways in which a router or switch can determine whether a particular device or user should be allowed access.

authentication server In 802.1X, the computer that stores usernames/passwords and verifies that the correct values were submitted before authenticating the user.

authenticator The 802.1X function implemented by a switch, in which the switch translates between EAPoL and RADIUS messages in both directions, and enables/disables ports based on the success/failure of authentication.

auto-negotiation Ethernet process by which devices attached to the same cable negotiate their speed and the duplex settings over the cable.

autonomous system In BGP, a set of routers inside a single administrative authority, grouped together for the purpose of controlling routing policies for the routes advertised by that group to the Internet.

Auto-RP Auto-Rendezvous Point. Cisco-proprietary protocol that can be used to designate an RP and send RP-Announce messages that advertise its IP address and groups. Also, it can be used to designate a mapping agent that interprets what IP address RP is advertising and for what groups. A mapping agent sends this information in the RP-Discovery messages so that all PIM-SM routers can learn the IP address of the RP and groups it is supporting automatically.

average queue depth Calculated measurement based on the actual queue depth and the previous average. Designed to allow WRED to adjust slowly to rapid changes of the actual queue depth.

B8ZS *See* Bipolar 8 Zero Substitution.

backbone area (OSPF) Area 0; the area to which all other OSPF areas much connect in order for OSPF to work.

BackboneFast Cisco-proprietary STP feature in which switches use messaging to confirm the loss of Hello BPDUs in a switch's Root Port, to avoid having to wait for maxage to expire, resulting in faster convergence.

Backup state An 802.1w RSTP port state in which the port is an alternative Designated Port on some LAN segment.

Backup Designated Router In OSPF, a router that is prepared to take over the Designated Router.

Backward Explicit Congestion Notification A bit inside the Frame Relay header that, when set, implies that congestion occurred in the direction opposite (or backward) as compared with the direction of the frame.

Bc *See* Committed Burst.

Bc bucket Jargon used to refer to the first of two buckets in the dual token bucket model; its size is Bc.

BDR *See* Backup Designated Router.

Be *See* Excess Burst.

Be bucket Jargon used to refer to the second of two buckets in the dual token bucket model; its size is Be.

beacon An 802.11 frame that access points or stations in ad hoc networks send periodically so that wireless stations can discover the presence of a wireless LAN and coordinate use of certain protocols, such as power-save mode.

BECN *See* Backward Explicit Congestion Notification.

BGP *See* Border Gateway Protocol.

BGP decision process A set of rules by which BGP examines the details of multiple BGP routes for the same NLRI and chooses the single best BGP route to install in the local BGP table.

BGP table A table inside a router that holds the path attributes and NLRI known by the BGP implementation on that router.

BGP Update A BGP message that includes withdrawn routes, path attributes, and NLRI.

Bipolar 8 Zero Substitution A serial-line encoding standard that substitutes Bipolar Violations in a string of eight binary 0s to provide enough signal transitions to maintain synchronization.

Bipolar Violation For some encoding schemes, consecutive signals must use opposite polarity in an effort to reduce DC current. A BPV occurs when consecutive signals are of the same polarity.

Blocking state An 802.1D STP port state in which the port does not send or receive frames, except for listening for received Hello BPDUs.

boot field The low-order 4 bits of the configuration register. These bits direct a router to load either ROMMON software (boot field 0x0), RXBOOT software (boot field 0x1), or a full-function IOS image.

Boot Protocol A standard (RFC 951) protocol by which a LAN-attached host can dynamically broadcast a request for a server to assign it an IP address, along with other configuration settings, including a subnet mask and default gateway IP address.

BOOTP *See* Boot Protocol.

Border Gateway Protocol An exterior routing protocol designed to exchange prefix information between different autonomous systems. The information includes a rich set of characteristics called path attributes, which in turn allows for great flexibility regarding routing choices.

BPDU Guard Cisco-proprietary STP feature in which a switch port monitors for STP BPDUs of any kind, err-disabling the port upon receipt of any BPDU.

BPV *See* Bipolar Violation.

broadcast address Ethernet MAC address that represents all devices on the LAN.

broadcast domain A set of all devices that receive broadcast frames originating from any device within the set. Devices in the same VLAN are in the same broadcast domain.

broadcast subnet When subnetting a class A, B, or C address, the subnet for which all subnet bits are binary 1.

CB Marking *See* Class-Based Marking.

CBWFQ *See* class-based weighted fair queuing.

CDPCP CDP Control Protocol. The portion of PPP focused on supporting the CDP protocol.

CEF *See* Cisco Express Forwarding.

Cell Loss Priority A bit in the ATM cell header that, when set to 1, means that if a device needs to discard frames, it should discard the frames with DE 1 first.

CGMP *See* Cisco Group Management Protocol.

Challenge Handshake Authentication Protocol An Internet standard authentication protocol that uses secure hashes and a three-way handshake to perform authentication over a PPP link.

CHAP *See* Challenge Handshake Authentication Protocol.

CIDR *See* Classless interdomain routing.

CIR *See* Committed information rate.

Cisco Express Forwarding An optimized Layer 3 forwarding path through a router or switch. CEF optimizes routing table lookup by creating a special, easily searched tree structure based on the contents of the IP routing table. The forwarding information is called the Forwarding Information Base (FIB), and by caching adjacency information is called the adjacency table.

Cisco Group Management Protocol A Cisco-proprietary feature. After a Cisco multicast router receives IGMP Join or Leave messages from hosts, it communicates to the connected Cisco switches, telling them which hosts (based on their unicast MAC addresses) have joined or left each multicast group. Switches examine their CAM tables and determine on which ports these hosts are connected and either forward multicast traffic or stop forwarding on those ports only.

Cisco Structured Wireless-Aware Network A framework for integrating wired and wireless networks based on the Cisco product line of wireless LAN products.

Cisco SWAN *See* Cisco Structured Wireless-Aware Network.

Cisco WDS *See* Cisco Wireless Domain Services.

Cisco Wireless Domain Services A set of Cisco IOS Software features that enhances and simplifies wireless LAN client mobility, security, deployment, and management.

Class-Based Marking An MQC-based feature of IOS that is used to classify and mark packets for QoS purposes.

class-based weighted fair queuing A Cisco IOS queuing tool that uses MQC configuration commands and reserves a minimum bandwidth for each queue.

class map A term referring to the MQC **class-map** command and its related subcommands, which are used for classifying packets.

Class of Service A 3-bit field in an ISL header used for marking frames. Also, used generically to refer to either the ISL CoS field or the 802.1Q User Priority field.

Class Selector A DiffServ PHB that defines eight values that provide backward compatibility with IP Precedence.

classful IP addressing A type of logic for how a router uses a default route. A convention for discussing and thinking about IP addresses by which class A, B, and C default network prefixes (of 8, 16, and 24 bits, respectively) are considered.

classful routing A type of logic for how a router uses a default route. When a default route exists, and the class A, B, or C network for the destination IP address does not exist in the routing table, the default route is used. If any part of that classful network exists in the routing table, but the packet does not match any existing subnet of that classful network, the packet does not match the default route and thus is discarded.

classless IP addressing A convention for IP addresses in which class A, B, and C default network prefixes (of 8, 16, and 24 bits, respectively) are ignored.

Classless interdomain routing Defined in RFCs 1517–1520, a scheme to help reduce Internet routing table sizes by administratively allocating large blocks of consecutive classful IP network numbers to ISPs for use in different global geographies. CIDR results in large blocks of networks that can be summarized, or aggregated, into single routes.

classless routing A type of logic for how a router uses a default route. When a default route exists, and no more specific match is made between the destination of the packet and the routing table, the default route is used.

Clear To Send On a serial cable, the pin lead set by the DCE to tell the DTE that the DTE is allowed send data.

client tracking Records client authentication and roaming events, which are sent to the CiscoWorks Wireless LAN Solution Engine (WLSE) to monitor client associations to specific access points.

CLP *See* Cell Loss Priority.

CLUSTER_LIST An optional nontransitive BGP path attribute that lists the route reflector cluster IDs through which a route has been advertised, as part of a loop-prevention process similar to the AS_PATH attribute.

collision domain A set of all devices for which any frame sent by one of the devices would collide with any frames transmitted at the same time by any of the other devices in the set.

Committed Burst With shaping, the number of bits allowed to be sent every Tc. Also defines the size of the token bucket when Be = 0.

Committed information rate In shaping and policing, commonly used to refer to the shaping or policing rate. For WAN services, a common reference to the bit rate defined in the WAN service business contract for each VC.

Common Spanning Tree A single instance of STP that is applied to multiple VLANs, typically when using the 802.1Q trunking standard.

COMMUNITY An optional transitive BGP path attribute used to store 32-bit decimal values. Used for flexible grouping of routes by assigning the group the same COMMUNITY value. Other routers can apply routing policies based on the COMMUNITY value. Used in a large number of BGP applications.

community VLAN With private VLANs, a secondary VLAN in which the ports can send and receive frames with each other, but not with ports in other secondary VLANS.

component route A term used in this book to refer to a route that is included in a larger summary route.

confederation A BGP feature that overcomes the requirement of a full mesh of iBGP peers inside a single AS by separating the AS into multiple sub-autonomous systems.

confederation ASN The ASN assigned to a confederation sub-AS.

confederation eBGP peer A BGP peer connection between two routers inside the same ASN, but in different confederation sub-autonomous systems.

confederation identifier In an IOS confederation configuration, the confederation identifier is the actual ASN as seen by eBGP peers.

configuration register A 16-bit number set with a router **config-register** command. It is used to set several low-level features related mainly to accessing the router and what the router does when powered on.

conform A category used by a policer to classify packets relative to the traffic contract. The bit rate implied by all conforming packets is within the traffic contract.

Congestion Avoidance A method for how a TCP sender grows its calculated CWND variable, thereby growing the allowed window for the connection. Congestion Avoidance grows CWND linearly.

congestion window A mechanism used by TCP senders to limit the dynamic window for a TCP connection, to reduce the sending rate when packet loss occurs. The sender considers both the advertised window size and CWND, using the smaller of the two.

control plane In IP routing, a term referring to the building of IP routing tables by IP routing protocols.

CoS *See* Class of Service.

counting to infinity A type of routing protocol convergence event in which the metric for a route increases slightly over time because of the advertisement of an invalid route.

cross-over cable Copper cable with RJ-45 connectors in which a twisted pair at pins 1,2 on the first end of the cable is connected to pins 3,6 on the other end, with a second pair connected to pins 3,6 on the first end and pins 1,2 on the other end.

CS *See* Class Selector.

CSMA/CD Carrier sense multiple access with collision detection. A media-access mechanism where devices ready to transmit data first check the channel for a carrier. If no carrier is sensed for a specific period of time, a device can transmit. If two devices transmit simultaneously, a collision occurs and is detected by all colliding devices. This collision subsequently causes each device to delay retransmissions of the collided frame for some random length of time.

CST *See* Common Spanning Tree.

CTS *See* Clear To Send.

custom queuing (CQ) A Cisco IOS queuing tool most notable for its reservation of a minimum bandwidth for each queue.

CWND *See* congestion window.

D4 framing Another name for Superframe.

DAI *See* Dynamic ARP Inspection.

Data Carrier Detect On a serial cable, the pin lead set by the DCE to imply a working link.

Data communications equipment DCE devices are one of two devices on either end of a communications circuit, specifically the device with more control over the communications. Frame Relay switches are DCE devices. DCEs are also known as data circuit-terminating equipment (DTE).

Database Description A type of OSPF packet used to exchange and acknowledge LSA headers. Sometimes called DBD.

Data-link connection identifier A Frame Relay address used in Frame Relay headers to identify the VC

data plane In IP routing, a term referring to the process of forwarding packets through a router.

Data Set Ready On a serial cable, the pin lead set by the DCE to imply that the DCE is ready to signal using pin leads

Data terminal equipment From one perspective, DTE devices are one of two devices on either end of a communications circuit, specifically the device with less control over the communications. In Frame Relay, routers connected to a Frame Relay access link are DTE devices.

Data Terminal Ready On a serial cable, the pin lead set by the DTE to imply that the DTE is ready to signal using pin leads.

DCD *See* Data Carrier Detect.

DCE *See* Data communications equipment.

DD *See* Database Description.

DE *See* Discard Eligible.

Dead Time/Interval With OSPF, the timer used to determine when a neighboring router has failed, based on a router not receiving any OSPF messages, including Hellos, in this timer period.

default route A route that is used for forwarding packets when the packet does not match any more specific routes in the IP routing table.

dense-mode protocol A multicast routing protocol whose default action is to flood multicast packets throughout a network.

Designated Port With Spanning Tree Protocol, the single port on each LAN segment from which the best Hello BPDU is forwarded.

designated router With PIM on a multiaccess network, the PIM router with the highest IP address on the subnet. With OSPF, the OSPF router that wins an election amongst all current neighbors. The DR is responsible for flooding on the subnet, and for creating and flooding the type 2 LSA for the subnet.

DHCP *See* Dynamic Host Configuration Protocol.

DHCP snooping A switch feature in which the switch examines DHCP messages and, for untrusted ports, filters all messages typically sent by servers and inappropriate messages sent by clients. It also builds a DHCP snooping binding table that is used by DAI and IP Source Guard.

DHCP snooping binding database The list of entries learned by the switch DHCP snooping feature. The entries include the MAC address used as the device's DHCP client address, the assigned IP address, the VLAN, and the switch port on which the DHCP assignment messages flowed.

Differentiated Services A set of QoS RFCs that redefines the IP header's ToS byte, and suggests specific settings of the DSCP field and the implied QoS actions based on those settings.

Differentiated Services Code Point The first 6 bits of the DS field, used for QoS marking.

differentiated tail drop A term relating to Cisco LAN switch tail-drop logic, in which multiple tail-drop thresholds may be assigned based on CoS or DSCP, resulting in some frames being discarded more aggressively than others.

DiffServ *See* Differentiated Services.

Diffusing Update Algorithm A term referring to EIGRP's internal processing logic.

Digital Signal Level 0 Inside telcos' original TDM hierarchy, the smallest unit of transmission at 64 kbps.

Digital Signal Level 1 Inside telcos' original TDM hierarchy, a unit that combines multiple DS0s into a single channel— 24 DS0s (plus overhead) for a T1, and 30 (plus overhead) for an E1.

Digital Signal Level 3 Inside telcos' original TDM hierarchy, a unit that combines multiple DS1s into a single channel— 28 DS1s (plus overhead) for a T3, and 16 E1 DS1s (plus overhead) for an E3.

Dijkstra Alternate name for the SPF algorithm, named for its inventor, Edsger W. Dijkstra.

Direct sequence spread spectrum A type of spread spectrum that spreads RF signals over the frequency spectrum by representing each data bit by a longer code. 802.11b specifies the use of DSSS.

Disabled state An 802.1D STP port state in which the port has been administratively disabled.

Discard Eligible A bit in the Frame Relay header that, when set to 1, means that if a device needs to discard frames, it should discard the frames with DE 1 first.

Discarding state An 802.1w RSTP port state in which the port is not forwarding or receiving; covers 802.1D port states disabled, blocking, and listening.

distance vector The underlying algorithms associated with RIP.

Distance Vector Multicast Routing Protocol Operates in dense mode and depends on its own unicast routing protocol that is similar to RIP to perform its multicast functions.

distributed coordination function The mandatory contention-based 802.11 access protocol that is also referred to as CSMA/CA.

distribution list A Cisco IOS configuration tool for routing protocols by which routing updates may be filtered.

DLCI *See* Data-link connection identifier.

downstream router The router that will receive the group traffic when a multicast router forwards group traffic to another router.

DR *See* Designated router.

DR election (OSPF) The process by which neighboring OSPF routers examine their Hello messages and elect the DR. The decision is based on priority (highest), or RID (highest) if priority is a tie.

DROther The term to describe a router that is neither the DR nor the BDR on a subnet that elects a DR and BDR.

DS field The second byte of the IP header, formerly known as the ToS byte and redefined by DiffServ.

DS0 *See* Digital Signal Level 0.

DS1 *See* Digital Signal Level 1.

DS3 *See* Digital Signal Level 3.

DSCP *See* Differentiated Services Code Point.

DSCP-to-CoS map A mapping between each DSCP value and a corresponding CoS value, often used in Cisco LAN switches when performing classification for egress queuing.

DSCP-to-threshold map A mapping between each DSCP value and a WRED threshold, often used in Cisco LAN switches when performing WRED.

DSR *See* Data Set Ready.

DSSS *See* Direct sequence spread spectrum.

DTE *See* Data terminal equipment.

DTIM interval The number of beacons that governs how often multicast frames are sent over a wireless LAN.

DTP *See* Dynamic Trunking Protocol.

DTR *See* Data Terminal Ready.

DUAL *See* Diffusing Update Algorithm.

Dual FIFO A Cisco IOS interface software queue queuing strategy implemented automatically when using either form of Frame Relay fragmentation. The system then interleaves packets from the high-priority queue between fragments of the medium-priority queue.

dual stack An IPv6 migration strategy in which a host or router supports both IPv4 and IPv6 natively.

dual token bucket A conceptual model used by CB Policing when using an excess burst.

dual-rate, three-color policer Policing in which two rates are metered, and packets are placed into one of three categories (conform, exceed, or violate).

DVMRP *See* Distance Vector Multicast Routing Protocol.

Dynamic ARP Inspection A switch feature with which the switch watches ARP messages, determines if those messages may or may not be part of some attack, and filters those that look suspicious.

Dynamic Host Configuration Protocol A standard (RFC 2131) protocol by which a host can dynamically broadcast a request for a server to assign to it an IP address, along with other configuration settings, including a subnet mask and default gateway IP address. DHCP provides a great deal of flexibility and functionality compared with RARP and BOOTP.

Dynamic Trunking Protocol A Cisco-proprietary protocol used to dynamically negotiate whether the devices on an Ethernet segment want to form a trunk and, if so, which type (ISL or 802.1Q).

E1 A name used for DS1 lines inside the European TDM hierarchy.

E1 route (OSPF) An OSPF external route for which internal OSPF cost is added to the cost of the route as it was redistributed into OSPF.

E2 route (OSPF) An OSPF external route for which internal OSPF cost is not added to the cost of the route as it was redistributed into OSPF.

E3 A name used for DS3 lines inside the European TDM hierarchy.

EAP *See* Extensible Authentication Protocol.

EAP over LAN The encapsulation of EAP messages directly inside LAN frames. This encapsulation is used between the supplicant and the authenticator.

EAPoL *See* EAP over LAN.

eBGP *See* External BGP.

eBGP multihop A BGP feature that defines the IP TTL field value in packets sent between two eBGP peers. This feature is required when using IP addresses other than the interface IP address on the link between peers.

EF *See* Expedited Forwarding.

EGP *See* Exterior Gateway Protocol.

ELMI *See* Enhanced Local Management Interface.

enable password The password required by the **enable** command. Also, this term may specifically refer to the password defined by the **enable password** command.

enable secret The MD5-encoded password defined by the **enable secret** command.

encapsulation The process of taking a protocol data unit (PDU) from some other source and placing a header in front of the original PDU, and possibly a trailer behind it.

encoding The process of changing the electrical characteristics on a transmission medium, based on defined rules, to represent data.

enhanced editing The Cisco IOS feature by which special short key sequences can be used to move the cursor inside the current command line to more easily change a command.

Enhanced Local Management Interface A Cisco-proprietary LMI protocol, implemented in Cisco WAN switches and routers, through which the switch can inform the router about parameters for each VC, including CIR, Bc, and Be.

ESF Extended Superframe. An enhanced version of T1 framing, as compared with the earlier Superframe (D4) standard.

established A BGP neighbor state in which the BGP neighbors have stabilized and can exchange routing information using BGP Update messages.

EUI-64 A specification for the 64-bit interface ID in an IPv6 address, composed of the first half of a MAC address, hex FFFE, and the last half of the MAC.

exceed A category used by a policer to classify packets relative to the traffic contract. With two-color policers, these packets are considered to be above the contract; for three-color, these packets are above the Bc setting, but within the Be setting.

Excess Burst With shaping and policing, the number of additional bits that may be sent after a period of relative inactivity.

expedite queue A term used with Cisco LAN switches, referring to a queue treated with strict-priority scheduling.

Expedited Forwarding A DiffServ PHB, based on DSCP EF (decimal 46), that provides low-latency queuing behavior as well as policing protection to prevent EF traffic from starving queues for other types of traffic.

Extensible Authentication Protocol Defined in RFC 3748, the protocol used by IEEE 802.1X for exchanging authentication information.

Exterior Gateway Protocol An exterior routing protocol that predates BGP. It is no longer used today.

exponential weighting constant Used by WRED to calculate the rate at which the average queue depth changes as compared with the current queue depth. The larger the number, the slower the change in the average queue depth.

External BGP A term referring to how a router views a BGP peer relationship, in which the peer is in another AS.

external route From the perspective of one routing protocol, a route that was learned by using route redistribution.

external route A characteristic of a route, as defined by a particular routing protocol, that means that the route was learned by that routing protocol through the route redistribution process.

Fast Secure Roaming Enables a wireless client to securely roam between access points in the same subnet or between subnets with access point handoff times within 50 ms.

fast switching An optimized Layer 3 forwarding path through a router. Fast switching optimizes routing table lookup by creating a special, easily searched table of known flows between hosts.

FD *See* Feasible distance.

feasibility condition With EIGRP, for a particular route, the case in which the RD is lower than the FD.

Feasible distance With EIGRP, the metric value for the lowest-metric route to a particular subnet.

feasible successor With EIGRP, a route that is not a successor route, but that meets the feasibility condition; can be used when the successor route fails, without causing loops.

FECN *See* Forward Explicit Congestion Notification.

FHSS *See* Frequency hopping spread spectrum.

FIB *See* Forwarding Information Base.

finish time (FT) A term used with WFQ for the number assigned to a packet as it is enqueued into a WFQ queue. WFQ schedules the currently lowest FT packet next.

flash updates *See* triggered updates.

Flush timer With RIP, a per-route timer, which is reset and grows with the Invalid timer. When the Flush timer mark is reached (default 240 seconds), the router removes the route from the routing table, and now accepts any other routes about the failed subnet.

ForeSight A Cisco-proprietary messaging protocol implemented in WAN switches that can be used to signal network status, including congestion, independent of end-user frames and cells.

Forward Delay timer An STP timer that dictates how long a port should stay in the listening state and the learning state.

Forward Explicit Congestion Notification A bit in the LAPF Frame Relay header that, when set to 1, implies that the frame has experienced congestion.

Forwarding Information Base A table used by CEF that holds information from the IP routing table in an mtrie structure for optimized searching and matching with destination IP addresses.

Forwarding state An 802.1D STP port state in which the port sends and receives frames.

fraggle attack An attack similar to a smurf attack, but using packets for the UDP Echo application instead of ICMP.

fragmentation In wireless LANs, a mechanism that counters issues related to RF interference by dividing a larger 802.11 data frame into smaller frames that are sent independently to the destination. *Also see* LFI.

framing From a Layer 1 perspective, the process of using special strings of electrical signals over a transmission medium to inform the receiver as to which bits are overhead bits, and which fit into individual subchannels.

Frequency hopping spread spectrum A type of spread spectrum that spreads RF signals over the frequency spectrum by transmitting the signal at different frequencies according to a hopping pattern. One of the original 802.11 physical layers used FHSS to offer data rates of 1 and 2 Mbps.

FRF Frame Relay Forum. A vendor consortium that formerly worked to further Frame Relay common vendor standards.

FRF.5 An FRF standard for Frame Relay-to-ATM Service Interworking in which both DTEs use Frame Relay, with ATM in between.

FRF.8 An FRF standard for Frame Relay-to-ATM Service Interworking in which one DTE uses Frame Relay and one uses ATM.

FRF.9 An FRF standard for payload compression.

FRF.11-c An FRF standard for LFI for VoFR (FRF.11) VCs, in which all voice frames are interleaved in front of data frames' fragments.

FRF.12 An FRF standard for LFI for data (FRF.3) VCs.

full drop A WRED process by which WRED discards all newly arriving packets intended for a queue, based on whether the queue's maximum threshold has been exceeded.

full duplex Ethernet feature in which a NIC or Ethernet port can both transmit and receive at the same instant in time. It can be used only when there is no possibility of collisions. Loopback circuitry on NIC cards is disabled to use full duplex.

full SPF calculation An SPF calculation as a result of changes inside the same area as a router, for which the SPF run must examine the full LSDB.

full update A routing protocol feature by which the routing update includes the entire set of routes, even if some or all of the routes are unchanged.

fully adjacent (OSPF) Any OSPF neighbor for which the database flooding process has completed.

Garbage timer *See* Flush timer.

gateway of last resort The notation in a Cisco IOS IP routing table that identifies the route used by that router as the default route.

Get In the context of SNMP, the Get command is sent by an SNMP manager, to an agent, requesting the value of a single MIB variable identified in the request. The Get request identifies the exact variable whose value the manager wants to retrieve. Introduced in SNMPv1.

GetBulk In the context of SNMP, the GetBulk command is sent by an SNMP manager, to an agent, requesting the values of multiple variables. The GetBulk command allows retrieval of complex structures, like a routing table, with a single command, as well as easier MIB walking.

GetNext In the context of SNMP, the GetNext command is sent by an SNMP manager, to an agent, requesting the value of a single MIB variable. The GetNext request identifies a variable for which the manager wants the variable name and value of the next MIB leaf variable in sequence.

GLBP Gateway Load Balancing Protocol. A Cisco-proprietary feature by which multiple routers can provide interface IP address redundancy, as well as cause a set of clients to load-balance their traffic across multiple routers inside the GLBP group.

global routing prefix The first 48 bits of an IPv6 global address, used for efficient route aggregation.

GLOP addressing The range 233.0.0.0 through 233.255.255.255 that IANA has reserved (RFC 2770) on an experimental basis. It can be used by anyone who owns a registered autonomous system number to create 256 global multicast addresses.

going active EIGRP jargon meaning that EIGRP has placed a route into active status.

Goodbye (EIGRP) An EIGRP message that is used by a router to notify its neighbors when the router is gracefully shutting down.

Graft Ack message Message sent by a PIM-DM router to a downstream router when it receives a Graft message from the downstream router; sent using the unicast address of the downstream router.

Graft message Message sent by a PIM-DM router to its upstream router asking to quickly restart forwarding the group traffic; sent using the unicast address of the upstream router.

granted window *See* receiver's advertised window.

half duplex Ethernet feature in which a NIC or Ethernet port can only transmit or receive at the same instant in time, but not both. Half duplex is required when a possibility of collisions exists.

hardware queue A small FIFO queue associated with each router's physical interface, for the purpose of making packets available to the interface hardware, removing the need for a CPU interrupt to start sending the next packet out the interface.

HDB3 *See* High Density Binary 3.

Hello (EIGRP) An EIGRP message that identifies neighbors, exchanges parameters, and is sent periodically as a keepalive function. Hellos do not require an Ack.

Hello (OSPF) A type of OSPF packet used to discover neighbors, check for parameter agreement, and monitor the health of another router.

hello interval With some routing protocols, the time period between successive Hello messages.

Hello timer An STP timer that dictates the interval at which the Root switch generates and sends Hello BPDUs.

High Density Binary 3 A serial-line encoding standard like B8ZS, but with each set of four consecutive 0s being changed to include a Bipolar Violation to maintain synchronization.

Hold timer With EIGRP, the timer used to determine when a neighboring router has failed, based on a router not receiving any EIGRP messages, including Hellos, in this timer period.

Holddown timer With RIP, a per-route timer (default 180 seconds) that begins when a route's metric changes to a larger value.

HSRP Hot Standby Router Protocol. A Cisco-proprietary feature by which multiple routers can provide interface IP address redundancy so that hosts using the shared, virtual IP address as their default gateway can still reach the rest of a network even if one or more routers fail.

I/G bit The most significant bit in the most significant byte of an Ethernet MAC address, its value implies that the address is a unicast MAC address (binary 0) or not (binary 1).

iBGP Internal BGP. A term referring to how a router views a BGP peer relationship, in which the peer is in the same AS.

IEEE 802.1X An IEEE standard that, when used with EAP, provides user authentication before their connected switch port allows the device to fully use the LAN.

IGMP Internet Group Management Protocol. A communication protocol between hosts and a multicast router by which routers learn of which multicast groups' packets need to be forwarded onto a LAN.

IGMP snooping A method for optimizing the flow of multicast IP packets passing through a LAN switch. The switch using IGMP snooping examines IGMP messages to determine which ports need to receive traffic for each multicast group.

IGMPv1 Host Membership Query A message sent by the multicast router, by default every 60 seconds, on each of its LAN interfaces to determine whether any host wants to receive multicast traffic for any group.

IGMPv1 Host Membership Report A message that each host sends, either in response to a router Query message or on its own, to all multicast groups for which it would like to receive multicast traffic.

IGMPv2 Group-Specific Query A message sent by a router, after receiving a Leave message from a host, to determine whether there are still any active members of the group. The router uses the group address as the destination address.

IGMPv2 Host Membership Query A message sent by a multicast router, by default every 125 seconds, on each of its LAN interfaces to determine whether any host wants to receive multicast traffic for any group.

IGMPv2 Host Membership Report A message sent by each host, either in response to a router Query or on its own, to all multicast groups for which it would like to receive multicast traffic.

IGMPv2 Leave A message sent by a host when it wants to leave a group, addressed to the All Multicast Routers address 224.0.0.2.

IGMPv3 Host Membership Query A message sent by a multicast router, by default every 125 seconds, on each of its LAN interfaces to determine whether any host wants to receive multicast traffic for any group.

IGMPv3 Host Membership Report A message sent by each host, either in response to a router query or on its own, to all multicast groups for which it would like to receive multicast traffic. The destination address on the Report is 224.0.0.22, and a host can specify the source address(es) from which it would like to receive the group traffic.

InARP *See* Inverse ARP.

Inform In the context of SNMP, the Inform command is sent by an SNMP manager to communicate a set of variables, and their values, to another SNMP manager. The main purpose is to allow multiple managers to exchange MIB information, and work together, without requiring each manager to individually use Get commands to gather the data.

infrastructure mode A wireless LAN that includes the use of access points. Infrastructure mode connects wireless users to a wired network and allows wireless users to roam throughout a facility between different access points. All 802.11 data frames in an infrastructure wireless LAN travel through the access point.

input event Any occurrence that could change a router's EIGRP topology table, including a received Update or Query, a failed interface, or the loss of a neighbor.

Inside Global address A NAT term describing an IP address representing a host that resides inside the enterprise network, with the address being used in packets outside the enterprise network.

Inside Local address A NAT term describing an IP address representing a host that resides inside the enterprise network, with the address being used in packets inside the enterprise network.

interface ID 64 bits at the end of an IPv6 global address, used to uniquely identify each host in a subnet.

internal DSCP A term used with Cisco LAN switches, referring to a DSCP value used when making QoS decisions about a frame. This value may not be the actual DSCP value in the IP header encapsulated inside the frame.

internal router (OSPF) A router that is not an ABR or ASBR in that all of its interfaces connect to only a single OSPF area.

Invalid timer With RIP, a per-route timer that increases until the router receives a routing update that confirms the route is still valid, upon which the timer is reset to 0. If the updates cease, the Invalid timer will grow, until reaching the timer setting (default 180 seconds), after which the route is considered invalid.

Inverse ARP Defined in RFC 1293, this protocol allows a Frame Relay–attached device to react to a received LMI "PVC up" message by announcing its Layer 3 addresses to the device on the other end of the PVC.

IPCP *See* IP Control Protocol.

IP Control Protocol The portion of PPP focused on negotiating IP features—for example, TCP or RTP header compression.

IP forwarding The process of forwarding packets through a router. Also call IP routing.

IP PBX A component that interfaces with a phone using IP and provides connections to the Public Switched Telephone Network (PSTN).

IP Precedence A 3-bit field in the first 3 bits of the ToS byte in the IP header, used for QoS marking.

IP prefix list *See* prefix list.

IP routing The process of forwarding packets through a router. Also called IP forwarding.

IP Source Guard A switch feature that examines incoming frames, comparing the source IP and MAC addresses to the DHCP snooping binding database, filtering frames whose addresses are not listed in the database for the incoming interface.

IPv4 Version 4 of the IP protocol, which is the generally deployed version worldwide (at publication), and uses 32-bit IP addresses.

IPv6 Version 6 of the IP protocol, which uses 128-bit IP addresses.

ISL Inter-Switch Link. Cisco-proprietary VLAN trunking protocol.

isolated VLAN With private VLANs, a secondary VLAN in which the ports can send and receive frames only with promiscuous ports in the primary VLAN.

Join/Prune message Sent by a PIM router to its upstream router to either request that the upstream router forward the group traffic or stop forwarding the group traffic that is currently being forwarded. If a PIM router wants to start receiving the group traffic, it lists the group address under the Join field. If it wants the upstream router to stop forwarding the group traffic, it lists the group address under the Prune field.

joining a group The process of installing a multicast application; also referred to as launching an application.

K value EIGRP (and IGRP) allows for the use of bandwidth, load, delay, MTU, and link reliability; the K values refer to an integer constant that includes these five possible metric components. Only bandwidth and delay are used by default, to minimize recomputation of metrics for small changes in minor metric components.

LACP Link Aggregation Control Protocol. Defined in IEEE 802.1AD, defines a messaging protocol used to negotiate the dynamic creation of PortChannels (EtherChannels) and to choose which ports can be placed into an EtherChannel.

LAPF *See* Link Access Procedure for Frame-Mode Bearer Services.

Layer 2 payload compression The process of taking the payload inside a Layer 2 frame, including the headers of Layer 3 and above, compressing the data, and then uncompressing the data on the receiving router.

Layer 2 protocol tunneling Another name for 802.1Q-in-Q. *See* 802.1Q-in-Q.

Layer *x* PDU The PDU used by a particular layer of a networking model, with *x* defining the layer.

LCP *See* Link Control Protocol.

Learning state An 802.1D STP transitory port state in which the port does not send or receive frames, but does learn the source MAC addresses from incoming frames.

LFI *See* Link Fragmentation and Interleaving.

limiting query scope (EIGRP) An effort to reduce the query scope with EIGRP, using route summarization or EIGRP stub routers.

line coding *See* encoding.

Link Access Procedure for Frame-Mode Bearer Services An ITU standard Frame Relay header, including the DLCI, DE, FECN, and BECN bits in the LAPF header, and a frame check in the LAPF trailer.

Link Control Protocol The portion of PPP focused on features that are unrelated to any specific Layer 3 protocol.

Link Fragmentation and Interleaving The process of breaking a frame into pieces, sending some of the fragments, and then sending all or part of a different packet, all of which is done to reduce the delay of the second packet.

Link-State Acknowledgment A type of OSPF packet used to acknowledge LSU packets.

Link-state advertisement The OSPF data structure that describes topology information.

Link-state database The data structure used by OSPF to hold LSAs.

link-state routing protocol Any routing protocol that uses the concept of using the SPF algorithm with an LSDB to compute routes.

Link-State Update A type of OSPF packet, used to communicate LSAs to another router.

Listening state An 802.1D STP transitory port state in which the port does not send or receive frames, and does not learn MAC addresses, but does wait for STP convergence and for CAM flushing by the switches in the network.

LMI *See* Local Management Interface.

local computation An EIGRP router's reaction to an input event, leading to the use of a feasible successor or going active on a route.

LOCAL_AS A reserved value for the BGP COMMUNITY path attribute that implies that the route should not be advertised outside the local confederation sub-AS.

Local Management Interface The Frame Relay protocol used between a DCE and DTE to manage the connection. Signaling messages for SVCs, PVC Status messages, and keepalives are all LMI messages.

LOCAL_PREF A BGP path attribute that is communicated throughout a single AS to signify which route of multiple possible routes is the best route to be taken when leaving that AS. A larger value is considered to be better.

LOF *See* Loss of Frame.

Loop Guard Protects against problems caused by unidirectional links between two switches. Watches for loss of received Hello BPDUs, in which case it transitions to a loop-inconsistent state instead of transitioning to a forwarding state.

loopback circuitry A feature of Ethernet NICs. When the NIC transmits an electrical signal, it "loops" the transmitted electrical current back onto the receive pair. By doing so, if another NIC transmits a frame at the same time, the NIC can detect the overlapping received electrical signals, and sense that a collision has occurred.

LOS Loss of Signal. A T1 alarm state that occurs when the receiver has not received any pulses of either polarity for a defined time period.

Loss of Frame A T1 alarm state that occurs when the receiver can no longer consistently identify the frame.

low-latency queuing (LLQ) A Cisco IOS queuing tool that uses MQC configuration commands, reserves a minimum bandwidth for some queues, provides high-priority scheduling for some queues, and polices those queues to prevent starvation of lower-priority queues during interface congestion.

LSA *See* Link-state advertisement.

LSA flooding The process of successive neighboring routers exchanging LSAs such that all routers have an identical LSDB for each area to which they are attached.

LSA type (OSPF) A definition that determines the data structure and information implied by a particular LSA.

LSAck *See* Link-State Acknowledgment.

LSDB *See* Link-state database.

LSRefresh Link-State Refresh. A timer that determines how often the originating router should reflood an LSA, even if no changes have occurred to the LSA.

LSU *See* Link-State Update.

LxPDU *See* Layer *x* PDU.

LZS The Lempel Ziv STAC compression algorithm is used in Frame Relay networks to define dynamic dictionary entries that list a binary string from the compressed data and an associated smaller string that represents it during transmission—thereby reducing the number of bits used to send data.

Management Information Base The definitions for a particular set of data variables, with those definitions following the SMI specifications. *See also* SMI.

man-in-the-middle attack A characterization of a network attack in which packets flow to the attacker, and then out to the true recipient. As a result, the user continues to send data, increasing the chance that the attacker learns more and better information.

map class An FRTS configuration construct, configured with the **map-class frame-relay** global configuration command.

mark probability denominator (MPD) Used by WRED to calculate the maximum percentage of packets discarded when the average queue depth falls between the minimum and maximum thresholds.

marking down Jargon referring to a policer action through which, instead of discarding an out-of-contract packet, the policer marks a different IPP or DSCP value, allowing the packet to continue on its way, but making the packet more likely to be discarded later.

MaxAge An OSPF timer that determines how long an LSA can remain in the LSDB without having heard a reflooded copy of the LSA.

Maxage timer An STP timer that dictates how long a switch should wait when it ceases to hear Hellos.

maximum reserved bandwidth A Cisco IOS interface setting, as a percentage between 1 and 99, that defines how much of the interface's bandwidth setting may be allocated by a queuing tool. The default value is 75 percent.

Maximum Response Time After a host receives an IGMP Query, the amount of time (default, 10 seconds) the host has to send the IGMP Report.

Maximum Segment Size A TCP variable that defines the largest number of bytes allowed in a TCP segment's Data field. The calculation does not include the TCP header. With a typical IP MTU of 1500 bytes, the resulting default MSS would be 1460. TCP hosts must support an MSS of at least 536 bytes

maximum threshold WRED compares this setting to the average queue depth to decide whether packets should be discarded. All packets are discarded if the average queue depth rises above this maximum threshold.

Maximum Transmission Unit An IP variable that defines the largest size allowed in an IP packet, including the IP header. IP hosts must support an MTU of at least 576 bytes.

MD5 *See* Message Digest 5.

MD5 hash A term referring to the process of applying the Message Digest 5 (MD5) algorithm to a string, resulting in another value. The original string cannot be easily computed even when the hash is known, making this process a strong method for storing passwords.

Measured Round-Trip Time A TCP variable used as the basis for a TCP sender's timer defining how long it should wait for a missing acknowledgement before resending the data.

Message Digest 5 A method of applying a mathematical formula, with input including a private key, the message contents, and sometimes a shared text string, with the resulting digest being included with the message. The sender and the receiver perform the same math to allow authentication and to prove that no intermediate device changed the message contents.

metric With routing protocols, the measurement of favorability that determines which entry will be installed in a routing table if more than one router is advertising that exact network and mask.

MIB *See* Management Information Base.

MIB walk In SNMP, the process of a manager using successive GetNext and GetBulk commands to discover the exact MIB structure supported by an SNMP agent. The process involves the manager asking for each successive MIB leaf variable.

MIB-I The original standardized set of generic SNMP MIB variables, defined in RFC 1158.

MIB-II The most recent standardized set of generic SNMP MIB variables, defined in RFC 1213 and updated in RFCs 2011 through 2013.

mincir *See* Minimum CIR.

minimum CIR Jargon referring to the minimum value to which adaptive shaping will lower the shaping rate.

minimum threshold WRED compares this setting to the average queue depth to decide whether packets should be discarded. No packets are discarded if the average queue depth falls below this minimum threshold.

MLD *See* Multicast Listener Discovery Protocol.

MLP *See* Multilink PPP.

MLP LFI The PPP function for fragmenting packets, plus interleaving delay-sensitive later-arriving packets between the fragments of the first packet.

MLS *See* Multilayer Switching.

modified tail drop A WFQ term referring to its drop logic, which is similar to tail-drop behavior.

Modular QoS CLI (MQC) The common set of IOS configuration commands that is used with each QoS feature whose name begins with "Class-Based."

MOSPF *See* Multicast Open Shortest Path First.

MPLS Experimental (EXP) A 3-bit field in an MPLS header used for marking frames.

MRT *See* Maximum Response Time.

MRTT *See* Measured Round-Trip Time.

MSS *See* Maximum Segment Size.

MST *See* Multiple Spanning Trees.

MTU *See* Maximum Transmission Unit.

MULTI_EXIT_DISC (MED) A BGP path attribute that allows routers in one AS to set a value and advertise it into a neighboring AS, impacting the decision process in that neighboring AS. A smaller value is considered better. Also called the BGP metric.

multi-action policing In MQC and CB Policing, a configuration style by which, for one category of packets (conform, exceed, or violate), more than one marking action is defined for a single category. For example, marking DSCP and DE.

multicast IP address range IP multicast address range from 224.0.0.0 through 239.255.255.255.

multicast IP address structure The first 4 bits of the first octet must be 1110. The last 28 bits are unstructured.

Multicast Listener Discovery Protocol The IPv6 protocol used for the discovery of which hosts are listening for which multicast IP addresses for IPv6.

multicast MAC address A 48-bit address that is calculated from a Layer 3 multicast address by using 0x0100.5E as the multicast vendor code (OUI) for the first 24 bits, always binary 0 for the 25th bit, and copying the last 23 bits of the Layer 3 multicast address.

Multicast Open Shortest Path First A multicast routing protocol that operates in dense mode and depends on the OSPF unicast routing protocol to perform its multicast functions.

multicast scoping The practice of defining boundaries that determine how far multicast traffic will travel in your network.

multicast state information The information maintained by a router for each multicast entry in its multicast routing table, such as incoming interface, outgoing interface list, Uptime timer, Expire timer, etc.

multicasting Sending a message from a single source or multiple sources to selected multiple destinations across a Layer 3 network in one data stream.

Multilayer Switching A process whereby a switch, when making a forwarding decision, uses not only Layer 2 logic but other OSI layer equivalents as well.

Multilink PPP A PPP feature used to load balance multiple parallel links at Layer 2 by fragmenting frames, sending one frame over each of the links in the bundle, and reassembling them at the receiving end of the link.

multipath An issue whereby parts of the RF signal take different paths from the source to the destination, which causes direct and reflected signals to reach the receiver at different times, and corresponding bit errors.

Multiple Spanning Trees Defined in IEEE 802.1s, a specification for multiple STP instances when using 802.1Q trunks

NAT Network Address Translation. Defined in RFC 1631, a method of translating IP addresses in headers with the goal of allowing multiple hosts to share single public IP addresses, thereby reducing IPv4 public address depletion.

native VLAN The one VLAN on an 802.1Q trunk for which the endpoints do not add the 4-byte 802.1Q tag when transmitting frames in that VLAN.

NCP *See* Network Control Protocol.

neighbor (EIGRP) With EIGRP, a router sharing the same primary subnet, with which Hellos are exchanged, parameters match, and with which routes can be exchanged.

neighbor (OSPF) Any other router, sharing a common data link, with which a router exchanges Hellos, and for which the parameters in the Hello pass the parameter-check process.

neighbor state A state variable kept by a router for each known neighbor or potential neighbor.

Neighbor Type In BGP, either external BGP (eBGP), confederation eBGP, or internal BGP (iBGP). The term refers to a peer connection, and whether the peers are in different ASs (eBGP), different confederation sub-ASs (confederation eBGP), or in the same AS (iBGP).

nested policy maps An MQC configuration style by which one policy map calls a second policy map. For example, a shaping policy map can call an LLQ policy map to implement LLQ for packets shaped by CB Shaping.

network allocation vector A time value that each wireless station must set based on the duration value found in every 802.11 frame. The time value counts down and must be equal to zero before a station is allowed to access the wireless medium. The result is a collision-avoidance mechanism.

Network Based Application Recognition (NBAR) A Cisco IOS feature that performs deep packet inspection to classify packets based on application layer information.

Network Control Protocol The portions of PPP focused on features that are related to specific Layer 3 protocols.

Network layer reachability information A BGP term referring to an IP prefix and prefix length.

Network Time Protocol An Internet standard (RFC 1305) that defines the messages and modes used for IP hosts to synchronize their time-of-day clocks.

network type (OSPF) A characteristic of OSPF interfaces that determines whether a DR election is attempted, whether or not neighbors must be statically configured, and the default Hello and Dead timer settings.

Next Hop field With a routing update, or routing table entry, the portion of a route that defines the next router to which a packet should be sent to reach the destination subnet. With routing protocols, the Next Hop field may define a router other than the router sending the routing update.

NEXT_HOP A BGP Path Attribute that lists the next-hop IP address used to reach an NLRI.

NLPID Network Layer Protocol ID is a field in the RFC 2427 header that is used as a Protocol Type field in order to identify the type of Layer 3 packet encapsulated inside a Frame Relay frame.

NLRI *See* Network layer reachability information.

no drop A WRED process by which WRED does not discard packets during times in which a queue's minimum threshold has not been passed.

NO_ADVERT A reserved value for the BGP COMMUNITY path attribute that implies that the route should not be advertised to any other peer.

NO_EXPORT A reserved value for the BGP COMMUNITY path attribute that implies that the route should not be advertised outside the local AS.

NO_EXPORT_SUBCONFED The RFC 1997 name for the reserved COMMUNITY path attribute known to Cisco IOS as LOCAL_AS. (*See* LOCAL_AS.)

Not-so-stubby area A type of OSPF stub area that, unlike stub areas, can inject external routes into the NSSA area.

NSSA *See* Not-so-stubby area.

NTP *See* Network Time Protocol.

NTP broadcast client An NTP client that assumes that a server will send NTP broadcasts, removing the requirement for the client to have the NTP server's IP address pre-configured.

NTP client mode An NTP mode in which an NTP host adjusts its clock in relation to an NTP server's clock.

NTP server mode An NTP mode in which an NTP host does not adjust its clock, but in which it sends NTP messages to clients so that the clients can update their clocks based on the server's clock.

NTP symmetric active mode An NTP mode in which two or more NTP servers mutually synchronize their clocks.

OAM *See* Operation, Administration, and Maintenance.

OFDM *See* Orthogonal frequency division multiplexing.

offset list A Cisco IOS configuration tool for RIP and EIGRP for which the list matches routes in routing updates, and adds a defined value to the sent or received metric for the routes. The value added to the metric is the *offset*.

One-time password Defined in RFC 2289, a mechanism by which a shared key and a secret key together feed into a hash algorithm, creating a password that is transmitted over a network. Because the shared key is not reused, the hash value is only valid for that individual authentication attempt.

OOF *See* Out of Frame.

Operation, Administration, and Maintenance A term referring to the processes and bits in the data stream used to manage the Telco TDM hierarchy.

optional nontransitive A characterization of a BGP path attribute in which BGP implementations are not required to support the attribute (optional), and for which if a router receives a route with such an attribute, the router should remove the attribute before advertising the route (nontransitive).

optional transitive A characterization of a BGP path attribute in which BGP implementations are not required to support the attribute (optional), and for which if a router receives a route with such an attribute, the router should forward the attribute unchanged (transitive).

ORIGIN A BGP path attribute that implies how the route was originally injected into some router's BGP table.

ORIGINATOR_ID Used by RRs to denote the RID of the iBGP neighbor that injected the NLRI into the AS.

Orthogonal frequency division multiplexing A technology that sends a high-speed data stream over multiple subcarriers simultaneously. It is highly immune to multipath interference. 802.11a and 802.11g specify the use of OFDM.

OTP *See* One-time password.

Out of Frame A T1 alarm state that occurs when the receiver can no longer consistently identify the frame. *See* LOF.

Outside Global address A NAT term describing an IP address representing a host that resides outside the enterprise network, with the address being used in packets outside the enterprise network.

Outside Local address A NAT term describing an IP address representing a host that resides outside the enterprise network, with the address being used in packets inside the enterprise network.

overloading Another term for Port Address Translation. See *PAT.*

PAgP *See* Port Aggregation Protocol.

PAP *See* Password Authentication Protocol.

partial SPF calculation An SPF calculation for which a router does not need to run SPF for any LSAs inside its area, but instead runs a very simple algorithm for changes to LSAs outside its own area.

partial update A routing protocol feature by which the routing update includes only routes that have changed, rather than include the entire set of routes.

passive (EIGRP) A state for a route in an EIGRP topology table that indicates that the router believes that the route is stable, and it is not currently looking for any new routes to that subnet.

passive mode FTP Defines a particular behavior for FTP regarding the establishment of TCP data connections. In passive mode, an FTP server uses the FTP PORT command, over the FTP control connection, to tell the FTP client the port on which the server will be listening for a new data connection. The client allocates an unused port, and initiates a connection to the FTP server's earlier-declared port.

passive scanning Each 802.11 station passively monitors each RF channel for a specific amount of time and listens for beacons. Stations use the signal strengths of found beacons to determine the access point or ad hoc network with which to attempt association.

Password Authentication Protocol An Internet standard authentication protocol that uses clear-text passwords and a two-way handshake to perform authentication over a PPP link.

PAT *See* Port Address Translation.

path attribute A term generally describing characteristics about BGP paths that are advertised in BGP Updates.

payload compression *See* Layer 2 payload compression.

PCM *See* pulse code modulation.

PDU *See* Protocol data unit.

Peak information rate In two-rate policing, the second and higher rate defined to the policer.

peer group In BGP, a configuration construct in which multiple neighbors' parameters can be configured as a group, thereby reducing the length of the configuration. Additionally, BGP performs routing policy logic against only one set of Updates for the entire peer group, improving convergence time.

Per-Hop Behavior With DiffServ, a DSCP marking and a related set of QoS actions applied to packets that have that marking.

permanent multicast group The multicast addresses assigned by IANA.

Permanent virtual circuit A predefined VC. A PVC can be equated to a leased line in concept.

Per-VLAN Spanning Tree Plus A Cisco-proprietary STP implementation, created many years before IEEE 802.1s and 802.1w, that speeds convergence and allows for one STP instance for each VLAN.

PHB *See* Per-Hop Behavior.

PIM Hello message Sent by a PIM router, by default every 30 seconds, on every interface on which PIM is configured to discover neighbors, establish adjacency, and maintain adjacency.

PIM-DM *See* Protocol Independent Multicast dense-mode routing protocol.

PIM-SM Protocol Independent Multicast sparse-mode routing protocol. Does not depend on any unicast routing protocol to perform its multicast functions.

PIM-SM (S,G) RP-bit Prune When a PIM-SM router switches from RPT to SPT, it sends a PIM-SM Prune message for the source and the group with the RP bit set to its upstream router on the shared tree. RFC 2362 uses the notation PIM-SM (S, G) RP-bit Prune for this message.

PIR *See* Peak information rate.

point coordination function An optional contention-free 802.11 access protocol that requires the access point to poll wireless stations before they are able to send frames. Not commonly implemented.

Point-to-Point Protocol An Internet standard serial data-link protocol, used on synchronous and asynchronous links, that provides data-link framing, link negotiation, Layer 3 interface features, and other functions.

poison reverse With RIP, the advertisement of a poisoned route out an interface, when that route was formerly not advertised out that interface due to split horizon rules.

policing rate The rate at which a policer limits the bits exiting or entering the policer.

policy map A term referring to the MQC **policy-map** command and its related subcommands, which are used to apply QoS actions to classes of packets.

policy routing Cisco IOS router feature by which a route map determines how to forward a packet, typically based on information in the packet other than the destination IP address.

Port Address Translation A NAT term describing the process of multiplexing TCP and UDP flows, based on port numbers, to a small number of public IP addresses. Also called *NAT overloading*.

Port Aggregation Protocol A Cisco-proprietary messaging protocol used to negotiate the dynamic creation of PortChannels (EtherChannels) and to choose which ports can be placed into an EtherChannel.

port security A switch feature that limits the number of allowed MAC addresses on a port, with optional limits based on the actual values of the MAC addresses.

PortFast Cisco-proprietary STP feature in which a switch port, known to not have a bridge or switch attached to it, transitions from disabled to forwarding state without using any intermediate states.

power-save mode A mechanism for conserving battery power in wireless stations. The access point buffers data frames destined to sleeping stations, which wake periodically to learn from information in the beacon frame whether or not data frames are waiting for transmission. The radio card receives applicable data frames and then goes back to sleep.

PPP *See* Point-to-Point Protocol.

prefix A numeric value between 0 and 32 (inclusive) that defines the number of beginning bits in an IP address for which all IP addresses in the same group have the same value. Alternative: The

number of binary 1s beginning a subnet mask, written as a decimal value between 0 and 32, used as a more convenient form of representing the subnet mask.

prefix list A Cisco IOS configuration tool that can be used to match routing updates based on a base network address, a prefix, and a range of possible masks used inside the values defined by the base network address and prefix.

priority (OSPF) An administrative setting, included in Hellos, that is the first criteria for electing a DR. The highest priority wins, with values from 1–255, with priority 0 meaning a router cannot become DR or BDR.

priority queue (PQ) Jargon referring to any queue that receives priority service, often used for queues in an LLQ configuration that have the **priority** command configured.

priority queuing (PQ) A Cisco IOS queuing tool most notable for its scheduler, which always services the high-priority queue over all other queues.

private addresses RFC 1918-defined IPv4 network numbers that are not assigned as public IP address ranges, and are not routable on the Internet. Intended for use inside enterprise networks.

private AS A BGP ASN whose value is between 64,512 and 65,535. These values are not assigned for use on the Internet, and can be used for private purposes, typically either within confederations or by ISPs to hide the ASN used by some customers.

private VLAN A Cisco switch feature that allows separation of ports as if they were in separate VLANs, while allowing the use of a single IP subnet for all ports.

process switching A Layer 3 forwarding path through a router that does not optimize the forwarding path through the router.

promiscuous port With private VLANs, a port that can send and receive frames with all other ports in the private VLAN.

Protocol data unit A generic term that refers to the data structure used by a layer in a layered network architecture when sending data.

Protocol Independent Multicast dense-mode routing protocol Does not depend on any unicast routing protocol to perform its multicast functions.

proxy ARP A router feature used when a router sees an ARP request searching for an IP host's MAC, when the router believes the IP host could not be on that LAN because the host is in another subnet. If the router has a route to reach the subnet where the ARP-determined host resides, the router replies to the ARP request with the router's MAC address.

Prune Override On a multiaccess network, when a PIM-DM or PIM-SM router receives a Prune message, it starts a 3-second timer. If it receives a Join message on the multiaccess network from another router before the timer expires, it considers the message as an override to the previously received Prune message and continues forwarding the group traffic on the LAN interface; otherwise, it stops forwarding the traffic on the LAN interface.

pruning *See* VTP pruning.

public wireless LAN A wireless LAN that offers connections to the Internet from public places, such as airports, hotels, and coffee shops.

pulse code modulation An early standard from AT&T for encoding analog voice as a digital signal for transmission over a TDM network. PCM requires 64 kbps, and is the basis for the DS0 speed.

PVC *See* Permanent virtual circuit.

PVST+ *See* Per-VLAN Spanning Tree Plus.

quartet A set of four hex digits listed in an IPv6 address. Each quartet is separated by a colon.

querier election When multiple routers are connected to a subnet, only one should be sending IGMP queries. It is called a querier. IGMPv1 does not have any rules for electing a querier. In IGMPv2 and IGMPv3, a router with the lowest interface IP address on the subnet is elected as a querier.

Query (EIGRP) An EIGRP message that is used to ask neighboring routers to verify their route to a particular subnet. Query messages require an Ack.

query scope (EIGRP) The characterization of how far EIGRP Query messages flow away from the router that first notices a failed route and goes active for a particular subnet.

queue starvation A possible side effect of a scheduler that performs strict-priority scheduling of a queue, which can result in lower-priority queues getting little or no service.

radio management aggregation Reduces the bandwidth necessary for radio management information, such as access point status messages, that is sent across the network by eliminating redundant management information.

RADIUS A protocol, defined in RFC 2865, that defines how to perform authentication between an authenticator (for example, a router) and an authentication server that holds a list of usernames and passwords.

Rapid Spanning Tree Protocol Defined in IEEE 802.1w, a specification to enhance the 802.1D standard to improve the speed of STP convergence.

RARP *See* Reverse ARP.

RD *See* Reported distance.

Ready To Send On a serial cable, the pin lead set by the DTE to tell the DCE that the DTE wants to send data.

receiver's advertised window In TCP, a TCP host sets the TCP header's Window field to the number of bytes it allows the other host to send before requiring an acknowledgement. In effect, the receiving host, by stating a particular window size, grants the sending host the right to send that number of bytes in a single window.

Red Alarm A T1 alarm state that occurs when a device has detected a local LOF/LOS/AIS condition. The device in Red alarm state then sends a Yellow alarm signal.

regular expression A list of interspersed alphanumeric literals and metacharacters that are used to apply complex matching logic to alphanumeric strings. Often used for matching AS_PATHs in Cisco routers.

Reliable Transport Protocol A protocol used for reliable multicast and unicast transmissions. Used by EIGRP.

remaining bandwidth A CBWFQ and LLQ term referring to the bandwidth on an interface that is neither reserved nor allocated via a **priority** command.

Rendezvous point In the PIM-SM design, the central distribution point to which the multicast traffic is first delivered from the source designated router.

Reply (EIGRP) An EIGRP message that is used by neighbors to reply to a query. Reply messages require an Ack.

Reported distance With EIGRP, the metric (distance) of a route as reported by a neighboring router.

Report Suppression mechanism When a Query is received from a router, each host randomly picks a time between 0 and the Maximum Response Time period to send a Report. When the host with the smallest time period first sends the Report, the rest of the hosts suppress their reports.

Response In the context of SNMP, the Response command is sent by an SNMP agent, back to a manager, in response to any of the three types of Get requests, or in response to a Set request.

It is also used by a manager in response to a received Inform command from another SNMP manager. The Response holds the value(s) of the requested variables.

Retransmission Timeout With EIGRP, a timer started when a reliable (to be acknowledged) message is transmitted. For any neighbor(s) failing to respond in its RTO, the RTP protocol causes retransmission. RTO is calculated based on SRTT.

Reverse ARP A standard (RFC 903) protocol by which a LAN-attached host can dynamically broadcast a request for a server to assign it an IP address. *See also* ARP.

RF channel The specific frequency subband on which the radio card or access point is operating. The RF channel is set in the access point or ad hoc stations.

RGMP *See* Router-Port Group Management Protocol.

RID *See* Router ID.

ROMMON An alternative software loaded into a Cisco router, used for low-level debugging and for password recovery.

Root Guard Cisco-proprietary STP feature in which a switch port monitors for incoming superior Hellos, and reacts to a superior Hello to prevent any switch connected to that port from becoming root.

Root Port The single port on each non-root switch upon which the best Hello BPDU is received.

route map A configuration tool in Cisco IOS that allows basic programming logic to be applied to a set of items. Often used for decisions about what routes to redistribute, and for setting particular characteristics of those routes—for instance, metric values.

route poisoning The process of sending an infinite-metric route in routing updates when that route fails.

route redistribution The process of taking routes known through one routing protocol and advertising those routes with another routing protocol.

route reflector A BGP feature by which a router learns iBGP routes, and then forwards them to other iBGP peers, reducing the required number of iBGP peers while also avoiding routing loops.

route reflector client A BGP router that, unknown to it, is aided by a route reflector server to cause all iBGP routers in an AS to learn all eBGP-learned prefixes.

route reflector non-client A BGP router in an AS that uses route reflectors, but that is not aided by any RR server.

route reflector server A BGP router that forwards iBGP-learned routes to other iBGP routers.

Route Tag field A field within a route entry in a routing update, used to associate a generic number with the route. It is used when passing routes between routing protocols, allowing an intermediate routing protocol to pass information about a route that is not natively defined to that intermediate routing protocol. Frequently used for identifying certain routes for filtering by a downstream routing process.

routed interface An interface on a Cisco IOS–based switch that is treated as if it were an interface on a router.

Router ID The 32-bit number used to represent an OSPF router.

Router-Port Group Management Protocol A Cisco-proprietary Layer 2 protocol that enables a router to communicate to a switch which multicast group traffic the router does and does not want to receive from the switch.

routing black hole A problem that occurs when an AS does not run BGP on all routers, with synchronization disabled. The routers running BGP may believe they have working routes to reach a prefix, and forward packets to internal routers that do not run BGP and do not have a route to reach the prefix.

RP *See* Rendezvous point.

RPF check Designed to solve the problems of multicast duplication and multicast routing loops. For every multicast packet received, a multicast router examines its source IP address, consults its unicast routing table, determines which interface it would use to go in the reverse direction toward the source IP address, compares it with the interface on which the packet was received, and, if they match, accepts the packet and forwards it; otherwise, the router drops the packet.

RSTP *See* Rapid Spanning Tree Protocol.

RTO *See* Retransmission Timeout.

RTP *See* Reliable Transport Protocol.

RTP header compression The process of taking the IP, UDP, and RTP headers of a voice or video packet, compressing them, and then uncompressing them on the receiving router.

RTS *See* Ready To Send.

RTS/CTS Request-to-send/clear-to-send. A mechanism that counters collisions caused by hidden nodes. If enabled, the station or access point must first send an RTS frame and receive a CTS frame before sending each data frame.

RXBOOT An alternative software loaded into a Cisco router, used for basic IP connectivity—most useful when Flash memory is broken and you need IP connectivity to copy a new IOS image into Flash memory.

SAFE Blueprint An architecture and set of documents that defines Cisco's best recommendations for how to secure a network.

same-layer interaction The two computers use a protocol with which to communicate with the same layer on another computer. The protocol defined by each layer uses a header that is transmitted between the computers to communicate what each computer wants to do.

scheduler A queuing tool's logic by which it selects the next packet to dequeue from its many queues.

sequence number (OSPF) In OSPF, a number assigned to each LSA, ranging from 0x80000001 and wrapping back around to 0x7FFFFFFF, which is used to determine which LSA is most recent.

sequence number (SN) A term used with WFQ for the number assigned to a packet as it is enqueued into a WFQ. WFQ schedules the currently lowest SN packet next.

Service Interworking The process, defined by FRF.5 and FRF.8, for combining ATM and FR technologies for an individual VC.

service policy A term referring to the MQC **service-policy** command, which is used to enable a policy map on an interface.

Service set identifier Defines a particular wireless LAN. The SSID configured in the radio card must match the SSID in the access point before the station can connect with the access point.

Set In the context of SNMP, the Set command is sent by an SNMP manager, to an agent, requesting that the agent set a single identified variable to the stated value. The main purpose is to allow remote configuration and remote operation, such as shutting down an interface by using an SNMP Set of an interface state MIB variable.

SF *See* Superframe.

shaping rate The rate at which a shaper limits the bits exiting the shaper.

shared distribution tree In PIM-SM, the path of the group traffic that flows from the RP to the routers that need the traffic. It is also called the root-path tree (RPT), because it is rooted at the RP.

shortest-path tree switchover In the PIM-SM design, the process by which a PIM-SM router can build the SPT between itself and the source of a multicast group and take advantage of the most efficient path available from the source to the router as long as it has one directly connected group member. Once it builds an SPT, it sends a PIM-SM (S, G) RP-bit Prune toward the upstream router on the shared tree.

single-rate, three-color policer Policing in which a single rate is metered, and packets are placed into one of three categories (conform, exceed, or violate).

single-rate, two-color policer Policing in which a single rate is metered, and packets are placed into one of two categories (conform or exceed).

Signal-to-noise ratio The difference between the measured signal power and the noise power that a particular receiver sees at a given time. Higher SNRs generally indicate better performance.

Slow Start A method for how a TCP sender grows its calculated CWND variable, thereby growing the allowed window for the connection. Slow Start grows CWND at an exponential rate.

Slow Start Threshold A calculated TCP variable, used along with the TCP CWND variable, to dictate a TCP sender's behavior when it recognizes packet loss. As CWND grows after packet loss, the TCP sender increases CWND based on Slow Start rules, until CWND grows to be as high as the SSThresh setting, at which point TCP Congestion Avoidance logic is used. Essentially, SSThresh is the threshold at which Slow Start logic ends.

SLSM *See* Static Length Subnet Masking.

SMI *See* Structure of Management Information.

Smoothed Round-Trip Time With EIGRP, a purposefully slowly changing measurement of round-trip time between neighbors, from which the EIGRP RTO is calculated.

smurf attack A style of attack in which an ICMP Echo is sent with a directed broadcast (subnet broadcast) destination IP address, and a source address of the host that is being attacked. The attack can result in the Echo reaching a large number of hosts, all of which reply by sending an Echo Reply to the host being attacked.

SNMP agent A process on a computing device that accepts SNMP requests, responds with SNMP-structured MIB data, and initiates unsolicited Trap messages back to an SNMP management station.

SNMP manager A process on a computing device that issues requests for SNMP MIB variables from SNMP agents, receives and processes the MIB data, and accepts unsolicited Trap messages from SNMP agents.

SNR *See* Signal-to-noise ratio.

socket A 3-tuple consisting of an IP address, port number, and transport layer protocol. TCP connections exist between a pair of sockets.

soft reconfiguration A BGP process by which a router reapplies routing policy configuration (route maps, filters, and the like) based on stored copies of sent and received BGP Updates.

software queue A queue created by Cisco IOS as a result of the configuration of a queuing tool.

source DR A designated router that is directly connected with a source of the multicast group.

source registration In the PIM-SM design, the process by which a source DR, after it starts to receive the group traffic, encapsulates the multicast packets in the unicast packets and sends them to the RP.

source-based distribution tree Method by which a dense-mode routing protocol distributes multicast traffic from a source to all the segments of a network. Also called shortest-path tree (SPT), because it uses the shortest routing path from the source to the segments of the network.

source-specific addresses The range 232.0.0.0 through 232.255.255.255 that is allocated by IANA for SSM destination addresses and is reserved for use by source-specific applications and protocols.

Source-Specific Multicast IGMPv3 is designed to support source filtering. IGMPv3 allows a host to indicate interest in receiving multicast packets only from specific source addresses, or from all but specific source addresses, sent to a particular multicast destination address.

sparse-mode protocol A multicast routing protocol that forwards the multicast traffic only when requested by a downstream router.

Spanning Tree Protocol Defined in IEEE 802.1D, a protocol used on LAN bridges and switches to dynamically define a logical network topology that allows all devices to be reached, but prevents the formation of loops.

SPF algorithm The algorithm used by OSPF and IS-IS to compute routes based on the LSDB.

SPF calculation The process of running the SPF algorithm against the LSDB, with the result being the determination of the current best route(s) to each subnet.

split horizon Instead of advertising all routes out a particular interface, the routing protocol omits the routes whose outgoing interface field matches the interface out which the update would be sent.

spread spectrum A technology that enables frequency reuse. Two variants exist: frequency hopping (FHSS) and direct sequence (DSSS). Both techniques spread the signal power over a relatively wide portion of the frequency spectrum over time, to reduce interference between systems.

SRTT *See* Smoothed Round-Trip Time.

SSID *See* Service set identifier.

SSM *See* Source-Specific Multicast.

SSThresh *See* Slow Start Threshold.

Static Length Subnet Masking A strategy for subnetting a classful network for which all masks/prefixes are the same value for all subnets of that one classful network.

sticky learning In switch port security, the process whereby the switch dynamically learns the MAC address(es) of the device(s) connected to a switch port, and then adds those addresses to the running configuration as allowed MAC addresses for port security.

STP *See* Spanning Tree Protocol.

straight-through cable Copper cable with RJ-45 connectors in which the wire at pin 1 on one end is connected to pin 1 on the other end; the wire at pin 2 is connected to pin 2 on the other end; and so on.

strict priority A queuing scheduler's logic by which, if a particular queue has packets in it, those packets always get serviced next.

Structure of Management Information The SNMP specifications, standardized in RFCs, defining the rules by which SNMP MIB variables should be defined.

stub area An OSPF area into which external (type 5) LSAs are not introduced by its ABRs; instead, the ABRs originate and inject default routes into the area.

stub network (OSPF) A network/subnet to which only one OSPF router is connected.

stub router (EIGRP) A router that should not be used to forward packets between other routers. Other routers will not send Query messages to a stub router.

stub router (OSPF) A router that should either permanently or temporarily not be used as a transit router. Can wait a certain time after OSPF process start, or after BGP notifies OSPF that BGP has converged, before ceasing to be a stub router.

stuck-in-active The condition in which a route has been in an EIGRP active state for longer than the router's Active timer.

sub-AS The term referring to a group of iBGP routers in a confederation, with the group members being assigned a hidden ASN for the purposes of loop avoidance.

subnet A subset of a classful IP network, as defined by a subnet mask, which used to address IP hosts on the same Layer 2 network in much the same way as a classful network is used.

subnet broadcast address A single address in each subnet for which packets sent to this address will be broadcast to all hosts in the subnet. It is the highest numeric value in the range of IP addresses implied by a subnet number and prefix/mask.

subnet ID 16 bits between the interface ID and global routing prefix in an IPv6 global address, used for subnet assignment inside an enterprise.

subnet mask A dotted-decimal number used to help define the structure of an IP address. The binary 0s in the mask identify the host portion of an address, and the binary 1s identify either the combined network and subnet part (when thinking classfully) or the network prefix (when thinking classlessly).

subnet number A dotted-decimal number that represents a subnet. It is the lowest numeric value in the range of IP addresses implied by a subnet number and prefix/mask.

subnet zero When subnetting a class A, B, or C address, the subnet for which all subnet bits are binary 0.

successor route With EIGRP, the route to each destination for which the metric is the lowest of all known routes to that network.

summary route A route that is created to represent one or more smaller component routes, typically in an effort to reduce the size of routing and topology tables.

Superframe An early T1 framing standard.

superior BPDU Jargon used by STP mostly when discussing the root election process; refers to a Hello with a lower bridge ID. Sometimes refers to a Hello with the same bridge ID as another, but with better values for the tiebreakers in the election process.

supplicant The 802.1X driver that supplies a username/password prompt to the user and sends/receives the EAPoL messages.

SVC *See* Switched virtual circuit.

switched interface An interface on a Cisco IOS–based switch that is treated as if it were an interface on a switch.

Switched virtual circuit A VC that is set up dynamically when needed. An SVC can be equated to a dial-on-demand connection in concept.

synchronization In BGP, a feature in which BGP routes cannot be considered to be a best route to reach an NLRI unless that same prefix exists in the router's IP routing table as learned via some IGP.

T1 A name used for DS1 lines inside the North American TDM hierarchy.

T3 A name used for DS3 lines inside the North American TDM hierarchy.

TACACS+ A Cisco-proprietary protocol that defines how to perform authentication between an authenticator (for example, a router) and an authentication server that holds a list of usernames and passwords.

tail drop An event in which a new packet arrives, needing to be placed into a queue, and the queue is full—so the packet is discarded.

Time Interval (Tc) Variable name for the time interval used by shapers and by CAR.

TCP code bits Single-bit fields in the TCP header. For example, the TCP SYN and ACK code bits are used during connection establishment.

TCP flags The same thing as TCP code bits. *See* TCP code bits.

TCP header compression The process of taking the IP and TCP headers of a packet, compressing them, and then uncompressing them on the receiving router.

TCP intercept A Cisco router feature in which the router works to prevent SYN attacks either by monitoring TCP connections flowing through the router, or by actively terminating TCP connection until the TCP connection is established and then knitting the client-side connection with a server-side TCP connection.

TCP SYN flood An attack by which the attacker initiates many TCP connections to a server, but does not complete the TCP connections, by simply not sending the third segment normally used to establish the connection. The server may consume resources and reject new connection attempts as a result.

TDM *See* Time-division multiplexing.

TDM hierarchy The structure inside telcos' original digital circuit build-out in the mid-1900s, based upon using TDM to combine and disperse smaller DS levels into larger levels, and vice versa.

Temporal Key Integrity Protocol An enhanced version of WEP that is part of the 802.11i standard and has an automatic key-update mechanism that makes it much more secure than WEP. TKIP is not as strong as AES in terms of data protection.

terminal history The feature in a Cisco IOS device by which a terminal session's previously typed commands are remembered, allowing the user to recall the old commands to the command line through a simple key sequence (for example, the up-arrow key).

Time-division multiplexing The process of combining multiple synchronized input signals over a single medium by giving each signal its own time slot, and then breaking out those signals.

Time to Live A field in the IP header that is decremented at each pass through a Layer 3 forwarding device.

TKIP *See* Temporal Key Integrity Protocol.

token bucket A conceptual model used by shapers and policers to represent their internal logic.

ToS byte *See* Type of Service byte.

totally NSSA area A type of OSPF NSSA area for which neither external (type 5) LSAs are introduced, nor type 3 summary LSAs; instead, the ABRs originate and inject default routes into the area. External routes can be injected into a totally NSSA area.

totally stubby area A type of OSPF stub area for which neither external (type 5) LSAs are introduced, nor type 3 summary LSAs; instead, the ABRs originate and inject default routes into the area. External routes cannot be injected into a totally stubby area.

traffic contract In shaping and policing, the definition of parameters that together imply the allowed rate and bursts.

transient multicast group Multicast addresses that are not assigned by IANA.

transit network (OSPF) A network/subnet over which two or more OSPF routers have become neighbors, thereby being able to forward packets from one router to another across that network.

transit router (OSPF) A router that is allowed to receive a packet from an OSPF router and then forward the packet to another OSPF router.

transmit power The signal strength of the RF signal at the output of the radio card or access point transmitter, before being fed into the antenna. Measured in milliwatts, watts, or dBm.

Trap In the context of SNMP, the Trap command is sent by an SNMP agent, to a manager, when the agent wants to send unsolicited information to the manager. Trap is not followed by a Response message from the receiving SNMP manager.

Triggered Extensions to RIP for On-Demand Circuits Defined in RFC 2091, the extensions define how RIP may send a full update once, and then send updates only when routes change, when an update is requested, or when a RIP interface changes state from down to up.

triggered updates A routing protocol feature for which the routing protocol sends routing updates immediately upon hearing about a changed route, even though it may normally only send updates on a regular update interval.

trunking Also called VLAN trunking, a method (using either the Cisco ISL protocol or the IEEE 802.1Q protocol) to support carrying traffic between switches for multiple VLANs that have members on more than one switch.

TTL *See* Time to Live.

TTL scoping Controls the distribution of multicast traffic by checking the TTL values configured on the interfaces. It forwards the multicast packet only on those interfaces whose configured TTL value is less than or equal to the TTL value of the multicast packet.

Type of Service byte A 1-byte field in the IP header, originally defined by RFC 791 for QoS marking purposes.

U/L bit The second most significant bit in the most significant byte of an Ethernet MAC address, a value of binary 0 implies that the address is a Universally Administered Address (UAA) (also known as Burned-In Address [BIA]), and a value of binary 1 implies that the MAC address is a locally configured address.

UDLD UniDirectional Link Detection. A protection against problems caused by unidirectional links between two switches. Uses messaging between switches to detect the loop, err-disabling the port when the link is unidirectional.

unicast MAC address Ethernet MAC address that represents a single NIC or interface.

Update (EIGRP) An EIGRP message that informs neighbors about routing information. Update messages require an Ack.

Update timer With RIP, the regular interval at which updates are sent. Each interface uses an independent timer, defaulting to 30 seconds.

UplinkFast Cisco-proprietary STP feature in which an access layer switch is configured to be unlikely to become Root or to become a transit switch. Also, convergence upon the loss of the switch's Root Port takes place in a few seconds.

upstream router From one multicast router's perspective, the upstream router is another router that has just forwarded a multicast packet to that router.

User Priority A 3-bit field in an 802.1Q header used for marking frames.

variance An integer setting for EIGRP and IGRP. Any FS route whose metric is less than this variance multiplier times the successor's metric is added to the routing table, within the restrictions of the **maximum-paths** command.

Variable-Length Subnet Masking A strategy for subnetting a classful network for which masks/prefixes are different for some subnets of that one classful network.

VC *See* Virtual circuit.

violate A category used by a policer to classify packets relative to the traffic contract. These packets are considered to be above the traffic contract in all cases.

Virtual circuit A logical concept that represents the path over which frames travel between DTEs. VCs are particularly useful when comparing Frame Relay to leased physical circuits.

Virtual LAN A group of devices on one or more LANs that are configured (using management software) so that they can communicate as if they were attached to the same wire, when, in fact, they are located on a number of different LAN segments. Because VLANs are based on logical instead of physical connections, they are extremely flexible.

virtual link With OSPF, the encapsulation of OSPF messages inside IP, to a router with which no common subnet is shared, for the purpose of either mending partitioned areas or providing a connection from some remote area to the backbone area.

Virtual Router Redundancy Protocol A standard (RFC 3768) feature by which multiple routers can provide interface IP address redundancy so that hosts using the shared, virtual IP address as their default gateway can still reach the rest of a network even if one or more routers fail.

VLAN *See* Virtual LAN.

VLAN Trunking Protocol A Cisco-proprietary protocol, used by LAN switches to communicate VLAN configuration.

VLSM *See* Variable-Length Subnet Masking.

VoFR *See* Voice over Frame Relay.

Voice over Frame Relay Defined in FRF.11, an FR VC that uses a slightly varied header, as compared with FRF.3 data VCs, to accommodate voice payloads directly encapsulated inside the Frame Relay LAPF header.

VRRP *See* Virtual Router Redundancy Protocol.

VTP *See* VLAN Trunking Protocol.

VTP pruning VTP process that prevents the flow of broadcasts and unknown unicast Ethernet frames in a VLAN from being sent to switches that have no ports in that VLAN.

weight A local Cisco-proprietary BGP setting that is not advertised to any peers. A larger value is considered to be better.

weighted fair queuing (WFQ) A Cisco IOS queuing tool most notable for its automatic classification of packets into separate per-flow queues.

weighted round-robin (WRR) A queuing scheduler concept, much like CQ's scheduler, in which queues are given some service in sequence. This term is often used with queuing in Cisco LAN switches.

well-known discretionary A characterization of a BGP path attribute in which all BGP implementations must support and understand the attribute (well known), but BGP Updates can either include the attribute or not depending on whether a related feature has been configured (discretionary).

well-known mandatory A characterization of a BGP path attribute in which all BGP implementations must support and understand the attribute (well known), and all BGP Updates must include the attribute (mandatory).

WEP *See* Wired Equivalent Privacy.

window Typically used by protocols that perform flow control (like TCP), a TCP window is the number of bytes that a sender can send before it must pause and wait for an acknowledgement of some of the yet-unacknowledged data.

Wired Equivalent Privacy The initial 802.11 common key encryption mechanism; vulnerable to hackers.

wireless LAN controller Controls access to the Internet in public wireless LANs.

Wireless LAN Threat Defense Solution An intrusion detection system that safeguards the wireless LAN from malicious and unauthorized access.

WLSE Cisco Wireless LAN Solution Engine. A centralized and required network management solution for Cisco Aironet SWAN solutions.

WPA Wi-Fi Protected Access. A security standard that includes both TKIP and AES and was ratified by the Wi-Fi Alliance.

Yellow Alarm A T1 alarm state that occurs when a device receives a Yellow Alarm signal. This typically means that the device on the other end of the line is in a red alarm state.

Index

Numerics

10BASE2, 22
10BASE5, 22
10BASE-T, 22
802.11. *See* IEEE 802.11
802.11a, 788-789
802.11b, 789-790
802.11g, 790-791
802.11n, 791
802.1D Spanning Tree Protocol. *See* STP
802.1Q
PVST+, 69
VLAN trunking, 44-45
configuration, 45-46
802.1Q-in-Q tunneling, 51-52
802.1X, 764-766
configuration, 766
802.2 LLC, Type fields, 18

A

AAA (authentication, authorization, and accounting), 747
authentication methods, 748-750
CLI, 747
groups of AAA servers, 750-751
overriding defaults for login security, 751
aaa authentication command, 749-750
aaa authentication ppp default, 752
ABRs (Area Border Routers), 276
stubby areas, 287
access lists, statements, 771
access points, 795, 832
access ports, protecting, 82-83
ACEs (Access Control Entries), 770
IP ACL, 770-771
ACK flags, 156
ACKs, 799
ACL, rate-limit ACL, 580
ACS (Cisco Secure Access Control Server), 747
active and not pruned VLANs, 48
active mode, FTP, 161-162
active routes (EIGRP), 243-245
stuck-in-active state, 245-246
active scanning, infrastructure mode, 799
AD (administrative distance), 317, 323-324
preventing suboptimal routes, 335-338
with route tags, 338-340
ad hoc mode, 800-801
wireless LANs, 794
adapters, wireless LAN client adapters, 833
adaptive shaping, 565
FRTS, 570
adaptive shaping, Frame Relay, 614
adding
default routes to BGP, 381-382
eBGP routes to IP routing tables, 392-393

iBGP routes to IP routing tables, 394-409
multiple BGP routes to IP routing tables, 450
address formats, Ethernet, 17-18
Address Resolution Protocol. *See* ARP
addresses
Ethernet, 16-17
unicast addresses, 16
inappropriate IP addresses, 774-775
IP. *See* IP addresses
MAC addresses
mapping to multicast IP addresses, 642-643
overriding, 18
tables, displaying, 53, 86, 197, 778
multicast IP addresses. *See* multicast IP addresses
adjacencies, EIGRP, 233-236
adjacency tables, 179
ARP and inverse ARP, 179-180
adjacent-layer interaction, 857
administration, SNMP, 164
administrative distance. *See* AD
administrative scoping, 690
administrative weight, 456-457
advertising BGP routes to neighbors, 383
BGP Update message, 383-384
determining contents of updates, 384-386
impact of decision process and NEXT_HOP, 386-391
AES (Advanced Encryption Standard), 810, 812
aggregatable global unicast addresses, 120-121
aggregate-address command, 378-379, 423, 462
BGP route summarization, 429-430
aggregate-address suppress-map command, 429
AID (association identifier), 799
Air/RF scanning and monitoring, CiscoWorks WLSE, 836
AIS (Alarm Indication Signal), 594
alarms, T1, 594
allocation of subnets, 105-106
allow-default keyword, 774
allowed and active VLANs, 48
allowed VLANs, 48
AMI (Alternate Mark Inversion), 592
versus B8ZS, 593
amplitude, 815
antenna diversity, 821
antennas, 795-796
Anycast RP with MSDP, 726-727
area authentication, 301
OSPF, 303
Area Border Routers (ABRs), 276
area filter-list, 298
area range command, 299
area stub command, 288
area virtual-link command, 303

ARP (Address Resolution Protocol), 137-139, 179-180
DAI, 758
gratuitous ARPs, 759
AS_CONFED_SEQ, 399, 439, 442
AS_CONFED_SEQ the show ip bgp command., 445
AS_PATH, 448
shortest AS_PATH, 459-460
prepending and route aggregation, 461-463
removing private ASNs, 460
AS_PATH filters, matching AS_PATHs, 436-439
AS_PATH PA, 446
AS_PATH segment types, 431-433
AS_PATHS, 360
AS_SET and AS_CONFED_SEQ, 445
filtering BGP updates, 430-431
AS_PATH filters, 436-439
AS_SET and AS_CONFED_SEQ, 439, 442, 445
BGP AS_PATH and AS_PATH segment types, 431-433
regular expressions, 433-434
and manual summaries, BGP tables, 378-381
AS_SEQ, 378
AS_SET, 439, 442, 445
ASBRs (Autonomous System Boundary Routers), 276
ASNs (autonomous system numbers), 360, 432, 459, 640
removing private ASNs, 460-461
Ass, 399
multiple adjacent AS, 466
single adjacent AS, 465
assert messages, PIM, 703-704
as-set option, 379
assigning interfaces to VLANs, 35
assisted site surveys, CiscoWorks WLSE, 835
association identifier (AID), 799
Assured Forwarding (AF) PHBs, 491-492
attenuation, RF signals, 816
authentication
configuring OSPF, 301-303
EIGRP, 250
RIP, 216-219
TCP/IP, 163
authentication methods, 748-750
authentication, authorization, and accounting. *See* AAA
auto-cost reference-bandwidth, 294
automatic access point configuration, CiscoWorks WLSE, 834
automatic medium-dependent interface (Auto-MDIX), 9
Auto-MDIX (automatic medium-dependent interface crossover), 9
auto-negotiation, Ethernet, 9
Autonomous System Boundary Routers (ASBRs), 276
autonomous system numbers. *See* ASNs
autonomous system path. *See* AS_PATHs
Auto-RP, 721-723
autosummarization
EIGRP, 250
RIP, 214-216
auto-summary, impact on redistributed routes and network command, 375-377
auto-summary command, 378
aux, 751

B

B8ZS (Bipolar 8 Zero Substitution), 592
versus AMI, 593
BackboneFast, optimizing STP, 73, 75
backdoor routes, IP routing tables, 393-394
Backward Explicit Congestion Notification (BECN), 559, 614
bandwidth, 815
CBWFQ, limiting, 532-534
LLQ, 537
bandwidth command, 529, 532, 564
bandwidth percent, configuring shaping, 564
bandwidth percent command, 533
bandwidth remaining percent command, 533
basic service set (BSS), 792
Bc (committed burst), 556
CB Policing defaults, 577
Be (excess burst size), 556
CB Policing defaults, 577
traffic shaping, 557
beamwidth, 797
BECN (Backward Explicit Congestion Notification), 559, 614
Frame Relay congestion, 614
BGP (Border Gateway Protocol), 276
advertising routes to neighbors, 383
BGP Update message, 383-384
determining contest of updates, 384-386
impact of decision process and NEXT_HOP, 386-391
AS_PATHs, 360
command references, 411, 479
filtering updates based on NLRI, 424-427
route maps, 427
soft reconfiguration, 428
maximum-paths command, 471-472
PAs, 360, 410
route maps, match and set commands, 479
subcommands for confederations, 402
bgp always-compare-med, 466
BGP AS_PATH, 431-433
BGP COMMUNITY PA, 472-474
filtering NLRI using COMMUNITY values, 479
matching with community lists, 474-475
removing COMMUNITY values, 475-476, 479
bgp confederation identifier command, 401
BGP decision process, 446-448
adding multiple BGP routes to IP routing tables, 450
BGP PAs, 453-454, 456
mnemonics for memorizing, 450, 452
tiebreakers, 448-450
maximum-paths command, 467
bgp deterministic-med command, 466
BGP filtering tools, 417, 423-424
BGP message types, 368-369
BGP metric, 464
BGP neighbor states, 368
resetting peer connections, 369-370
BGP neighbors, 361-362
checks before becoming neighbors, 366-368
eBGP, 365-366
iBGP, 362-365
BGP Open messages, 361
BGP PAs, 446
BGP decision process, 453-454, 456
BGP path attributes BGP path attributes, 446
BGP policies, configuring, 452
Step 0 NEXT_HOP reachable, 456
Step 1 Administrative weight, 456-457

Step 2 Highest Local Preference (LOCAL_PREF), 457-458
Step 3 Choose Between Locally Injected Routes Based on, 458
Step 4 Shortest AS_PATH, 459-463
Step 5 Best ORIGIN PA, 463
Step 6 Smallest Multi-Exit Discriminator, 464
Step 7 Prefer Neighbor Type eBGP over iBGP, 466
Step 8 Smallest IGP Metric to the NEXT_HOP, 466
Step 9 Lowest BGP Router ID of Advertising Router, 467
Step 10 Lowest Neighbor ID, 467, 469
BGP prefixes, 370
BGP route summarization, aggregate-address command, 429-430
BGP router ID of advertising router, 467
BGP routes, 370
BGP Routing Information Base (RIB), 370
BGP routing policies, 417

BGP synchronization, 395-398
disabling, 398
BGP tables, 370
injecting routes/prefixes, 370
impact of auto-summary on redistributed routes and, 375-377
network command, 370-371, 373
redistributing from IGP, static or connected routes, 373-375
manual summaries and AS_PATH, 378-381
routes
adding default routes, 381-382
ORIGIN, 382-383
BGP Update messages, 361
advertising BGP routes to neighbors, 383-384
determining contents of updates, 384-386
impact of decision process and NEXT_HOP, 386-391
BGP updates
filtering by matching AS_PATHS, 430-431
AS_PATH filters, 436-439
AS_SET and AS_CONFED_SEQ, 439, 442
BGP AS_PATH and AS_PATH segment, 431-433
regular expressions, 433-434
rules for routes, 392
bidirectional PIM, 729-730
binary method
exclusive summary routes, 110
inclusive summary routes, 108-109
subnet numbers, broadcast addresses, range of IP addresses, determining, 98-99, 102-104
binary phase shift (BPSK), 818
binary values, converting to decimal and hexadecimal values, 955, 959
Bipolar 8 Zero Substitution (B8ZS), 592
Bipolar Violations (BPVs), 593
blocking transitioning to forwarding, STP, 67-68
blocking state, 61
Spanning Tree, 68
bogons, 774
boot fields, 859
boot sequences, IOS software, 858-859
boot system command, IOS, 859
boot system flash, 859
boot system ROM, 859
boot system tftp, 860

BOOTP, 139-141
BootStrap Router (BSR), 721
Border Gateway Protocol (BGP), 276
BPDU (bridge protocol data unit), 62, 70
BPDU Guard, 82-83
enabling, 754
BPSK (binary phase shift), 818
BPVs (Bipolar Violations), 593
bridge protocol data unit (BPDU), 62
bridges, 797
wireless and workgroup bridges, 833
broadcast addresses, 16, 43
determining
binary method, 98-99
decimal method, 99-101
broadcast clients (NTP), 144
broadcast domains, 31
broadcast methods, 633
broadcast subnets, 98
BSR (BootStrap Router), 721, 724-725
BSS (basic service set), 792
buckets, refilling dual token buckets, 573
burst size, 509

C

C&M tools. *See* classification and marking tools
cabling, UTP cabling, 24
cabling standards, Ethernet, 24
calculating
metrics for types 1 and 2, 285-286
SN, 525
STP costs to determine RPs, 63
CAM (Content Addressable Memory), 66, 643
updating, 66-67
CAR (committed access rate), 551
CB Policing, 579-581
carrier detects, synchronous serial links, 594-595
carrier sense multiple access with collision avoidance (CSMA/CA), 809
Carrier Sense Multiple Access with Collision Detection (CSMA/CD), 10
carrier transitions, 595
catalyst IOS commands for catalyst switch configuration, 23
Category 5 wiring, 8-9
CatOS, 38
CB Marking tool, 500-505
CoS and DSCP, 505-507
locations for marking, 508-509
NBAR, 507-508
CB Policing (Class-Based Policing), 551, 571
CAR, 579-581
command references, 584
configuring, 575
defaults for Bc and Be, 577
dual-rate policing, 577-578
multi-action policing, 578
policing by percentage, 578-579
policing subsets of traffic, 576-577
single-rate, three-color policing, 575-576
single-rate, three-color policing, 573
single-rate, two-color policing, 571-572
two-rate, three-color policing, 573-575
CB Shaping (Class-Based Shaping), 551
command references, 582
configuring, 559-561
to peak rates, 565
CBT (Core-Based Tree), 687
CBWFQ (class-based WFQ), 521, 529-530, 532, 538
bandwidth, 532-534
command references, 530
features of, 529

CCM (Cisco CallManager), 839
CCP (Compression Control Protocol), 602
CDP, disabling, 754
CDT (congestive discard threshold), 526
ceased updates (RIP), 210-212
CEF (Cisco Express Forwarding), 178
adjacency tables, ARP and inverse ARP, 179-180
FIB, 178
Cell Loss Priority (CLP) bit, 493
centralized firmware updates, CiscoWorks WLSE, 835
CGMP (Cisco Group Management Protocol), 635, 663-666, 668
join message process, 666
leave message, 667
messages, 668
change notification, STP topology, 66-67
channels
nonoverlapping channels, 802.11g, 790
RF channels, 803-804
characteristics of RF signals, 815-816
Checksum, 160
CIDR (classless interdomain routing), 111-112
CIR (committed information rate), 556
Cisco 2000 Series Wireless LAN Controller, 833
Cisco 2950 switches versus Cisco 3550 switches, 547-548
Cisco 3550, queuing, 545-546
Cisco 3550 switches, 543-545
versus Cisco 2950 switches, 547-548
Cisco 4100 Series Wireless LAN Controller, 833
Cisco Aironet 1100 Series, 832
Cisco Aironet 1130AG Series, 832
Cisco Aironet 1200 Series, 832
Cisco Aironet 1230AG Series, 832
Cisco Aironet 1300 Series, 832
Cisco Aironet 1300 Series Outdoor Access Point/Bridge, 833
Cisco Aironet 1400 Series Wireless Bridge, 833
Cisco Aironet 350 Series, 833
Cisco Aironet 350 Wireless LAN Client Adapter, 833
Cisco Aironet 5-GHz 54-Mbps Wireless LAN Client Adapter (CB20A), 833
Cisco Aironet 802.11a/b/g PCI Adapter, 833
Cisco Aironet 802.11a/b/g Wireless CardBus Adapter, 833
Cisco CallManager (CCM), 839
Cisco Express Forwarding (CEF), 178
Cisco Group Management Protocol. *See* CGMP
Cisco SAFE Blueprint, Layer 3 security, 768
Cisco SAFE Blueprint document, 752
Cisco. Secure Access Control Server (ACS), 747
CiscoWorks WLSE (Wireless LAN Solution Engine), 834
air/RF scanning and monitoring, 836
assisted site surveys, 835
automatic access point, 834
centralized firmware updates, 835
customizable thresholds, 835
dynamic grouping, 835
fault status, 836
IDS, 836
secure user interfaces, 836
security policy monitoring, 836
self-healing functions, 837
SSIDs, 835
troubleshooting, 837
VLAN configuration, 835
class maps, MQC classification with, 497-499

Class of Service (CoS) field, 493
Class Selector (CS) PHBs, 491
Class-Based Marking. *See* CB Marking tool
Class-Based Policing. *See* CB Policing
Class-Based Shaping. *See* CB Shaping
class-based WFQ. *See* CBWFQ
class-default queues, 529
classful compared to classless, 227
classful IP addressing, 94-95
subnets, 95-96
classful routing, 185-186
classification and marking tools, 485
CB Marking, 500-505
CoS and DSCP, 505-507
locations for marking, 508-509
NBAR, 507-508
CoS (Class of Service) field, 493
drop probability bits, 493
DSCP (Differentiated Services Code Point) field, 490-493
field locations, 494-495
IP Precedence (IPP) field, 489-490
MPLS Experimental (EXP) field, 494
MQC (Modular QoS CLI), 495-496
class maps, 497-499
match commands, 511-512
NBAR, 499-500
policers, 509-510
policy routing, 510
classless compared to classful, 227
classless interdomain routing (CIDR), 111-112
classless IP addressing, 94, 97
classless routing, 185-186
class-map command (MQC), 496
clear command, 370, 423
clear ip cgmp, 668
clearing
IP routing tables, EIGRP, 250
OSPF processes, 292-295
CLI
AAA, 747
passwords, 745-746
enable and username passwords, 746-747
CLI help features, 860-861
client hardware address, DHCP, 762
client mode (NTP), 143
client tracking, 828
code bits, TCP, 156
collision domains, Ethernet, 10-11
command references
BGP, 411, 479
CB Marking tool, 501
CB Policing, 584
CB Shaping, 582
CBWFQ, 530
EIGRP, 251-252
Frame Relay, 623
FRTS, 583
IP addresses, 126
IP ACL, 769
IP forwarding, 197
IP multicast routing, 732
match commands (MQC), 511-512
OSPF, 305-307
packet routing, 146-148
redistribution, 352
RIP, 225-226
STP, 86
synchronous serial links, 604
WFQ, 527
commands
aaa authentication, 749-750
aaa authentication ppp default, 752
aggregate-address, 378-379, 423, 462
BGP route summarization, 429-430
aggregate-address suppress-map, 429
area authentication, 301
OSPF, 303

area filter-list, 298
area range command, 299
area stub, 288
area virtual-link, 303
auto-cost reference-bandwidth, 294
auto-summary, 378
bandwidth, 529, 532, 564
bandwidth percent, 533
bandwidth remaining percent, 533
bgp always-compare-med, 466
bgp confederation identifier, 401
bgp deterministic-med, 466
boot system, 859
boot system flash, 859
boot system ROM, 859
boot system tftp, 860
clear, 370, 423
clear ip cgmp, 668
compress, 602
debug ip arp, 196
debug ip ospf adjacency, 301
debug ip policy, 195
debug policy, 196
default-information originate, 348-349, 382
DHCP snooping, 763
distance, 324, 394
distance router, 337
distribute-list command, 295-297
do, 182
enable, 745
enable password, 746
enable secret, 746
encapsulation, 50
encapsulation ppp, 597
fair-queue, 527
frame-relay class, 566
frame-relay fragment, 621
frame-relay fragment size, 621
frame-relay interface-dlci, 566, 615-616
frame-relay map, 183-184, 615-616
frame-relay mincir rate, 570
hold-queue, 527
hold-queue x out, 521
IOS, 862
ip access-group, 769
ip bgp-community new-format, 474
ip cef global configuration, 179
ip classless, 186, 345
ip community-list, 474, 479
ip default-network, 349-350
ip multicast-routing, 692, 708
ip ospf area, 295
ip ospf authentication, 301
ip ospf cost, 50, 293-294
ip ospf network, 269
ip pim dense-mode, 692
ip pim rp-address, 720
ip pim sparse-mode, 708
ip pim spt-threshold, 717
ip policy, 192
ip proxy-arp, 196
ip summary-address rip, 345
ip verify source command, 764
keyboard commands, 861
log-adjacency-changes detail, 292
login authentication, 751
map-class frame-relay, 566
map-class shape-with-LLQ, 620
match, 319-320
match as-path list-number, 439
match ip address, 192
match length, 192
maximum-paths, 450, 470, 472
BGP decision process tiebreakers, 466-467
max-metric router-lsa on-startup announce-time, 304
max-metric router-lsa on-startup wait-for-bgp, 304
max-reserved-bandwidth, 532

neighbor, 270, 274, 468
neighbor default-originate, 382
neighbor ebgp-multihop, 401, 469
neighbor filter-list command, 439
neighbor peer-group, 365
neighbor remote-as, 365-366
neighbor route-map, 439
neighbor shutdown, 370
neighbor weight, 456
network, 295
injecting prefixes/routes into BGP tables, 370-371, 373
network backdoor, 394
no auto-summary, 370
no frame-relay inverse-arp, 184
no ip classless, 186, 345
no ip directed-broadcast, 773
no ip route-cache cef, 179
no synchronization, 395
no terminal editing, 861
ospf auto-cost reference-bandwidth, 294
password, 745
police, 575, 578
police commands, 577
policy-map queue-voip, 563
port security configuration, 755
ppp authentication, 752
ppp multilink fragment-delay, 601
ppp multilink interleave, 600
prefix-list, BGP, 425
prefix-list commands, 322
priority, 535
priority-queue out, 543
radius-server host, 750
rate-limit, 579
recalling, 861
redistribute, 321
redistribute command, 324-325
redistribute connected, 458
redistribute ospf, 328
redistribute static, 347-348
route-map, 317-319
BGP, 425
router bgp, 365, 401
service password encryption, 746
service password-encryption, 303, 747
service-policy, 532
service-policy out, 538
service-policy output, 532, 559
service-policy output policy-map-name, 564
set, 320
set as-path prepend command, 461
set community none, 476
set fr-de, 615
shape, 559, 561
shape average, 565
shape fecn-adapt, 614
shape peak mean-rate, 565
shape percent, 564
show interface trunk command, 48
show interfaces, 595
show ip, 23
show ip arp, 195
show ip bgp, 382, 439, 453-455
show ip bgp neighbor advertised-routes, 388
show ip bgp neighbor neighbor-id advertised routes, 439
show ip bgp neighbor neighbor-id received routes, 439
show ip bgp regexp expression, 439
show ip mroute, 692, 714
show ip ospf border-routers, 283
show ip ospf database, 281
show ip ospf database summary link-id, 283
show ip ospf neighbor, 262
show ip ospf statistics, 283
show ip route, 290
show queue, 528

spanning-tree portfast, 79
spanning-tree vlan, 73
summary-address, 344
switchport access vlan, 38, 43
switchport mode, 49
switchport nonegotiate, 49
switchport port-security maximum, 756
switchport trunk allowed, 48
switchport trunk encapsulation, 49
tacacs-server host, 750
terminal editing, 861
terminal history size, 860
traffic-shape fecn-adapt, 614
traffic-rate, FRTS configuration, 567-568
username password, 747
username, 748
wrr-queue dscp-map, 546
wrr-queue random detect, 546

committed access rate. *See* CAR
committed burst (Bc), 556
committed information rate (CIR), 556
Common Spanning Tree (CST), 69
community lists, matching with COMMUNITY, 474-475
COMMUNITY PA, BGP, 473-474
filtering NLRI using COMMUNITY values, 479
matching with community lists, 474-475
removing COMMUNITY values, 475, 479
community VLANs, 37
comparing
802.11 standards, 791
BGP, prefix lists, distribute lists, and route maps, 428-429
Cisco 3550 and 2950 switches, 547-548
IGMP versions, 661
queuing tools, 520
wireless security, 813
compatibility, trunk configuration compatibility, 48-49
compress command, 602
compression
Frame Relay payload compression, 619-620
PPP, 601-602
header compression, 602-603
layer 2 payload compression, 602
Compression Control Protocol (CCP), 602
confederation eBGP peers, 399
confederations
BGP subcommands, 402
IP routing tables, 399-401
configuring, 401-404
configuration
CB Marking tool, 500-505
CoS and DSCP, 505-507
locations for marking, 508-509
NBAR, 507-508
EIGRP, 246-249
LCP, 597-598
MQC (Modular QoS CLI), 495-496
class maps, 497-499
NBAR, 499-500
RIP, 213-214
authentication, 216-219
autosummarization, 214-216
distribution list and prefix list filtering, 222-224
next-hop and split horizon features, 219-220
offset lists, 220-222
static configuration of Frame Relay mapping, 183-184
trunks, 53, 779
configuration mode
creating VLANs, 35-36
putting interfaces into VLANs, 34-35

configuration register, 858-859
configurations, switch ports, 12-14
configuring
BGP policies. *See* BGP policies, configuring
CB Policing, 575
defaults for Bc and Be, 577
dual-rate policing, 577-578
multi-action policing, 578
policing by percentage, 578-579
policing subsets of traffic, 576-577
single-rate, three-color policing, 575-576
CB Shaping, 559-561
confederations, IP routing tables, 401-404
Frame Relay, 615-619
FRTS, 565-567
adaptive shaping, 570
setting parameters, 568-569
traffic-rate command, 567-568
with LLQ, 569-570
MED
multiple adjacent AS, 465-466
single adjacent AS, 465
MLS, 188-191
MST, 81
OSPF, 290-292
alternatives to OSPF network command, 295
authentication, 301-303
costs, 292-295
stub router, 303-304
virtual links, 299-301
PortChannels, 77-78
RADIUS server groups, 750
route maps with route-map command, 317-319
shaping by bandwidth percent, 564
STP, 70-73
TCP intercept, 777
VLAN trunking on routers, 49-51
VLANs, 31
storing, 43-44
VLAN database configuration mode, 32-34
VTP, 40-41
extended-range VLANs, 42
normal-range VLANs, 42
WFQ, 527-528
WRED, 542
conforming packets, 571
congestion, Frame Relay, 613
adaptive shaping, FECN, and BECN, 614
DE big, 615
congestion management. *See* queuing
congestion window (CWND), 158
congestive discard threshold (CDT), 526
connecting with networks, infrastructure mode, 799
connections, TCP, 155-156
console, 751
Content Addressable Memory (CAM), 66, 643
convergence
EIGRP, 240-241
going active on routes, 243-245
input events and local computation, 241-243
limiting query scope, 246
stuck-in-active state, 245-246
RIP, 205-206
ceased updates, 210-212
steady-state operation, 206-208
triggered updates and poisoned routes, 208-210
tuning, 212-213
converging to STP topology, 65-66

conversion, decimal-to-hexadecimal-to-binary values, 955, 959
Core-Based Tree (CBT), 687
CoS (Class of Service) field, 493
CB Marking tool, 505-507
CQ (custom queuing), 523-524
cross-over cables, 9
CSMA/CA (carrier sense multiple access with collision avoidance), 809
CST (Common Spanning Tree), 69
CSU/DSU, 594
custom queuing (CQ), 523-524
customizable thresholds, CiscoWorks WLSE, 835
cut-through, switches, 23
CWND (congestion window), 158

D

D4, 592
DAI (dynamic ARP inspection, 758-761
data communications equipment (DCE), 594
data definition, SNMP, 164
Data Link Connection Identifier (DLCI), 610-611
data rates, wireless configuration parameters (IEEE 802.11), 804-805
data terminal equipment (DTE), 594
data transfers, infrastructure mode, 799-800
Database Description (DD or DBD), 264
databases, OSPF. *See* OSPF database exchange
DBD (Database Description), 264
DCE (data communications equipment), 594
DCF (distributed coordination function), 809-810
DD (Database Description), 264
DD messages, flooding LSA headers to neighbors, 264
DE (Discard Eligibility) bit, 615
Dead, 308
debug ip arp, 196
debug ip ospf adjacency, 301
debug ip policy command, 195
debug policy, 196
decimal method
inclusive summary routes, 109-110
subnet numbers, broadcast addresses, range of IP addresses determining, 99-101, 104-105
decimal values, converting to hexadecimal and binary values, 955, 959
deep packet inspection, 499
Deering, Dr. Steve, 632
default routes, 345-346
adding to BGP, 381-382
creating using route summarization, 350-351
default-information originate command, 348-349
ip default-network command, 349-350
default-information originate command, 348-349, 382
dense mode, multicast forwarding, 684-685
dense-mode routing protocols, 690
DVMRP, 706
MOSPF, 706
PIM-DM, 691
forming adjacencies with PIM hello messages, 691
Graft messages, 700-702
Prune messages, 693-695
reacting to failed links, 695-697

rules for pruning, 697-699
source-based distribution trees, 692-693
steady-state operation and state refresh messages, 699-700
deny clauses, route maps, 333
designated routers. *See* DR
designated port (DP), 65
determining, 64-65
designated routers, PIM, 704
designated switches, 64
DHCP, 139-141
DHCP snooping, 761-762
commands, 763
DHCP snooping binding table, 761
Differentiated Services Code Point (DSCP) field, 490-493
DiffServ, RFCs, 512
Diffusing Update Algorithm (DUAL), 245
Digital Signal Level 0 (DS0), 592
direct sequence spread spectrum (DSSS), 788
directed broadcasts, 772-774
disabling
BGP synchronization, 398
CDP and DTP, 754
InARP, 184-185
discard categories, WRED, 539
Discard Eligibility (DE) bit, 493, 615
discarding logic, 540
discovering neighbors, hello messages, 263-264
discretionary PAs, 446
discriminators, multi-exit discriminators, 464
distance command, 324, 394
preventing suboptimal routes, 336
distance router command, 337
Distance Vector Multicast Routing Protocol (DVMRP), 635, 687, 706
distribute lists versus prefix lists and route maps (BGP), 428-429
distributed coordination function (DCF), 809
distribute-list command, 295-297
distribution list filtering, RIP, 222-224
DIX, Type fields, 18
DIX Ethernet Version 2, 22
DLCI (Data Link Connection Identifier), 610-611
do command, 182
domains, broadcast domains, 31
downstream routers, 697
DP (designated port), 65
DR (designated routers)
on LANs, 266
election, 268-269
optimizing, 266-268
on WANs, 269
OSPF network types, 269
drop probability bits, 493
DS0 (Digital Signal Level 0), 592
DSCP (Differentiated Services Code Point) field, 490-493
CB Marking tool, 505-507
DSCP-based WRED, 541
DSSS (direct sequence spread spectrum), 788, 818
DTE (data terminal equipment), 594
DTP (Dynamic Trunk Protocol), 45
disabling, 754
DUAL (Diffusing Update Algorithm), 245
dual stacks, IPv6 configuration, 121-123
dual-rate policing, CB Policing configuration, 577-578
duplex, Ethernet, 9
DVMRP (Distance Vector Multicast Routing Protocol), 635, 687, 706
dynamic ARP inspection (DAI), 758-761

dynamic grouping, CiscoWorks WLSE, 835
dynamic NAT, configuration, 118-119
dynamic NAT (without PAT), 116-117
Dynamic Trunk Protocol (DTP), 45

E

E1, 592 versus T1, 593
EAP (Extensible Authentication Protocol), 764
802.1X, 764-766
EAPoL (EAP over LAN), 764
eBGP (external BGP), 362, 365-366
over iBGP, 466
eBGP routes, adding to IP routing tables, 392-393
egress blocking, 555, 613
egress queuing, Cisco 3550 switches, 543-545
EIGRP, 233
adjacencies, 233-236
authentication, 250
autosummarization, 250
clearing IP routing tables, 250
command reference, 251-252
configuration, 246-249
convergence, 240-241
going active on routes, 243-245
input events and local computation, 241-243
limiting query scope, 246
stuck-in-active state, 245-246
IS-IS configuration for creating default summary routes, 351
load balancing, 249
offset lists, 250
packet types, 252
route filtering, 250
split horizon, 250
static routes, redistribute static, 347
topology table, 238-240
updates, 236-238
EIGRP route summarization, 344
electing root switches, STP, 61-63
election, DR election on LANs, 268-269
enable command, 745
enable password command, 746
enable secret command, 746
enabling Root Guard and BPDU Guard, 754
encapsulation, Frame Relay, 612-613
encapsulation command, 50
encapsulation ppp command, 597
encrypted enable passwords, 748
enterprises, applying wireless LANs, 837
security, 837-838
voice services, 839
EoMPLS (Ethernet over MPLS), 51
error recovery, TCP, 157
ESF (Extended Superframe), 592
established keyword, 771
ACL, 776
Ethernet
address formats, 17-18
addresses, 16-17
unicast addresses, 16
auto-negotiation, 9
cabling standards, 24
Category 5 wiring, 8-9
collision domains, 10-11
cross-over cables, 9
CSMA/CD, 10
duplex, 9
frames, 14
framing and addressing, 14
header fields, 15
multicast Ethernet frames, 16
packets, 14
RJ-45 pinouts, 8-9
speed, 9

switch buffering, 10-11
switch port configuration, 12-14
twisted pairs, 8-9
Type fields, 18
types, 24
types of Ethernet, 22
VLANs. *See* VLANs
Ethernet over MPLS (EoMPLS), 51
EUI-64 format, 121
exceeding packets, 571
excess burst size (Be), 556
exclusive summary routes, 108
binary method, 110
Expedited Forwarding (EF) PHBs, 492-493
Extended Superframe (ESF), 592
extended-range VLANs, 42
Extensible Authentication Protocol. *See* EAP
external BGP. *See* eBGP

F

failed links, reacting to (PIM-DM), 695-697
fair-queue command, 527
Fast Link Pulses (FLP), 9
Fast Secure Roaming (FSR), 828
fast switching, IP forwarding, 178
FastE, 22
fast-switching cache, 178
fault status, CiscoWorks WLSE, 836
FCC rules, RF signals, 819
FCS (frame check sequence), 177
FD (feasible distance), 240
FDX (full duplex), 9
feasibility conditions, 241
feasible distance (FD), 240
FECN (Forward Explicit Congestion Notification), 614
FHSS (frequency hopping spread spectrum), 788, 817
FIB (Forwarding Information Base), 178
fields
classification and marking tools
Cell Loss Priority (CLP) field, 493
Class of Service (CoS) field, 493
Differentiated Services Code Point (DSCP) field, 490-493
Discard Eligibility (DE) field, 493
IP Precedence (IPP) field, 489-490
locations of, 494-495
MPLS Experimental (EXP) field, 494
Type fields, 18
FIFO (first-in, first-out), 519
FIFO queuing, 521
filtering
BGP filtering tools. *See* BGP filtering tools
BGP updates by matching AS_PATHs, 430-431
BGP updates based on NLRI, 424-427
route maps, 427
soft reconfiguration, 428
BGP updates by matching AS_PATHs
AS_PATH filters, 436-439
AS_SET and AS_CONFED_SEQ, 439, 442
BGP AS_PATH and AS_PATH segment, 431-433
regular expressions, 433-434
distribution list and prefix list filtering (RIP), 222-224
NLRI, using COMMUNITY values, 479

OSPF, 295
ABR LSA type 3 filtering, 297-298
distribute-list command, 295-297
subnets of summaries, using aggregate-address command, 429-430
finding RPs, 720, 728
Anycast RP with MSDP, 726-727
Auto-RP, 721-723
BSR, 724-725
finish time (FT), 525
flags
mroute, 735
TCP, 156
flood (pacing), 309
flooding LSA headers to neighbors, 264
flow, 524
FLP (Fast Link Pulses), 9
ForeSight, 559
Forward Explicit Congestion Notification (FECN), 614
forwarding, transitioning from blocking (STP), 67-68
Forwarding Information Base (FIB), 178
Forwarding state (Spanning Tree), 68
fraggle attacks, 774
fragmentation
Frame Relay, 621-622
wireless configuration parameters, IEEE 802.11, 808
fragment-free, switches, 23
frame check sequence (FCS), 177
Frame Relay, 610
command references, 623
configuring, 615-619
congestion, 613
adaptive shaping, FECN, and BECN, 614
DE bit, 615
DLCI, 610-611
fragmentation, 621-622
headers and encapsulation, 612-613
LMI, 611-612
payload compression, 619-620
traffic shaping, 559
Frame Relay Forum (FRF), 610
Frame Relay Forum Implementation Agreement 9 (FRF.9), 619
Frame Relay Inverse ARP, 180-183
disabling, 184-185
Frame Relay Inverse ARP (InARP), 180-183
Frame Relay mapping, static configuration, 183-184
Frame Relay Traffic Shaping. *See* FRTS
frame-relay class, 566
frame-relay fragment command, 621
frame-relay fragment size command, 621
frame-relay interface-dlci, 566, 615-616
frame-relay map, 615-616
frame-relay map commands, 183-184
frame-relay mincir rate command, 570
frames
Ethernet, 14
multicast Ethernet frames, 16
framing T1, 592-593
frequency hopping spread spectrum (FHSS), 788
frequency spectrum, 815
FRF (Frame Relay Forum), 610
FRF.12, 622
FRF.9 (Frame Relay Forum Implementation Agreement 9), 619
FRTS (Frame Relay Traffic Shaping), 551, 565
command references, 583
configuring, 565-567
adaptive shaping, 570
setting parameters, 568-569
with LLQ, 569-570
with traffic-rate command, 567-568

FSR (Fast Secure Roaming), 828
FT (finish time), 525
FTP (file transfer protocol), active and passive mode, 161-162
full duplex (FDX), 9
full-featured IOS, 858
functions of CBWFQ, 529

G–H

gain, RF signals, 816
gang of four, 610
GDA (Group Destination Address), 664
GigE, 22
GLBP (Gateway Load Balancing Protocol), 141-143
global addresses, 120-121
global addressing, 611
GlOP, 640
GLOP addressing, multicast IP addresses, 640
Goodbye messages (EIGRP), 236
Graft messages, PIM-DM, 700-702
granted window, 157
gratuitous ARPs, 759
Group Destination Address (GDA), 664
group radius, 750
group tacacs+, 750
groups of AAA servers, 750-751
group-specific query messages, IGMPv2, 654-656

half duplex (HDX), 9
hardware for SWAN, 831
hardware queues, 519
hashed enable passwords, 748
HDLC (High-Level Data Link Control), 595-596
HDX (half duplex), 9
header compression, PPP, 602-603
header fields, Ethernet, 15
header format for IP addresses, 126-128
headers
  Frame Relay, 612-613
  LSA headers, 265
  TCP, 159
Hello, 308
hello intervals (EIGRP), 234
hello messages
  discovering neighbors, 263-264
  forming adjacencies with PIM hello messages, 691
hellos, EIGRP, 233-236
help
  CLI help features, 860-861
  IOS commands, 861
hexadecimal values, converting to decimal and binary values, 955, 959
High-Level Data Link Control (HDLC), 595-596
HLEN (Offset), 160
hold-queue command, 527
hold-queue limit, 526
hold-queue x out, 521
home wireless LANs, 842-844
host membership query functions, IGMPv1, 646-647
host membership report functions, IGMPv1, 647-648
  solicited host membership report, 648-649
  unsolicited host membership report, 650
host part (classful IP addressing), 94
host part (IP addressing), 97
HSRP (Hot Standby Router Protocol), 141-143
HTTP protocol, TCP/IP, 856

IANA (Internet Assigned Numbers Authority), 638
iBGP (internal BGP), 362-365
over eBGP, 466
iBGP routes, adding to IP routing tables, 394-396
BGP synchronization and redistributing routes, 396-398
confederations, 399-401
configuring confederations, 401-404
disabling BGP synchronization, 398
RRs, 404-409
ICMP (Internetwork Control Message Protocol), 134-135
ICMP Redirect message, 137
ICMP Time Exceeded message, 136
ICMP Unreachable message, 135
port numbers, 771
ICMP echo replies, 134
ICMP echo requests, 134
ICMP Redirect message, 137
ICMP Time Exceeded message, 136
ICMP Unreachable message, 135
IDS (intrusion detection system), 829-830
CiscoWorks WLSE, 836
neighbor ID, 468-469
maximum-paths command, 471-472
IEE 802.1D STP Timers, 85
IEE 802.3u, 22
IEEE 802.2, 22
IEEE 802.3, 22
IEEE 802.3ab, 22
IEEE 802.3z, 22
IEEE 802.11, 788
802.11a, 788-789
802.11b, 789-790
802.11g, 790-791
802.11n, 791
ad hoc mode. *See* ad hoc mode
comparing standards, 791
infrastructure mode. *See* infrastructure mode
RF signals. *See* RF signals
wireless configuration parameters, 801
data rates, 804-805
fragmentation, 808
power-save mode, 805-806
RF channels, 803-804
RTS/CTS, 806-808
SSID, 802-803
transmit power, 804
wireless hardware components, 794
access points, 795
antennas, 795-796
bridges, 797
radio cards, 795
radio frequency peripherals, 797
repeaters, 796
routers, 797
wireless medium access, 809-810
wireless security, 810
AES, 812
open system authentication, 812
shared key authentication, 812-813
TKIP, 811
VPNs, 813
WEP, 811
WPA, 812
wireless system configuration, 791
ad hoc mode configuration, 794
infrastructure mode configuration, 792-794
IGMP snooping, 669, 671-673
comparing all versions, 661
managing distribution of multicast traffic, 643-644
joining groups, 644

IGMP (Internet Group Management Protocol), 635, 645
joining groups, 670
RGMP, 674
IGMPv1, 645-646
host membership query functions, 646-647
host membership report functions, 647-648
solicited host membership report, 648-649
unsolicited host membership report, 650
interoperability with IGMPv2, 657-658
leave mechanisms, 651
querier, 651
routers, 658
timers, 659
IGMPv2, 651-653
interoperability with IGMPv1, 657-658
leave groups and group-specific query messages, 654-656
querier, 656
routers, 658
timers, 659
IGMPv3, 659-661
SSM, 640, 660
IGPs (Interior Gateway Protocols), 360
in keyword, 428
Inactivity, 308
inappropriate IP addresses, 774-775
InARP (Frame Relay Inverse ARP), 180-183
disabling, 184-185
inclusive summary routes, 107
binary method, 108-109
decimal method, 109-110
Individual/Group (I/G) bit, 17
Inform message, SNMP, 166
infrastructure mode, 798
connecting with networks, 799
data transfers, 799-800
roaming, 800
scanning, 798-799
wireless LANs, 792-794
infrastructure wireless LAN. *See* infrastructure mode
input events, EIGRP, 241-243
Inside Global addresses, 114
Inside Local addresses, 114
intercept mode, TCP intercept, 776
interface ID (IPv6 global addresses), 121
interface resets, synchronous serial links, 594-595
interfaces, 521
assigning to VLANs, 35
using configuration mode to put interfaces into VLANs, 34-35
versus subinterfaces and virtual circuits, queuing, 520
Interior Gateway Protocol (IGPs), 360
internal BGP. *See* iBGP
internal processing, switches, 23
Internal Spanning Tree (IST), 82
International Organization for Standardization (ISO), 851
Internet Assigned Numbers Authority (IANA), 638
Internet Group Management Protocol. *See* IGMP
Internet Standard Management Framework. *See* SNMP
Internetwork Control Message Protocol. *See* ICMP
internetworks, 96
interoperability, IGMPv1 and IGMPv2, 657-658
intersymbol interference (ISI), 820
intrusion detection system. *See* IDS
Inverse ARP (Frame Relay Inverse ARP), 179-183

IOS
boot system command, 859
command help, 860
commands, 862
queuing, 521
IOS commands, catalyst IOS commands for catalyst switch configuration, 23
IOS software boot sequences, 858-859
IP
ARP, 137-139
BOOTP, 139-141
command reference, 146-148
DHCP, 139-141
GLBP, 141-143
HSRP, 141-143
ICMP, 134-135
ICMP Redirect message, 137
ICMP Time Exceeded message, 136
ICMP Unreachable message, 135
NTP, 143-145
proxy ARP, 137-139
RARP, 139-141
standards documents for, 146
and VLANs, 31
VRRP, 141-143
ip access-group command, 769
IP ACE, port matching, 771
IP ACL (IP access control lists), 769
ACEs, 770-771
command references, 769
wildcard masks, 772
IP addresses, 94-95
CIDR (classless interdomain routing), 111-112
classful logic, 94-95
subnets, 95-96
classless logic, 94, 97
command reference, 126
determining range of
binary method, 98-99
decimal method, 99-101
header format, 126-128
inappropriate IP addresses, 774-775
IPv6, 119-120
address formats, 120
aggregatable global unicast addresses, 120-121
dual stack configuration, 121-123
types of addresses, 123-124
NAT (network address translation), 113-114, 125
dynamic NAT (without PAT), 116-117
dynamic NAT configuration, 118-119
static NAT, 115-116
PAT (Port Address Translation), 117
private addressing, 113
protocol field values, 128
protocols for, 125
route summarization, 107-108
exclusive summary routes (binary method), 110
inclusive summary routes (binary method), 108-109
inclusive summary routes (decimal method), 109-110
standards documents, 125
subnet numbers, determining all
binary method, 102-104
decimal method, 104-105
subnets
allocation, 105-106
size of, 97-98
ip bgp-community new-format command, 474
IP cef global configuration command, 179
ip classless command, 186, 345
IP community lists, matching, 475
ip community-list command, 474, 479

IP forwarding, 177-178
classful routing, 185-186
classless routing, 185-186
command references, 197
fast switching, 178
switching paths, 178-179
IP hosts, 94
IP multicast routing, command references, 732
ip multicast routing, 692
IP multicasting, 629, 632
ip multicast-routing, 708
ip ospf area commands, 295
ip ospf authentication, 301
ip ospf cost, 50, 293-294
ip ospf network, 269
ip pim dense-mode, 692
ip pim rp-address, 720
ip pim sparse-mode, 708
ip pim spt-threshold, 717
ip policy command, 192
IP Precedence (IPP) field, 489-490
IP prefix lists, 321-322
ip proxy-arp, 196
IP routing tables, 392
adding eBGP routes, 392-393
adding iBGP routes, 394-396
BGP synchronization and redistributing routes, 396-398
confederations, 399-401
configuring confederations, 401-404
disabling BGP synchronization, 398
RRs, 404-409
adding multiple BGP routes, 450
backdoor routes, 393-394
clearing EIGRP, 250
IP routing. *See* IP forwarding
IP Source Guard, 763-764
ip summary-address rip, 345
ip verify source command, 764
IPP (IP Precedence) field, 489-490
IPsec, 163
IPv4 addresses. *See* IP addresses
IPv6, 119-120
address formats, 120
aggregatable global unicast addresses, 120-121
dual stack configuration, 121-123
types of addresses, 123-124
ISI (intersymbol interference), 820
ISL, VLAN trunking, 44-45
configuration, 45-46
ISO (International Organization for Standardization), 851
isolated VLANs, 37
IST (Internal Spanning Tree), 82

J–K

J1, 592
join message process, CGMP, 666
Join messages, 699
joining shared trees, PIM-SM, 710-712
joining groups, 635
IGMP, 644
IGMP snooping, 670

K values (EIGRP), 234
keepalive timer, 368
key chains, RIP authentication, 216
key sequences for command edit and recall, 861
keyboard commands, 861
keywords
allow-default, 774
established, 771
ACL, 776
group radius, 750
group tacacs+, 750

in, 428
not-advertise, 344
out, 428
passive, 603
summary-only, 429

L

L3PDU (Layer 3 PDU), 856
LACP (Link Aggregation Control Protocol), 77-78
LAN switch forwarding behavior, 19
LANs (local area networks)
802.11. *See* IEE 802.11
DRs, 266
optimizing, 266-268
wireless LANs. *See* wireless LANs
LAPF (Link Access Procedure for Frame-Mode Bearer Services, 612
launching applications, 635
Layer 2 payload compression, 602
Layer 2 security
private VLANs. *See* private VLANs
switch ports, 752
Layer 3 interfaces, MLS, 188
Layer 3 PDU (L3PDU), 856
Layer 3 protocol, 852
Layer 3 security, 768-769
directed broadcasts, 772-774
established keyword, 776
inappropriate IP addresses, 774-775
IP ACL, 769
ACEs, 770-771
wildcard masks, 772
RFCs, 768
RPF checks, 772-774
smurf attacks, 772-774
TCP intercept, 776-777
TCP SYN flood, 775
Layer 3 switching, 186
layers
OSI models, 851-852, 854
benefits, 854-855
layer interactions, 856-857
TCP/IP, 855
LCP (Link Control Protocol), 596-597
configuration, 597-598
LFI, 600-601
MLP, 598-600
Learning state (Spanning Tree), 68
leave groups, IGMPv2, 654-656
leave mechanism, IGMPv1, 651
leave messages, CGMP, 667
Lempel-Ziv (LZ) compression, 602
Lempel-Ziv Stacker (LZS), 602
LFI (Link Fragmentation and Interleaving), 600
LCP, 600-601
lightweight access points, 833
limited-function IOS, 858
limiting
bandwidth, LLQ, 537
CBWFQ bandwidth, 532-534
line coding, 592
Link Access Procedure for Frame-Mode Bearer Services (LAPF), 612
Link Aggregation Control Protocol (LACP), 77-78
Link Control Protocol. *See* LCP
Link Fragmentation and Interleaving (LFI), 600-601
Link Quality Monitoring (LQM), LCP, 597
link state, 260
link-state advertisements (LSAs), 260
link-state ID (LSID), 278
Link-State Refresh (LSRefresh), 275
Link-State Request (LSR), 265

Linksys WRT55AG dual-band access point, 801
Listening state (Spanning Tree), 68
lists most of the settings that impact show ip bgp command, 454
little-endian, 17
LLC (Logical Link Control), 14
LLQ (low-latency queuing), 521, 529, 534-536, 538
bandwidth, 537
FRTS configuration, 569-570
priority queues, 538
tuning shaping for voice, 561-564
LMI (Local Management Interface), 611-612
load balancing
EIGRP, 249
PortChannels, 76
local computation, EIGRP, 241-243
Local Management Interface (LMI), 611-612
LOCAL_PREF, 457-458
LOCAL_PREF PA, 447
LOF (Loss of Frame), 594
log-adjacency-changes detail, 292
logic
discarding, 540
MLS logic, 186
Logical Link Control (LLC), 14
login authentication command, 751
login security, overriding defaults for, 751
Loop Guard, 83-84
loop prevention, RIP, 205-206
ceased updates, 210-212
steady-state operation, 206-208
triggered updates and poisoned routes, 208-210
tuning, 212-213
loopback circuitry, NICs, 11
looped link detection, LCP, 597
loop-inconsistent state, 84
LOS (Loss of Signal), 594
Loss of Frame (LOF), 594
low-latency queuing. *See* LLQ
LQM (Link Quality Monitoring), LCP, 597
LSA headers, 265
flooding to neighbors, 264
LSA summary, 287
LSA type 1, 278-281
LSA type 2, 278-281
LSA type 3, 281-284
LSA type 4, 284-285
LSA type 5, 284-285
LSA types and OSPF design, 286-287
lsa-group, 309
LSAs (link-state advertisements), 260
LSID (link-state ID), 278
LSR (Link-State Request), 265
LSRefresh, 308
LSRefresh (Link-State Refresh), 275
LZ (Lempel-Ziv) compression, 602
LZS (Lempel-Ziv Stacker), 602

M

MAC address reduction, 63
MAC addresses
mapping to multicast IP addresses, 642-643
overriding, 18
tables, displaying, 53, 86, 197, 778
Management Information Base (MIB), 163
mandatory PAs, 446
many-to-few multicasts, 632
many-to-many multicasts, 632
map-class frame-relay command, 566
map-class shape-with-LLQ, 620

mapping multicast IP addresses to MAC addresses, 642-643
mark probability denominator (MPD), 540
marking tools. *See* classification and marking tools
masks
classful IP addressing, 94
wildcard masks, 424
match as-path list-number command, 439
match command, 319-320
match commands (MQC), 498-499, 511-512
match ip address command, 192
match length command, 192
MaxAge, 308
maximum segment size (MSS), 158
maximum-paths command, 450, 470, 472
BGP decision process tiebreakers, 466-467
max-metric router-lsa on-startup announce-time command, 304
max-metric router-lsa on-startup wait-for-bgp command, 304
max-reserved-bandwidth, 532
MBGP (Multiprotocol Border Gateway Protocol), 687
Measured Round-Trip Time (MRTT), 157
MED (MULTI_EXIT_DISC), 464
configuring
multiple adjacent autonomous systems, 465-466
single adjacent AS, 465
features of, 464
scope of, 466
memorizing, mnemonics for memorizing BGP decision process, 451-452
messages
Assert messages, PIM, 703-704
CGMP, 668
Graft messages, PIM-DM, 700-702
Join, 699
OSPF messages, 261-262
OSPF messages. *See* OSPF messages
PIM-DM, summary of messages, 705
Prune messages, PIM-DM, 693-695
state refresh messages, PIM-DM, 699-700
metacharacters, 433

metric types
redistribution routes, 340-342
route redistribution, 328-329
metrics
calculating for types 1 and 2, 285-286
redistribution routes, 340-342
route redistribution, 328-329
MIB (Management Information Base), 163
SNMP, 164-166
Microsoft Point-to-Point Compression (MPPC), 602
MIMO (multiple input-multiple output), 791
minimum shaping rate, 559
MIR (minimum information rate), 559
MISTP (Multiple Instance STP), 80
MLD (Multicast Listener Discovery), 662
MLP (Multilink PPP), 597
LCP, 598-600
LFI, 600-601
MLS (multilayer switching), 186
configuring, 188-191
Layer 3 interfaces, 188
logic, 186
routed ports, 187

mnemonics
for memorizing BGP decision process, 451-452
OSI terminology, 855
modified tail drop, 526
modifying queue length, 520
Modular QoS CLI. *See* MQC, 495
modulation, RF signals, 814
MOSPF (Multicast Open Shortest Path First), 635, 691, 706
MPD (mark probability denominator), 540
MPLS Experimental (EXP) field, 494
MPPC (Microsoft Point-to-Point Compression), 602
MQC (Modular QoS CLI), 495-496
class maps, 497-499
match commands, 511-512
NBAR, 499-500
mroute flags, 735
MRTT (Measured Round-Trip Time), 157
MSDP (Multicast Source Discovery Protocol), 721
Anycast RP, 726-727
MSS (maximum segment size), 158
MST (Multiple Spanning Trees), 80, 82
configuring, 81
MSTP (Multiple STP), 80
mtrie, 179
multicasting, traffic, 635
MULTI_EXIT_DISC (MED), 464
multi-action policing, CB Policing configuration, 578
multicast addresses, 16
multicast applications, 632
multicast Ethernet frames, 16
multicast forwarding using dense mode, 684-685
multicast forwarding using sparse mode, 687-689
multicast IP addresses, 638
GLOP addressing, 640
mapping to MAC addresses, 642-643
permanent multicast groups, 639
private multicast domains, 640
ranges, 641
SSM, 640
transient groups, 639, 641
multicast IP addresses range and structure, 638
Multicast Listener Discovery (MLD), 662
Multicast Open Shortest Path First. *See* MOSPF
multicast routing, 683-684
dense-mode routing protocols. *See* dense-mode routing protocols
multicast forwarding using dense mode, 684-685
multicast forwarding using sparse mode, 687-689
multicast scoping. *See* multicast scoping
problems, 683
RPF check, 685-687
multicast scoping, 689
administrative scoping, 690
TTL scoping, 689-690
Multicast Source Discovery Protocol. *See* MSDP
multicasting, 632
broadcast method, 633
requirements for, 635
scaling, 637
traffic, 637
unicast, 632-633
multicasts
CGMP. *See* CGMP
IGMP snooping. *See* IGMP snooping
RGMP. *See* RGMP

multi-exit discriminators, 464
multilayer switching. *See* MLS
Multilink PPP (MLP), 597
multipath, RF signals, 820-821
multiple adjacent AS, 466
multiple input-multiple output (MIMO), 791
Multiple Instance STP (MISTP), 80
Multiple Spanning Trees (MST), 80, 82
configuring, 81
Multiple STP (MSTP), 80
Multiprotocol Border Gateway Protocol (MBGP), 687
mutual redistribution at multiple routers, 333-335

N

naming VLANs, 53, 197
NAT (Network Address Translation), 111, 113-114, 125
dynamic NAT, configuration, 118-119
dynamic NAT (without PAT), 116-117
static NAT, 115-116
NAT overloading. *See* PAT (Port Address Translation)
NBAR (Network-Based Application Recognition), 507
CB Marking tool, 507-508
MQC classification with, 499-500
NBMA (nonbroadcast multi-access) networks, 269
NBMA networks
OSPF network types, 270-272, 274
setting priority on, 272-273
NCP (Network Control Protocol), 596
neighbor command, 270, 274, 468
neighbor default-originate, 382
neighbor ebgp-multihop command, 401, 468
neighbor filter-list command, 439
neighbor ID, 467, 469
maximum-paths command, 471-472
neighbor peer-group command, 365
neighbor remote-as command, 365-366
neighbor route-map command, 439
neighbor shutdown command, 370

neighbor states, 262
OSPF, 309
neighbor weight command, 456
neighbors
advertising BGP routes to, 383
BGP Update message, 383-384
determining contents of updates, 384-386
impact of decision process and NEXT_HOP, 386-391
BGP neighbors. *See* BGP neighbors
discovering, 263-264
EIGRP, 233-236
Network Address Translation. *See* NAT
network backdoor command, 394
network command, 295
injecting prefixes and routes into BGP tables, 370-371, 373
Network Control Protocol (NCP), 596
network layer protocol, 852
Network Layer Protocol ID (NLPID), 612
network layer reachability information (NLRI), 370
network part (classful IP addressing), 94
Network-Based Application Recognition. *See* NBAR
networks, 94, 96
connecting to networks with infrastructure mode, 799
NBMA networks
OSPF network types, 270-272, 274
setting priority on, 272-273

NEXT_HOP, 466
NEXT_HOP PA, 385, 446
NEXT_HOP reachable, 456
next-hop feature, RIP, 219-220
NICs, loopback circuitry, 11
NLPID (Network Layer Protocol ID), 612
NLRI (network layer reachability information), 370
filtering using COMMUNITY values, 479
filtering BGP updates, 424-427
route maps, 427
soft reconfiguration, 428
NLRI filtering tools, 424
no auto-summary command, 370
no frame-relay inverse-arp, 184
no frame-relay inverse-art, 184
no ip classless, 186
no ip classless command, 345
no ip directed-broadcast command, 773
no ip route-cache cef commands, 179
no synchronization command, 395
no terminal editing, 861
nonbroadcast multi-access (NBMA) networks, 269
noncanonical, 17
nonoverlapping channels, IEEE 802.11g, 790
nontransitive, PAs, 446
normal-range VLANs, 42
not-advertise keyword, 344
NTP (Network Time Protocol), 143-145
numeric ranges, OSPF, 310

O

OAM (Operation Administration, and Maintenance), 594
Offset (HLEN), 160
offset lists
EIGRP, 250
RIP, 220-222
OOF (Out of Frame), 594
open system authentication, 811-812
Open System Interconnection reference model. *See* OSI models
Operation, Administration, and Maintenance (OAM), 594
optimizing
DRs on LANs, 266-268
STP, 73
BackboneFast, 73, 75
discovery and configuration of PortChannels, 77-78
load balancing PortChannels, 76
PortChannels, 76
PortFast, 73-75
UplinkFast, 73-75
Organizationally Unique Identifier (OUI), 17
ORIGIN, BGP tables, 382-383
ORIGIN PA, 458, 463
orthogonal frequency division multiplexing, RF signals, 818

OS categories four routers, 858
OSI (Open System Interconnection) models, 851
OSI models, 851
layer interactions, 856-857
layers, 851-852, 854
benefits, 854-855
terminology, 855-856
OSPF (Open Shortest Path First)
command references, 305-307
configuring, 290-292
alternatives to OSPF network command, 295
authentication, 301-303
costs, 292-295

stub router, 303-304
virtual links, 299-301
costs, 292-295
DRs on LANs, 266
election on, 268-269
optimizing, 266-268
filtering, 295
ABR LSA type 3 filtering, 297-299
distribute-list command, 295-297
neighbor states, 309
numeric ranges, 310
processes, clearing, 292-295
RIDs, 260-261
SPF calculation, 274-275
steady-state operation, 275
stubby areas, 287-290
ABRs, 287
OSPF ABR LSA type 3 filtering, 297-298
area range command, 299
ospf auto-cost reference-bandwidth, 294
OSPF database exchange, 260
IP protocol 89, 261
RIDs, 260-261
OSPF design, 276-277
and LSA types, 286-287
OSPF LSA types, 277-278
and OSPF design, 286-287
types 1 and 2, 278-281
types 3, 281-284
types 4 and 5, 284-285
OSPF messages, 261-262
DD messages, flooding LSA headers to neighbors, 264
hello messages, discovering neighbors, 263-264
LSA headers, 265
OSPF network types, 269
NBMA networks, 270-272, 274
OSPF route summarization, 344
OSPF timer, 308
OSPF wait time, 268
OUI (Organizationally Unique Identifier), 17
out keyword, 428
Out of Frame (OOF), 594
outgoing interface lists, 694
Outside Global addresses, 114
Outside Local addresses, 114
overriding
defaults for login security, 751
MAC addresses, 18

P

Packet Description Language Modules (PDLMs), 508
packet routing
ARP, 137-139
BOOTP, 139-141
classification and marking tools. *See* classification and marking tools
command references, 146-148
DHCP, 139-141
EIGRP, 233
adjacencies, 233-236
authentication, 250
autosummarization, 250
clearing IP routing tables, 250
command reference, 251-252
configuration, 246-249
convergence, 240-246
load balancing, 249
offset lists, 250
packet types, 252
route filtering, 250
split horizon, 250
topology table, 238-240
updates, 236-238
GLBP, 141-143
HSRP, 141-143

ICMP, 134-135
ICMP Redirect message, 137
ICMP Time Exceeded message, 136
ICMP Unreachable message, 135
NTP, 143-145
proxy ARP, 137-139
RARP, 139-141
RIP, 204-205
authentication, 216-219
command reference, 225-226
configuration, 213-216
convergence and loop prevention, 205-213
distribution list and prefix list filtering, 222-224
next-hop and split horizon features, 219-220
offset lists, 220-222
standards documents, 225
standards documents for, 146
VRRP, 141-143
packets
conforming packets, 571
Ethernet, 14
exceeding packets, 571
queuing, 521
violating packets, 571
PAgP (Port Aggregation Protocol), 77-78
parameters, FRTS configuration, 568-569
PAs (path attributes), 360
BGP, 410
NEXT_HOP, 385
ORIGIN, BGP, 382-383
ORIGIN PA, 458-459, 463
passive, 603
passive mode, FTP, 161-162
passive scanning, infrastructure mode, 798
password command, 745
passwords
CLI, 745-746
enable and username passwords, 746-747
hashed/encrypted enable passwords, 748
PAT (Port Address Translation), 117
configuration, 118-119
path attributes. *See* PAs
path vector logic, 360
payload compression, Frame Relay, 619-620
PCF (point coordination function), 809
PCM (pulse code modulation), 592
PDLMs (Packet Description Language Modules), 508
PDU (protocol data unit), 856
peak information rate (PIR), 573
peak rates, CB Shaping, 565
Per VLAN Spanning Tree Plus (PVST+), 68-70
percentages, CB Policing configuration, 578-579
Per-Hop Behaviors. *See* PHBs
permanent multicast groups, multicast IP addresses, 639
permit .*, 442
phase shift keying (PSK), 815
PHBs (Per-Hop Behaviors), 490
Assured Forwarding (AF) PHBs, 491-492
Class Selector (CS) PHBs, 491
Expedited Forwarding (EF) PHBs, 492-493
PIM (Protocol Independent Multicast), 687
bidirectional PIM, 729-730
sparse-dense mode, 723

PIM-DM (Protocol Independent Multicast dense mode), 635-636, 638, 690-691
Assert messages, 703-704
designated routers, 704
forming adjacencies with PIM hello messages, 691
Graft messages, 700-702
Prune messages, 693-695
Prune Override, 702-703
reacting to failed links, 695-697
rules for pruning, 697-699
source-based distribution trees, 692-693
steady-state operation and state refresh messages, 699-700
summary of messages, 705
versus PIM-SM, 707, 730
PIM-SM (Protocol Independent Multicast Sparse Mode), 635, 689, 707
Assert messages, 703-704
designated routers, 704
finding RPs, 720
Anycast RP with MSDP, 726-727
Auto-RP, 721-723
BSR, 724-725
joining shared trees, 710-712
Prune Override, 702-703
pruning shared trees, 719-720
RPs multicast routing tables, 716-717
shared distribution trees, 714
shortest-path tree switchovers, 717-719
source registration process, 712, 714
sources sending packets to RP, 708-710
steady-state operations by continuing to send joins, 715-716
versus PIM-DM, 707, 730
ping packets, 134
PIR (peak information rate), 573
point coordination function (PCF), 809
Point-to-Point Protocol. *See* PPP
poisoned routes (RIP), 208-210
police command, 575, 577-578
policers, 509-510
policies, BGP routing policies. *See* BGP routing policies
policing
CB Policing. *See* CB Policing
single-rate, three-color policing, 573
single-rate, two-color policing, 571-572
subsets of traffic, CB Policing, 576-577
two-rate, three-color policing, 573-575
policy maps, 564
policy routing, 191, 193-196, 510
set commands, 192
policy-map command (MQC), 496
policy-map queue-voip, 563
poll interval, 308
Port Address Translation. *See* PAT
Port Aggregation Protocol (PAgP), 77-78
port matching, IP ACE, 771
port numbers, TCP, 155-156
port security configuration commands, 755
PortChannels, optimizing STP, 76
discovery and configuration, 77-78
load balancing, 76
PortFast, optimizing STP, 73-75
ports, 61
access ports, protecting, 82-83
designated ports, 64-65
root ports, 63-64
routed ports, MLS, 187
switch ports, 752
switches, assigning to VLANs, 53
trusted ports, 752
unused ports, 752
best practices for, 753-754
unused ports. *See* unused ports
user ports. *See* user ports

POST (power-on self-test), 858
power-save mode, wireless configuration parameters (IEEE 802.11), 805-806
PPP (Point-to-Point Protocol), 595-596
compression, 601-602
header compression, 602-603
layer 2 payload compression, 602
LCP, 596-597
configuration, 597-598
LFI, 600-601
MLP, 598-600
security, 752
ppp authentication, 752
ppp multilink fragment-delay commands, 601
ppp multilink interleave command, 600
PQ (priority queuing), 522
prefix length, 97
prefix list filtering, RIP, 222-224
prefix lists versus route maps and distribute lists (BGP), 428-429
prefix part (IP addressing), 97
prefixes, 94, 97
injecting into BGP tables, 370
network command, 370-371, 373
redistributing from IGP, static, or connected routes, 373-375
prefix-list commands, 322
BGP, 425
prepending AS_PATH, 461, 463
preventing
suboptimal routes by setting the AD, 335-338
suboptimal routes by using route tags, 338-340
primary subnets, 234
primary VLANs, 37
priority command, 529, 535
priority queuing (PQ), 522
priority-queue out command, 543
privacy, TCP/IP, 163
private IP addressing, 113
private multicast domains, multicast IP addresses, 640
private VLANs, 36-38, 767-768
process switching, IP forwarding, 179
protecting
access ports, 82-83
STP, 82
BPDU Guard, 82-83
Loop Guard, 83-84
Root Guard, 82-83
UDLD, 83-84
trunks, 83-84
protocol data unit (PDU), 856
protocol field values, IP addressing, 128
Protocol Independent Multicast (PIM), 687
Protocol Independent Multicast Dense Mode.
***See* PIM-DM**
Protocol Independent Multicast-Sparse Mode. *See* PIM-SM
Protocol Type fields. *See* Type fields
protocols
application protocols, TCP/IP, 160-161
ARP, 137-139
BOOTP, 139-141
DHCP, 139-141
EIGRP, 233
adjacencies, 233-236
authentication, 250
autosummarization, 250
clearing IP routing tables, 250
command reference, 251-252
configuration, 246-249
convergence, 240-246
load balancing, 249
offset lists, 250
packet types, 252
route filtering, 250
split horizon, 250

topology table, 238-240
updates, 236-238
GLBP, 141-143
HSRP, 141-143
ICMP, 134-135
ICMP Redirect message, 137
ICMP Time Exceeded message, 136
ICMP Unreachable message, 135
IGMP, 635
IGPs, 360
for IP addresses, 125
MLD, 662
NTP, 143-145
packet routing, standards documents for, 146
PPP. *See* PPP
proxy ARP, 137-139
RARP, 139-141
RIP, 204-205
authentication, 216-219
command reference, 225-226
configuration, 213-216
convergence and loop prevention, 205-213
distribution list and prefix list filtering, 222-224
next-hop and split horizon features, 219-220
offset lists, 220-222
standards documents, 225
RTP, 236
SNMP, 164-166
TCP. *See* TCP
UDP, 154-155
VRRP, 141-143
proxy ARP, 137-139
Prune messages, PIM-DM, 693-695
Prune Override, PIM, 702-703
pruning
PIM-DM, 697-699
shared trees, PIM-SM, 719-720
pseudonodes, 279
PSH (Push), 160
PSK (phase shift keying), 815
public wireless LANs, 840-842
pulse code modulation (PCM), 592
PVST+ (Per VLAN Spanning Tree Plus), 68-70

Q

QAM (quandrature amplitude modulation), 815, 818
QoS service classes, 496
QoS tools, classification and marking. *See* classification and marking tools
quartets, IPv6 addresses, 120
querier
IGMPv1, 651
IGMPv2, 656
Query Response Interval, 649
query scope (EIGRP), limiting, 246
queue lengths
modifying, 520
WFQ, 526-527
queuing, 515, 521
Cisco 3550, 545-546
Cisco 3550 switches, egress queuing, 543-545
discard categories, WRED, 539
hardware queues, 519
interfaces versus subinterfaces and virtual circuits, 520
protocol comparison, 538
software queues, 519
WRED
configuration, 542
weight packets, 541

queuing tools
CBWFQ, 529-530, 532, 538
bandwidth, 532-534
command references, 530
comparing, 520
CQ, 523-524
FIFO queuing, 521
LLQ, 529, 534-536, 538
bandwidth, 537
more than one priority queue, 538
PQ, 522
WFQ, 524-525
configuration, 527-528
drop policy, number of queues, and queue lengths, 526-527
types of queues, 527
WFQ scheduler, 525-526
queue-voip, 569

R

R2, 407
R2# sh ip bgp | include 10.1.34.4, 444
R2# show ip bgp neighbor 3.3.3.3 received-routes | include 303, 443
R3, 407
R9, 407
radio cards, 795
radio frequency peripherals, 797
radio management aggregation, 828
RADIUS, 747
configuring server groups, 750
RADIUS attribute, 764
radius-server host, 750
ranges, multicast addresses, 641
Rapid Spanning Tree Protocol (RSTP), 78-80
RARP, 139-141
RAT (Router Audit Tool), 775
rate-limit ACL, 580
rate-limit command, 579
RD (reported distance), 239
reacting to failed links, PIM-DM, 695-697
reassociation, 793
receiver's advertised window, 157
receiver's window, 157
Red alarm, 594
redistribute command, 321, 324-325
redistribute connected command, 458
redistribute ospf commands, 328
redistribute static, 347-348
redistributing
from IGP, static, or connected routes, BGP tables, 373-375
route maps
with match command, 319-320
with set commands, 320
redistribution, 324, 329-333
command references, 352
metrics and metric types, 340-342
mutual redistribution at multiple routers, 333-335
setting metrics, metric types, and tags, 328-329
using default settings, 325-328
refilling dual token buckets, 573
regular expressions, matching AS_PATH, 433-434
relay agents (DHCP), 140
Reliable Transport Protocol (RTP), 236
Remote Monitoring MIB, 166
removing
COMMUNITY values, 475, 479
private ASNs, 460-461
rendezvous point (RP), 687
repeaters, 796
reported distance (RD), 239
requesting LSA headers, 265

request-to-send/clear-to-send. ***See*** **RTS/CTS**
Resource Reservation Protocol (RSVP), 527
Retransmission, 308
reverse-path-forwarding (RPF) paths, 684
revision numbers, VTP, 39-40
RF channels, wireless configuration parameters (IEEE 802.11), 803-804
RF interference, RF signals, 819-820
RF signals, 814
characteristics, 815-816
FCC rules, 819
gain, 816
modulation, 814
multipath, 820-821
orthogonal frequency division multiplexing, 818
RF interference, 819-820
SNR, 816-817
spread spectrum, 817-818
RFCs
DiffServ, 512
Layer 3 security, 768
RGMP (Router-Port Group Management Protocol), 663, 673-675
IGMP snooping, 674
RIB (BGP Routing Information Base), 370
RID (router identifier), 260
RIP (Routing Information Protocol), 204-205
command references, 225-226
configuration, 213-214
authentication, 216-219
autosummarization, 214-216
distribution list and prefix list filtering, 222-224
next-hop and split horizon features, 219-220
offset lists, 220-222
convergence and loop prevention, 205-206
ceased updates, 210-212
steady-state operation, 206-208
triggered updates and poisoned routes, 208-210
tuning, 212-213
standards documents, 225
RIP route summarization, 345
RJ-45 pinouts, 8-9
roaming, infrastructure mode, 800
ROMMON, 858
Root Guard, 82-83
enabling, 754
root port (RP), 63-64
root switches, electing, 61-63
root-path tree (RPT), 710
route aggregation, AS_PATH, 461, 463
route cache, 178
route default routes. ***See*** **default routes**
route filtering, EIGRP, 250
route maps
configuring with route-map command, 317-319
deny clauses, 333
match and set commands for BGP, 479
match commands for route redistribution, 319-320
NLRI filtering, 427
policy routing, 510
versus prefix lists and distribute lists (BGP), 428-429
redistributing subsets of routes, 329-333
set commands for route redistribution, 320

route redistribution, 324
influencing with metrics and metric types, 340-342
redistribute command, 324-325
setting metrics, metric types, and tags, 328-329
using default settings, 325-328
using route maps, 329-333
route reflectors. *See* RRs
route summarization, 107-108, 342-343
creating default routes, 350-351
EIGRP route summarization, 344
exclusive summary routes, binary method, 110
inclusive summary routes
binary method, 108-109
decimal method, 109-110
OSPF route summarization, 344
RIP route summarization, 345
route tags, preventing suboptimal routes, 338-340
routed ports, MLS, 187
route-map command, 317-319
BGP, 425
Router Audit Tool (RAT), 775
router bgp command, 365, 401
router identifier (RID), 260
Router-Port Group Management Protocol. *See* RGMP
routers
ABRs, 276
BGP router ID of advertising router, 467
configuring VLAN trunking on, 49-51
designated routers, PIM, 704
downstream routers, 697
IGMPv1 and IGMPv2, 658
mutual redistribution at multiple routers, 333-335
OS categories, 858
OSPF router IDs, 260-261
queuing, 521
upstream routers, 697
Wi-Fi, 843
wireless LAN routers, 797
routes
backdoor routes, IP routing tables, 393-394
BGP. *See* BGP
default routes, 345-346
adding to BGP, 381-382
injecting into BGP tables, 370
impact of auto-summary on redistributed routes and, 375-377
manual summaries and AS_PATHs, 378-381
network command, 370-371, 373
redistributing from IGP, static, or connected routes, 373-375
ORIGIN, BGP tables, 382-383
preventing suboptimal routes by setting AD, 335-338
preventing suboptimal routes by using route tags, 338-340
static routes, redistribute static, 347-348
routing
classful routing, 185-186
classless routing, 185-186
policy routing, 191, 193-196
set commands, 192
Routing Information Protocol. *See* RIP
routing packets. *See* packet routing

RP (rendezvous point), 687
finding, 720, 728
Anycast RP with MSDP, 726-727
with Auto-RP, 721-723
with BSR, 724-725
multicast routing tables, PIM-SM, 716-717
sources sending packets to, PIM-SM, 708-710
RP (root port), 63
RPF (reverse-path-forwarding) paths, 684
RPF checks, 772-774
multicast routing, 685-687
RPT (root-path tree), 710
RRs (route reflectors), 404
IP routing tables, 404-409
RSTP (Rapid Spanning Tree Protocol), 78-80
RSVP (Resource Reservation Protocol), 527
RTP (Reliable Transport Protocol), 236
RTS/CTS (request-to-send/clear-to-send), 806
wireless configuration parameters, IEEE 802.11, 806-808

S

same-layer interaction, 857
SAP (Service Advertising Protocol), 644
scaling, multicasting, 637
scanning, infrastructure mode, 798
active scanning, 799
passive scanning, 798
scheduling
CQ logic, 523
PQ logic, 522
strict-priority scheduling, 547
schemes, queuing, 521
scoping, multicast scoping. ***See*** **multicast scoping**
SDP (Session Description Protocol), 644
secondary VLANs, 37
Secure Shell (SSH), 163, 768
Secure Sockets Layer (SSL), 163
secure user interfaces, CiscoWorks WLSE, 836
security
AAA, 747
authentication methods, 748-750
groups of AAA servers, 750-751
overriding defaults for login security, 751
enterprise security, 837-838
Layer 2 security, switch ports. *See* switch ports
Layer 3 security, 768-769
Layer 3 security. *See* Layer 3 security
passwords. *See* passwords
port security, 754-758
PPP, 752
sniffer traces, 40
SNMP, 164, 167
switches. *See* switch ports
TCP/IP, 163
wireless security. *See* wireless security
security policy monitoring, CiscoWorks WLSE, 836
self-healing functions, CiscoWorks WLSE, 837
sequence number (SN), 525
serial cable control pin leads, 595
servers, groups of AAA servers, 750-751
Service Advertising Protocol (SAP), 644
service password-encryption, 303, 746-747
service set identifier (SSID), 799
service-policy command, 532
service-policy command (MQC), 496

service-policy out command, 538
service-policy output, 532, 559
service-policy output policy-map-name, 564
Session Description Protocol (SDP), 644
set as-path prepend command, 461
set commands, 320
policy routing, 192
set community none command, 475
set fr-de command, 615
SF (Superframe), 592
shape average, 565
shape command, 559, 561
shape fecn-adapt command, 614
shape peak mean-rate command, 565
shape percent command, 564
shaped rate, 556
shaping. *See also* traffic shaping
adaptive shaping, 565
adaptive shaping, FRTS, 570
CB Shaping, 551
configuring by bandwidth percent, 564
FRTS, 551
tuning shaping for voice using LLQ and Tc, 561-564
shaping queues, 555
shaping rate, 556-557
minimum shaping rate, 559
shared distribution trees, PIM-SM, 714
shared key authentication, 811-813
shared trees
creating, 711
joining with PIM-SM, 710-712
pruning, PIM-SM, 719-720
Shortest Path First (SPF), 265
shortest-path tree (SPT), 692
shortest-path tree switchovers, PIM-SM, 717-719
show command, WFQ, 528
show interface trunk command, 48
show interfaces command, 595
show ip arp command, 195
show ip bgp command, 382, 439, 453, 455
show ip bgp neighbor advertised-routes command, 388
show ip bgp neighbor neighbor-id advertised-routes command, 439
show ip bgp neighbor neighbor-id received routes, 439
show ip bgp regexp expression command, 439
show ip command, 23
show ip mroute, 714
show ip mroute command, 692
show ip ospf border-routers, 283
show ip ospf database command, 281
show ip ospf database summary link-id command, 283
show ip ospf neighbor command, 262
show ip ospf statistics command, 283
show ip route command, 290
show queue command, 528
signal-to-noise ratio (SNR), 816-817
Simple Network Management Protocol. *See* SNMP
single adjacent AS, 465
single-bucket, two-color policing, 572
single-rate, three color policing, CB Policing configuration, 575-576
single-rate, three-color policing, 573
single-rate, two-color policing, 571-572
sliding windows, TCP, 157-159
Slow Start Threshold (SSThresh), 159
SLSM (Static Length Subnet Masking), 104
small office wireless LANs, 842-844
smurf attacks, 772-774
SN (sequence number), 525
SNAP (Sub-Network Access Protocol), 14
sniffer traces, 40

SNMP (Simple Network Management Protocol), 163
Get message, 166
Inform message, 166
MIBs, 164, 166
protocol messages, 165-166
protocols, 164
Response message, 166
security, 167
security and administration, 164
Set command, 166
versions, 164
SNMP Traps, 166
SNR (signal-to-noise ratio), 816
RF signals, 816-817
soft reconfiguration, NLRI filtering, 428
software queues, 519
solicited host membership report, IGMPv1, 648-649
source registration process, PIM-SM, 712, 714
source-based distribution trees, PIM-DM, 692-693
Source-Specific Multicast (SSM), 638
Spanning Tree Protocol. *See* STP
spanning-tree portfast command, 79
spanning-tree vlan command, 73
sparse mode multicast forwarding, 687-689
sparse-dense mode, PIM, 723
sparse-mode routing protocols, PIM-SM, 707
joining shared trees, 710-712
pruning shared trees, 719-720
RP's multicast routing tables, 716-717
shared distribution trees, 714
shortest-path tree switchovers, 717-719
source registration process, 712, 714
sources sending packets to RP, 708-710
steady-state operations by continuing to send, 715-716
versus PIM-DM, 707
speed, Ethernet, 9
SPF (Shortest Path First), 265
SPF calculation, OSPF, 274-275
split horizon
EIGRP, 250
RIP, 219-220
spread spectrum, RF signals, 817-818
SPT (shortest-path tree), 692
SSH (Secure Shell), 163, 768
SSID (service set identifier), 799
CiscoWorks WLSE, 835
wireless configuration parameters, IEEE 802.11, 802-803
SSL (Secure Sockets Layer), 163
SSM (Source-Specific Multicast), 638
IGMPv3, 660
multicast IP addresses, 640
SSThresh (Slow Start Threshold), 159
standards documents
for IP addressing, 125
for packet routing protocols, 146
RIP, 225
state refresh messages, PIM-DM, 699-700
static clients (NTP), 144
static configuration, Frame Relay mapping, 183-184
Static Length Subnet Masking (SLSM), 104
static NAT, 115-116
static routes, redistribute static, 347-348
steady-state operation, 275
by continuing to send joins, PIM-SM, 715-716
steady-state operation (RIP), 206-208
store-and-forward, switches, 23

storing VLAN configurations, 43-44
STP (Spanning Tree Protocol), 57, 61
calculating costs to determine RPs, 63
choosing which ports forward, 61
determining designated ports, 64-65
determining root ports, 63-64
electing root switches, 61-63
command references, 86
configuring, 70-73
converging to STP topology, 65-66
optimizing, 73
BackboneFast, 73, 75
discovery and configuration of PortChannels, 77-78
load balancing PortChannels, 76
PortChannels, 76
PortFast, 73-75
UplinkFast, 73-75
protecting, 82
BPDU Guard, 82-83
Loop Guard, 83-84
Root Guard, 82-83
UDLD, 83-84
topology change notification and updating the CAM, 66-67
transitioning from blocking to forwarding, 67-68
STP forwarding, 61
stratum level (NTP), 144
strict-priority scheduling, 547
Structured Wireless-Aware Network. *See* SWAN
stub network, OSPF LSA types, 278
stub router
configuring OSPF, 303-304
EIGRP, 246-248
stubby areas, OSPF, 287-290
ABRs, 287
stuck-in-active state (EIGRP), 245-246
sub-AS, 399
subinterfaces, queuing, 520
subnet broadcast address, 98
subnet ID field (IPv6 global addresses), 121
subnet numbers
determining
binary method, 98-99
decimal method, 99-101
determining all
binary method, 102-104
decimal method, 104-105
subnets, 94, 97
allocation, 105-106
classful IP addressing, 95-96
primary subnet, 234
route summarization, 107-108
exclusive summary routes (binary method), 110
inclusive summary routes (binary method), 108-109
inclusive summary routes (decimal method), 109-110
size of, 97-98
subnets of traffic, CB Policing, 576-577
Sub-Network Access Protocol (SNAP), 14
successor routes, 240
summaries
LSAs, 287
manual summaries and AS_PATHs (BGP tables), 378-381
summarization. *See also* route summarization
BGP, 429-430
summary-address command, 344
summary-only keyword, 429
Superframe (SF), 592
supernetting, 113
SVIs (switched virtual interfaces), 186

SWAN (Structured Wireless-Aware Network), 828
CiscoWorks WLS
Air/RF scanning and monitoring, 836
assisted site surveys, 835
automatic access point configuration, 834
centralized firmware updates, 835
customizable thresholds, 835
dynamic grouping, 835
fault status, 836
IDS, 836
secure user interfaces, 836
security policy monitoring, 836
self-healing functions, 837
SSIDs, 835
troubleshooting, 837
VLAN configuration, 835
CiscoWorks WLSE, 834
hardware, 831
IDS, 829-830
WDS, 828-829
wireless LAN hardware, 832-834
switch buffering, Ethernet, 10-11
switch ports, 752-753
best practices for unused and user ports, 753-754
802.1X authentication using EAP, 764-766
DAI, 758-761
DHCP snooping, 761-762
IP Source Guard, 763-764
port security, 754-758
switched virtual interfaces (SVIs), 186
switches, 19, 22
command output showing MAC address table, 19, 21
cut-through, 23
Ethernet, 9
fragment-free switches, 23
internal processing, 23
LAN switch forwarding behavior, 19
ports, assigning to VLANs, 53
root switches, electing, 61-63
store-and-forward, 23
switch port configuration, 12-14
unicast forwarding, 19
VLANs, 31
switching, Layer 3 switching, 186
switching paths, 178
IP forwarding, 179
switchport access vlan command, 38, 43
switchport mode command, 49
switchport nonegotiate interface command, 49
switchport port-security maximum command, 756
switchport trunk allowed command, 48
switchport trunk encapsulation command, 49
symmetric active mode (NTP), 144
SYN flag, 156
synchronous serial links, 592
carrier detect and interface resets, 594-595
command references, 604
T1 alarms, 594
T1 framing and encoding, 592-593
System ID Extension, 62

T

T1, 592
alarms, 594
versus E1, 593
framing, 592-593
tables
adjacency tables, 179
ARP and inverse ARP, 179-180
BGP tables. *See* BGP tables

IP routing tables. *See* IP routing tables
TACACS+, 747
tacacs-server host commands, 750
tags, route redistribution, 328-329
Tc, 555-556
tuning shaping for voice, 561-564
TCN (Topology Change Notification), 67
TCP (Transmission Control Protocol), 154
connections and port numbers, 155-156
error recovery, 157
flags, 156
headers, 159
sliding windows, 157-159
versus UDP, 154-155
TCP Congestion Avoidance logic, 158
TCP Congestion Control, 158
TCP intercept, 776-777
TCP SYN flood, 775
TCP/IP
application authentication and privacy, 163
application protocols, 160-161
FTP, passive and active mode, 161-162
HTTP protocol, 856
layers, 855
models, 851
UDP, functions of, 154
TDM (time-division multiplexing), 592
telcos, 592
Temporal Key Integrity Protocol (TKIP), 810-811
terminal editing, 861
terminal history size command, 860
terminology, traffic shaping, 555-556
thresholds, discarding logic, 540
tiebreakers, BGP decision process, 449-450
TIM (traffic indication map), 805
time synchronization, NTP, 143-145
time-division multiplexing (TDM), 592
timers, IGMPv1 and IGMPv2, 659
TKIP (Temporal Key Integrity Protocol), 810-811
token bucket model, 558
tools
BGP filtering tools. *See* BGP filtering tools
NLRI filtering tools, 424
Topology Change Notification (TCN), 67
topology table, EIGRP, 238-240
traffic, multicast traffic, 635, 637
IGMP. *See* IGMP
traffic contracts, 509
traffic indication map (TIM), 805
traffic policers, 551
traffic profiles, WRED, 541
traffic rates, 509
traffic shaping, 555. *See also* shaping
CB Shaping. *See* CB Shaping
egress blocking, 555
Frame Relay, 559
mechanics of, 557-558
shaping rate, 556
terminology, 555-556
with Be, 557
traffic-rate command, FRTS configuration, 567-568
traffic-shape fecn-adapt command, 614
transient groups, multicast IP addresses, 639, 641
transient multicast addresses. *See* transient groups
transit network, OSPF LSA types, 278
transitioning from blocking to forwarding, STP, 67-68
transitive, PAs, 446
Transmission Control Protocol. *See* TCP, 154
transmit power, wireless configuration parameters (IEEE 802.11), 804
transmit queue, 519

Transport Layer Security (TLS), 163
triggered extensions to RIP, 210
triggered updates (RIP), 208-210
troubleshooting CiscoWorks WLSE, 837
trunk configuration compatibility, 48-49
trunk ports, 752
trunking, VLAN trunking, 44
802.1Q, 44-45
ISL, 44-45
ISL configuration, 45-46
trunks
configuration, 53, 779
native VLANs, 766
protecting, 83-84
trusted ports, 752
TTL scoping, 689-690
tuning RIP convergence, 212-213
tunneling, 802.1Q-in-Q, 51-52
twisted pairs, 8
Ethernet, 9
two-rate, three-color policing, 573-575
TX queue, 519
type fields, 18
802.2 LLC, 18
Ethernet, 18

U

UDLD, 83-84
UDLD (UniDirectional Link Detection), 83
UDLD aggressive mode, 83
UDP (User Datagram Protocol), 154
functions of, 154
versus TCP, 154-155
unicast, 632-633
multicast routing, 683
unicast addresses, 16
unicast forwarding, switches, 19
Unicast Source Address (USA), 664
UniDirectional Link Detection (UDLD), 83
Universal/Local (U/L) bit, 17
unsolicited host membership report, IGMPv1, 650
unused ports, 752
best practices for, 753-754
802.1X authentication using EAP, 764-766
DAI, 758-761
DHCP snooping, 761-762
IP Source Guard, 763-764
port security, 754-758
updates
CAM, 66-67
EIGRP, 236-238
UplinkFast, optimizing STP, 73-75
upstream routers, 697
URG (Urgent), 160
Urgent Pointer, 160
USA (Unicast Source Address), 664
User Datagram Protocol. *See* UDP
user mode CLI password protection, 746
user ports, best practices for, 753-754
802.1X authentication using EAP, 764-766
DAI, 758-761
DHCP snooping, 761-762
IP Source Guard, 763-764
port security, 754-758
username commands, 748
username password command, 747
user-priority bits, 493
UTP cabling, 24

V

values, COMMUNITY values, 475, 479
Variable Length Subnet Masking (VLSM), 104
subnet allocation, 105-106
VC (virtual circuit), 610
violating packets, 571
virtual circuits, queuing, 520
virtual LANs. *See* VLANs
virtual links, configuring OSPF, 299-301
Virtual Private Networks (VPNs), 811, 813
Virtual Router Redundancy Protocol (VRRP), 141-143
VLAN configuration, CiscoWorks WLSE, 835
VLAN DAT, 32, 43-44
VLAN database configuration mode, 32-34
VLAN interfaces, MLS logic, 186
VLAN MPLS (VMPLS), 51
VLAN trunking, 44
802.1Q, 44-45
802.1Q configuration, 45-46
allowed and active VLANs, 48
allowed VLANs, 48
configuring on routers, 49-51
ISL, 44-45
configuration, 45-46
trunk configuration compatibility, 48-49
VLAN Trunking Protocol. *See* VTP
VLANs (virtual LANs), 31, 53
active and not pruned, 48
and IP, 31
community VLANs, 37
configuration mode, creating VLANs, 35-36
configuring, 31
VLAN database configuration mode, 32-34
defining, 53, 197
extended-range VLANs, 42
information, displaying, 53, 197, 779
interfaces, assigning, 35
isolated VLANs, 37
layer 2 switches, 31
naming, 53, 197
native VLANs, trunks, 766
normal-range VLANs, 42
primary VLANs, 37
private VLANs, 36-38, 767-768
secondary VLANs, 37
storing configurations, 43-44
trunks, configuration, 53, 779
using configuration mode to put interfaces into VLANs, 34-35
VLSM (Variable Length Subnet Masking), 104
subnet allocation, 105-106
VMPLS (VLAN MPLS), 51
voice, tuning shaping for voice with LLQ and Tc, 561-564
voice services, wireless LANs, 839
VoWLAN (voice over wireless LAN), 839
VPNs (Virtual Private Networks), 811, 813
VRRP (Virtual Router Redundancy Protocol), 141-143
VTP (VLAN Trunking Protocol), 38
configuring, 40-41
extended-range VLANs, 42
normal-range VLANs, 42
revision numbers, 39-40
vty, 751

W

Wait, 308
watch mode, TCP intercept, 776
WC mask (wildcard mask), 424
WCS (Wireless Control System), 833
WDS, 828-829
WDS (Wireless Domain Services), 828
weight packets, WRED, 541
weighted fair queuing. *See* WFQ
Weighted Random Early Detection (WRED), 539-540
  discard categories, 539
WEP (Wired Equivalent Privacy), 799, 810-811
WFQ (weighted fair queuing), 521, 524-525
  command references, 527
  configuration, 527-528
  drop policy, 526-527
  number of queues, 526-527
  queue lengths, 526-527
  show command, 528
  types of queues, 527
WFQ scheduler, 524-526
While AS_SET and AS_CONFED_SET, 445
Wi-Fi, 839, 843-844
Wi-Fi Protected Access (WPA), 811-812
wildcard masks, 424
  IP ACL, 772
windows
  receiver's window, 157
  sliding windows, TCP, 157-159
Wired Equivalent Privacy (WEP), 799, 810-811
wireless bridges, 833
wireless configuration parameters, IEEE 802.11, 801
  data rates, 804-805
  fragmentation, 808
  power-save mode, 805-806
  RF channels, 803-804
  RTS/CTS, 806-808
  SSID, 802-803
  transmit power, 804
Wireless Control System (WCS), 833
Wireless Domain Services (WDS), 828
wireless hardware components, 802.11, 794
  access points, 795
  antennas, 795-796
  bridges, 797
  radio cards, 795
    radio frequency peripherals, 797
    repeaters, 796
    routers, 797
wireless LAN client adapters, 833
wireless LAN controllers, 833
wireless LAN hardware, SWAN, 832-834
wireless LANs
  applying in enterprises, 837
    security, 837-838
    voice services, 839
  public wireless LANs, 840-842
  small office or home, 842-844
wireless medium access, IEEE 802.11, 809-810
wireless security
  comparing, 813
  IEEE 802.11, 810
    AES, 812
    open system authentication, 812
    shared key authentication, 812-813
    TKIP, 811
    VPNs, 813
    WEP, 811
    WPA, 812

wireless system configurations, 802.11, 791
ad hoc mode configuration, 794
infrastructure mode configuration, 792-794
workgroup bridges, 833
WPA (Wi-Fi Protected Access), 811-812
WRED (Weighted Random Early Detection), 539-540
configuration, 542
discard categories, 539
DSCP-based WRED, 541
traffic profiles, 541
weight packets, 541
wrr-queue dscp-map, 546
wrr-queue random-detect, 546

Y–Z

Yellow alarm, 594

zero subnets, 98