1. Bridging and Switching

Task 1.1

```
R5:
interface FastEthernet0/1
!
interface FastEthernet0/1.52
 encapsulation dot1Q 52
 ip address 192.10.1.5 255.255.255.0
!
interface FastEthernet0/1.53
 encapsulation dot1Q 53
 ip address 204.12.1.5 255.255.255.0

SW1 and SW2:
interface range FastEthernet0/13 - 15
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
 no shutdown

SW3:
interface FastEthernet0/5
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
```

Further Reading

Configuring and Troubleshooting FastEthernet 10/100/1000Mb Half/Full Duplex Auto-Negotiation

Troubleshooting Cisco Catalyst Switches to NIC Compatibility Issues

Task 1.1 Verification

```
Rack1SW1#show interfaces trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      on           802.1q         trunking      1
Fa0/14      on           802.1q         trunking      1
Fa0/15      on           802.1q         trunking      1

Port        Vlans allowed on trunk
Fa0/13      1-4094
Fa0/14      1-4094
Fa0/15      1-4094

Port        Vlans allowed and active in management domain
```

```
Fa0/13      1,3-4,7,13,32,45,52-53,67,1001
Fa0/14      1,3-4,7,13,32,45,52-53,67,1001
Fa0/15      1,3-4,7,13,32,45,52-53,67,1001

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      1,3-4,7,13,32,45,52-53,67,1001
Fa0/14      none
Fa0/15      none

Rack1SW3#show interfaces fa0/5 trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/5       on           802.1q         trunking      1

Port        Vlans allowed on trunk
Fa0/5       1-4094

Port        Vlans allowed and active in management domain
Fa0/5       1,3-4,7,13,32,45,52-53,67,1001

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/5       1,3-4,7,13,32,45,52-53,67,1001


Rack1SW3#show interfaces fa0/5 switchport | include Neg
Negotiation of Trunking: Off

Rack1R5#show vlans 52

Virtual LAN ID:  52 (IEEE 802.1Q Encapsulation)

   vLAN Trunk Interface:   FastEthernet0/1.52

   Protocols Configured:   Address:       Received:        Transmitted:
           IP              192.10.1.5             0                   0

   0 packets, 0 bytes input
   0 packets, 0 bytes output

Rack1R5#show vlans 53

Virtual LAN ID:  53 (IEEE 802.1Q Encapsulation)

   vLAN Trunk Interface:   FastEthernet0/1.53

   Protocols Configured:   Address:       Received:        Transmitted:
           IP              204.12.1.5             0                   0

   0 packets, 0 bytes input
   0 packets, 0 bytes output
```

Task 1.2

SW1:

```
interface range FastEthernet0/16 - 18
 switchport trunk encapsulation dot1q
 no shutdown
```

SW3:

```
interface range FastEthernet0/13 - 15
 switchport trunk encapsulation dot1q
 switchport mode dynamic desirable
 no shutdown
```

> ☞ **Quick Note**
> The switchport mode could have been configured on SW1 rather than SW3

Task 1.2 Verification

```
Rack1SW3#show interfaces trunk | exclude Fa0/5|Fa0/19

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      desirable    802.1q         trunking      1
Fa0/14      desirable    802.1q         trunking      1
Fa0/15      desirable    802.1q         trunking      1

Port        Vlans allowed on trunk
Fa0/13      1-4094
Fa0/14      1-4094
Fa0/15      1-4094

Port        Vlans allowed and active in management domain
Fa0/13      1,3-4,7,13,32,45,52-53,67,1001
Fa0/14      1,3-4,7,13,32,45,52-53,67,1001
Fa0/15      1,3-4,7,13,32,45,52-53,67,1001

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      1,3-4,7,13,32,45,52-53,67,1001
Fa0/14      none
Fa0/15      none
```

Task 1.3

SW1:

```
interface range FastEthernet0/19 - 21
 no shutdown
```

SW4:

```
interface range FastEthernet0/13 - 15
 switchport mode dynamic desirable
 no shutdown
```

Task 1.3 Verification

```
Rack1SW4#show interfaces trunk | exclude Fa0/16|Fa0/19

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

Port        Vlans allowed on trunk
Fa0/13      1-4094
Fa0/14      1-4094
Fa0/15      1-4094

Port        Vlans allowed and active in management domain
Fa0/13      1,3-4,7,13,32,45,52-53,67,1001
Fa0/14      1,3-4,7,13,32,45,52-53,67,1001
Fa0/15      1,3-4,7,13,32,45,52-53,67,1001

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      1,3-4,7,13,32,45,52-53,67,1001
Fa0/14      1,3-4,7,13,32,45,52-53,67,1001
Fa0/15      1,3-4,7,13,32,45,52-53,67,1001
```

☞ **Quick Note**

n-isl = negotiated ISL

Task 1.4 and 1.7

```
SW1, SW2, SW3, and SW4:
spanning-tree mode mst
!
spanning-tree mst configuration
instance 1 vlan 3-7
instance 2 vlan 13-45
instance 3 vlan 52-67
instance 4 vlan 1,1001

SW1:
spanning-tree mst 1 root primary

SW2:
spanning-tree mst 2 root primary

SW3:
spanning-tree mst 3 root primary

SW4:
spanning-tree mst 4 root primary
```

Task 1.4 and 1.7 Verification

```
Rack1SW1#show spanning-tree mst 1

##### MST1    vlans mapped:   3-7
Bridge        address 0019.55e6.6580  priority    24577 (24576 sysid 1)
Root          this switch for MST1
<output omitted>

Rack1SW2#show spanning-tree mst 2

##### MST2    vlans mapped:   13-45
Bridge        address 0016.9d31.8380  priority    24578 (24576 sysid 2)
Root          this switch for MST2

Rack1SW3#show spanning-tree mst 3

##### MST3    vlans mapped:   52-67
Bridge        address 0015.63c8.8800  priority    24579 (24576 sysid 3)
Root          this switch for MST3

Rack1SW4#show spanning-tree mst 4

##### MST4    vlans mapped:   1,1001
Bridge        address 000e.83b2.9480  priority    24580 (24576 sysid 4)
Root          this switch for MST4
```

◎ Strategy Tip

Always remember to read ahead. In this case doing Task 1.4 and 1.7 together is the simplest and fastest solution.

Task 1.4 and 1.7 Breakdown

MST uses the concept of instances, where multiple VLANs are mapped to a single instance, rather than having a separate spanning tree instance for each VLAN. By default, all VLANs will be in instance 0, unless you assign them to another instance. After instances have VLANs assigned, the spanning tree root can be adjusted with the spanning tree mst X root primary command, where X is the number of the instance.

☠ Pitfall

When using the spanning tree mode MST, the switches will still accept the command “spanning-tree vlan X root primary”, but it will have no effect on the root election. Make sure that you verify that you have the correct switches configured as root for the respective VLANs, as stated in the section.

Task 1.5

SW2, SW3, and SW4:

```
system mtu 1504
<reload>
```

☞ Quick Note

Although not needed on SW3 for this task, SW3 will be running OSPF with SW4 later in the lab. By ensuring that the MTUs match, it will alleviate any potential problems.

SW2:

```
vlan dot1q tag native
!
interface FastEthernet0/2
 switchport mode dot1q-tunnel
```

SW4:

```
vlan dot1q tag native
!
interface FastEthernet0/6
 switchport mode dot1q-tunnel
```

Task 1.5 Verification

```
Rack1SW2#show run | include vlan 3-4,
vlan 3-4,7,13,32,45,52-53,67,1001

Rack1SW4#show run | include vlan 3-4,
vlan 3-4,7,13,32,45,52-53,67,1001

Rack1SW2#show dot1q-tunnel interface fa0/2

dot1q-tunnel mode LAN Port(s)
-----------------------------
Fa0/2

Rack1SW4#show dot1q-tunnel interface fa0/6

dot1q-tunnel mode LAN Port(s)
-----------------------------
Fa0/6

Rack1R2#ping 174.1.26.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 174.1.26.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Rack1R2#
```

Task 1.6

SW1:

```
interface FastEthernet0/14
 spanning-tree mst 1 port-priority 32
!
interface FastEthernet0/15
 spanning-tree mst 1 port-priority 16
!
interface FastEthernet0/17
 spanning-tree mst 1 port-priority 32
!
interface FastEthernet0/18
 spanning-tree mst 1 port-priority 16
!
interface FastEthernet0/20
 spanning-tree mst 1 port-priority 32
!
interface FastEthernet0/21
 spanning-tree mst 1 port-priority 16
```

Task 1.6 Verification

```
Rack1SW2#show spanning-tree mst 1

##### MST1    vlans mapped:   3-7
Bridge        address 0016.9d31.8380  priority    32769 (32768 sysid 1)
Root          address 0019.55e6.6580  priority    24577 (24576 sysid 1)
              port    Fa0/15          cost          200000    rem hops
19

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/13           Altn BLK 200000    128.15   P2p
Fa0/14           Altn BLK 200000    128.16   P2p
Fa0/15           Root FWD 200000    128.17   P2p
Fa0/19           Altn BLK 200000    128.21   P2p

Rack1SW3#show spanning-tree mst 1

##### MST1    vlans mapped:   3-7
Bridge        address 0015.63c8.8800  priority    32769 (32768 sysid 1)
Root          address 0019.55e6.6580  priority    24577 (24576 sysid 1)
              port    Fa0/15          cost          200000    rem hops
19

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/5            Desg FWD 2000000   128.5    Shr
Fa0/13           Altn BLK 200000    128.13   P2p
Fa0/14           Altn BLK 200000    128.14   P2p
Fa0/15           Root FWD 200000    128.15   P2p
Fa0/19           Altn BLK 200000    128.19   P2p
```

```
Rack1SW4#show spanning-tree mst 1

##### MST1    vlans mapped:   3-7
Bridge        address 000e.83b2.9480  priority    32769 (32768 sysid 1)
Root          address 0019.55e6.6580  priority    24577 (24576 sysid 1)
              port    Fa0/15          cost          200000    rem hops
19

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/13           Altn BLK 200000    128.13   P2p
Fa0/14           Altn BLK 200000    128.14   P2p
Fa0/15           Root FWD 200000    128.15   P2p
Fa0/16           Desg FWD 200000    128.16   P2p
Fa0/19           Desg FWD 200000    128.19   P2p
```

Task 1.7

Solution and verification provided in **Task 1.4**

Task 1.8

SW2:

```
interface Port-channel24
 no switchport
 ip address 174.1.24.8 255.255.255.0
!
interface FastEthernet0/20
 no switchport
 no ip address
 channel-group 24 mode on
 no shut
!
interface FastEthernet0/21
 no switchport
 no ip address
 channel-group 24 mode on
 no shut
```

SW3 and SW4:

```
interface FastEthernet0/20
 no switchport
 no ip address
 channel-group 43 mode on
 no shut
!
interface FastEthernet0/21
 no switchport
 no ip address
 channel-group 43 mode on
 no shut
```

SW3:

```
interface Port-channel43
 no switchport
 ip address 174.1.34.9 255.255.255.0
```

SW4:

```
interface Port-channel43
 no switchport
 ip address 174.1.34.10 255.255.255.0

interface Port-channel24
 no switchport
 ip address 174.1.24.10 255.255.255.0
!
interface FastEthernet0/17
 no switchport
 no ip address
 channel-group 24 mode on
!
interface FastEthernet0/18
 no switchport
 no ip address
 channel-group 24 mode on
```

Task 1.8 Verification

```
Rack1SW4#show etherchannel summary | begin Group
Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------
24     Po24(RU)         -        Fa0/17(P)   Fa0/18(P)
43     Po43(RU)         -        Fa0/20(P)   Fa0/21(P)

Rack1SW4#ping 174.1.24.8

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 174.1.24.8, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms

Rack1SW4#ping 174.1.34.9

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 174.1.34.9, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

Task 1.9

```
SW1:
interface FastEthernet0/3
 duplex full
 speed 100
!
interface FastEthernet0/9
 duplex full
 speed 100
!
interface FastEthernet0/10
 duplex full
 speed 100
!
interface FastEthernet0/11
 duplex full
 speed 100
```

Task 1.9 Breakdown

Errors in speed and duplex negotiation typically result in one side of the link appearing up while the other side appears down, or very low performance on the line. In order to avoid errors in auto negotiation of speed and duplex these values may be hard coded with the interface level commands **speed** and **duplex** respectively. Looking at the output of show vlan brief on the switches will show that ports 3, 9, 10, and 11 on SW1 are in VLAN 3. The section states ALL ports in VLAN 3, so make sure that you also configure for the port connecting to R3. You may also want to configure R3's interface as 100/full, since having one side trying to negotiate and the other side hard-coded can lead to erratic behavior.

Task 1.9 Verification

```
Rack1SW1#show interfaces status | include Fa0/(3|9|10|11)
Fa0/3           connected    3            full    100 10/100BaseTX
Fa0/9           notconnect   3            full    100 10/100BaseTX
Fa0/10          notconnect   3            full    100 10/100BaseTX
Fa0/11          notconnect   3            full    100 10/100BaseTX
```

Task 1.10

R2:

```
username Rack1R3 password CISCO
!
interface Multilink1
 ip address 174.1.23.2 255.255.255.0
 ppp multilink
 ppp multilink group 1
!
interface Serial0/0
 encapsulation frame-relay
 no frame-relay inverse-arp
!
interface Serial0/0.203 point-to-point
 frame-relay interface-dlci 203 ppp Virtual-Template1
!
interface Serial0/0.213 point-to-point
 frame-relay interface-dlci 213 ppp Virtual-Template1
!
interface Virtual-Template1
 ppp multilink
 ppp multilink group 1
 ppp authentication chap
```

R3:

```
username Rack1R2 password CISCO
!
interface Multilink1
 ip address 174.1.23.3 255.255.255.0
 ppp multilink
 ppp multilink group 1
!
interface Serial1/0
 encapsulation frame-relay
 no frame-relay inverse-arp
 frame-relay interface-dlci 302 ppp Virtual-Template1
!
interface Serial1/1
 encapsulation frame-relay
 no frame-relay inverse-arp
 frame-relay interface-dlci 312 ppp Virtual-Template1
!
interface Virtual-Template1
 ppp multilink
 ppp multilink group 1
 ppp authentication chap
```

Task 1.10 Breakdown

Since Frame Relay does not natively support features such as authentication, link quality monitoring, and reliable transmission, it is sometimes advantageous to run PPP over Frame Relay in order to enable these features. A useful feature of PPP that can be used over Frame Relay as well is PPP multilink. This configuration is aptly named Multilink PPP over Frame Relay (MLPoFR).

Note

As of IOS release 12.2(8)T Multilink Frame Relay (FRF.16) is supported on platforms as low as the 2600. Previously this feature was reserved for high end platforms such as the 12000.

The first step in configuring MLPoFR is to define the *multilink* interface. This is the interface where all logical configuration such as IP addressing will be placed. Create this interface by issuing the **`interface multilink [num]`** global configuration command. Next, multilink should be enabled an a group number defined. This is accomplished by issuing the **`ppp multilink`** and **`ppp multilink group [num]`** interface commands.

Next, create a *virtual-template* interface using the **`interface virtual-template [num]`** global command. This interface is where the Frame Relay VCs will be bound. This interface should be part of the previously created multilink group, as well as have any authentication options desired.

Lastly, configure Frame Relay on the physical interfaces in question, and bind the respective VCs to the virtual-template interface. This is accomplished by issuing the **`frame-relay interface-dlci [num] virtual-template [num].`**

Note

PPP over Frame Relay can be configured in the same manner by leaving out the multilink interface and the multilink-group.

☑ Multilink PPP Verification

```
Rack1R3#show ppp multilink

Multilink1, bundle name is Rack1R2
  Bundle up for 00:08:28, 1/255 load
  Receive buffer limit 24384 bytes, frag timeout 1000 ms
    0/0 fragments/bytes in reassembly list
    0 lost fragments, 3 reordered
    0/0 discarded fragments/bytes, 0 lost received
    0x1C received sequence, 0x1A sent sequence
  Member links: 2 active, 1 inactive (max not set, min not set)
  Vi3, since 00:08:28 ← Virtual-Access for the bundle
  Vi2, since 00:08:27 ← Virtual-Access for the individual link
  Vt1 (inactive) ← Virtual-Template not used, Virtual-Access is
                    the individual instance of the template
```

Task 1.10 Verification

```
Rack1R2#show ppp multilink

Multilink1, bundle name is Rack1R3
  Bundle up for 00:05:46, 1/255 load
  Receive buffer limit 24384 bytes, frag timeout 1000 ms
    0/0 fragments/bytes in reassembly list
    0 lost fragments, 0 reordered
    0/0 discarded fragments/bytes, 0 lost received
    0x16 received sequence, 0x14 sent sequence
  Member links: 2 active, 1 inactive (max not set, min not set)
    Vi3, since 00:05:46
    Vi2, since 00:05:40
    Vt1 (inactive)
```

```
Rack1R2#ping 174.1.23.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 174.1.23.3, timeout is 2 seconds:
!!!!!
      Success rate is 100 percent (5/5), round-trip min/avg/max =
20/23/24 ms
```

You can also run an extended ping and then verify that both PVCs are used by looking at the output of **show frame pvc** and verifying that the packet counts are similar for both PVCs.

```
Rack1R3(config-if)#do show frame pvc 312

PVC Statistics for interface Serial1/1 (Frame Relay DTE)

DLCI = 312, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial1/1

  input pkts 2093          output pkts 2080         in bytes 124677
  out bytes 126714         dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 0         out bcast bytes 0
  5 minute input rate 0 bits/sec, 8 packets/sec
  5 minute output rate 0 bits/sec, 8 packets/sec
  pvc create time 01:43:21, last time pvc status changed 01:38:01
  Bound to Virtual-Access2 (up, cloned from Virtual-Template1)
Rack1R3(config-if)#do show frame pvc 302

PVC Statistics for interface Serial1/0 (Frame Relay DTE)

DLCI = 302, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial1/0

  input pkts 2081          output pkts 2155         in bytes 128397
  out bytes 146673         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 72        out bcast bytes 23904
  5 minute input rate 0 bits/sec, 7 packets/sec
  5 minute output rate 0 bits/sec, 7 packets/sec
  pvc create time 02:05:02, last time pvc status changed 01:38:04
  Bound to Virtual-Access1 (up, cloned from Virtual-Template1)
Rack1R3(config-if)#
```

Task 1.11

```
R6:
interface Serial0/0
 encapsulation frame-relay
 no frame-relay inverse-arp ip 51
 no frame-relay inverse-arp ip 101
 no frame-relay inverse-arp ip 201
 no frame-relay inverse-arp ip 301
 no frame-relay inverse-arp ip 401
```

Task 1.11 Breakdown

For this task, we will need to discover all of the DLCIs that are being learned through LMI from the Frame Relay switch. To do this, shutdown the interface, enable Frame Relay encapsulation, disable inverse-ARP, and finally bring the interface back up by doing a **`no shutdown`**. Finally issue the **`show frame-relay pvc`** command and include only the “DLCI” keyword. This will simplify the output.

```
Rack1R6#show run interface s0/0
Building configuration...

Current configuration : 147 bytes
!
interface Serial0/0
 ip address 54.1.2.6 255.255.255.0
 encapsulation frame-relay
 no frame-relay inverse-arp
end

Rack1R6#show frame-relay pvc | include DLCI
DLCI = 51, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0
DLCI = 100, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0
DLCI = 101, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0
DLCI = 201, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0
DLCI = 301, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0
DLCI = 401, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0
Rack1R6#
```

Now that the DLCIs have been discovered we can disable inverse-ARP for the unused DLCIs and re-enable it for the whole interface by using the **`frame-relay inverse-arp`** command.

2. IP IGP Routing

Task 2.1

```
R2:
interface FastEthernet0/0.26
 ip ospf authentication
 ip ospf authentication-key CISCO
!
router ospf 1
 router-id 150.1.2.2
 network 150.1.2.2 0.0.0.0 area 0
 network 174.1.23.2 0.0.0.0 area 0
 network 174.1.26.2 0.0.0.0 area 0

R3:
router ospf 1
 router-id 150.1.3.3
 network 150.1.3.3 0.0.0.0 area 0
 network 174.1.23.3 0.0.0.0 area 0

R6:
interface FastEthernet0/1.26
 ip ospf authentication
 ip ospf authentication-key CISCO
!
router ospf 1
 router-id 150.1.6.6
 network 174.1.26.6 0.0.0.0 area 0
```

Task 2.1 Breakdown

The above task states to authenticate the OSPF adjacency between R2 and R6. Although this task can be accomplished by authenticating all area 0 adjacencies by issuing the **`area 0 authentication`** command under the OSPF process, the wording of the question implies that interface authentication between R2 and R6 should be used. To enable interface authentication, issue **`the ip ospf authentication [message-digest]`** interface level command. To define the authentication key, issue the interface level **`command ip ospf authentication-key [key].`** Type 0 is no authentication, type 1 is plaintext, and type 2 is MD5.

Task 2.1 Verification

Verify the OSPF neighbors:

```
Rack1R6#show ip ospf neighbor

Neighbor ID   Pri   State   Dead Time   Address      Interface
150.1.2.2      1   FULL/DR 00:00:35    174.1.26.2 FastEthernet0/1.26
```

Verify that the OSPF adjacency between R2 and R6 is authenticated:

```
Rack1R6#show ip ospf interface g0/1.26
FastEthernet0/1.26 is up, line protocol is up
  Internet Address 174.1.26.6/24, Area 0
  Process ID 1, Router ID 150.1.6.6, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State BDR, Priority 1
  Designated Router (ID) 150.1.2.2, Interface address 174.1.26.2
  Backup Designated router (ID) 150.1.6.6, Interface address 174.1.26.6
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:07
  Supports Link-local Signaling (LLS)
  Index 1/1, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.2.2  (Designated Router)
  Suppress hello for 0 neighbor(s)
  Simple password authentication enabled
```

Task 2.2

R1:

```
interface FastEthernet0/0.13
 ip ospf mtu-ignore
!
router ospf 1
 area 38 stub
 network 174.1.31.1 0.0.0.0 area 38
```

R3:

```
interface FastEthernet0/1
 ip ospf mtu-ignore
!
router ospf 1
 area 38 stub
 network 174.1.38.3 0.0.0.0 area 38
```

SW2:

```
ip routing
!
router ospf 1
 router-id 150.1.8.8
 area 38 stub
 network 174.1.24.8 0.0.0.0 area 38
 network 174.1.38.8 0.0.0.0 area 38
 network 150.1.8.8 0.0.0.0 area 38
```

SW3:

```
ip routing
!
router ospf 1
 router-id 150.1.9.9
 area 38 stub
 network 174.1.34.9 0.0.0.0 area 38
 network 150.1.9.9 0.0.0.0 area 38
 network 174.1.31.9 0.0.0.0 area 38
```

SW4:

```
ip routing
!
router ospf 1
 router-id 150.1.10.10
 area 38 stub
 network 174.1.24.10 0.0.0.0 area 38
 network 174.1.34.10 0.0.0.0 area 38
 network 150.1.10.10 0.0.0.0 area 38
```

Task 2.2 Breakdown

The above task requires that external routing information not be injected into OSPF area 38. However, inter area routing information should be preserved. This task can be accomplished by defining area 38 as either a stub area or a not-so-stubby area (NSSA). Both stub and NSSA areas will block external (LSA 5) OSPF information, but will preserve inter-area (LSA 3 and 4) OSPF information. If an NSSA is used to meet this task, ensure to add the **default-information** statement in order to ensure that SW2 has full reachability to all networks external to the routing domain.

Task 2.2 Verification

Verify the stub area configuration:

```
Rack1SW2#show ip ospf | begin Area 38
    Area 38
        Number of interfaces in this area is 1
        It is a stub area
        Area has no authentication
        SPF algorithm last executed 00:00:34.524 ago
        SPF algorithm executed 4 times
        Area ranges are
        Number of LSA 8. Checksum Sum 0x037AA3
        Number of opaque link LSA 0. Checksum Sum 0x000000
        Number of DCbitless LSA 0
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0
```

Verify the OSPF routes on SW3 (note the default and inter-area routes):

```
Rack1SW2#show ip route ospf
174.1.0.0/24 is subnetted, 6 subnets
O       174.1.38.0 [110/3] via 174.1.34.10, 00:02:07, Port-channel43
O IA    174.1.26.0 [110/5] via 174.1.34.10, 00:00:17, Port-channel43
O       174.1.24.0 [110/2] via 174.1.34.10, 00:02:07, Port-channel43
O IA    174.1.23.0 [110/4] via 174.1.34.10, 00:00:37, Port-channel43
     150.1.0.0/16 is variably subnetted, 5 subnets, 2 masks
O       150.1.10.10/32 [110/2] via 174.1.34.10, 00:02:07, Port-
channel43
O       150.1.8.8/32 [110/3] via 174.1.34.10, 00:02:07, Port-channel43
O IA    150.1.3.3/32 [110/4] via 174.1.34.10, 00:02:07, Port-channel43
O IA    150.1.2.2/32 [110/5] via 174.1.34.10, 00:00:17, Port-channel43
O*IA 0.0.0.0/0 [110/4] via 174.1.34.10, 00:02:07, Port-channel43
```

Task 2.3

R6:

```
interface FastEthernet0/0
 ip ospf mtu-ignore
!
router ospf 1
 network 150.1.6.6 0.0.0.0 area 67
 network 174.1.67.6 0.0.0.0 area 67
 area 67 range 150.1.6.0 255.255.254.0
```

SW1:

```
ip routing
!
router ospf 1
 router-id 150.1.7.7
 network 150.1.7.7 0.0.0.0 area 67
 network 174.1.67.7 0.0.0.0 area 67
```

Task 2.3 Breakdown

The above task requires networks 150.1.6.0/24 and 150.1.7.0/24 to be summarized without overlapping address space. As previously shown, the first step in summarizing address space is to write the address out in binary. As it is evident that the first 16 bits of the above addresses are the same (150.1), the summary will start in the second octet.

	128	64	32	16	8	4	2	1
6	0	0	0	0	0	1	1	0
7	0	0	0	0	0	1	1	1

From the above table, it can be seen that the first seven bits of the third octet are also the same. Therefore, the summary for these two networks is 150.1.0.0/(16 + 7) = 150.1.0.0/23, or 150.1.0.0 255.255.254.0.

Since the above summary is created as internal, OSPF prefixes are moving between areas, the **`area range`** syntax is used. The **`summary-address`** keyword is only used when external address space (redistributed prefixes) are summarized.

🕮 Further Reading

OSPF Design Guide: OSPF and Route Summarization

Task 2.3 Verification

Verify the summary route generation:

```
Rack1R2#show ip route 150.1.6.0
Routing entry for 150.1.6.0/23
  Known via "ospf 1", distance 110, metric 2, type inter area
  Last update from 174.1.26.6 on FastEthernet0/0.26, 00:02:00 ago
  Routing Descriptor Blocks:
  * 174.1.26.6, from 150.1.6.6, 00:02:00 ago, via FastEthernet0/0.26
      Route metric is 2, traffic share count is 1
```

Task 2.4

```
R1:
interface Serial0/0
 no ip split-horizon eigrp 1024
!
router eigrp 1024
 network 150.1.1.1 0.0.0.0
 network 174.1.13.1 0.0.0.0
 network 174.1.145.1 0.0.0.0
 no auto-summary
 eigrp router-id 150.1.1.1

R3:
router eigrp 1024
 network 150.1.3.3 0.0.0.0
 network 174.1.13.3 0.0.0.0
 no auto-summary
 eigrp router-id 150.1.3.3

R4:
router eigrp 1024
 network 150.1.4.4 0.0.0.0
 network 174.1.45.4 0.0.0.0
 network 174.1.145.4 0.0.0.0
 no auto-summary
 eigrp router-id 150.1.4.4

R5:
router eigrp 1024
 network 150.1.5.5 0.0.0.0
 network 174.1.45.5 0.0.0.0
 network 174.1.145.5 0.0.0.0
 no auto-summary
 eigrp router-id 150.1.5.5
```

Task 2.4 Breakdown

The above section illustrates a basic EIGRP configuration. Note that EIGRP split horizon must be disabled on R1's Serial interface, as this interface connects to both R4 and R5. Therefore, when R1 learns a route in this Serial interface from R4, it can be advertised back out the same interface to reach R5. By default, EIGRP split-horizon is on, and would result in partial reach ability by R4 and R5 to the routing domain.

Further Reading

EIGRP: Split Horizon and Poison Reverse

Task 2.4 Verification

Verify the EIGRP neighbors. For instance, at R1:

```
Rack1R1#show ip eigrp neighbors
IP-EIGRP neighbors for process 1024
H   Address         Interface       Hold Uptime   SRTT   RTO  Q  Seq Type
                                    (sec)         (ms)       Cnt Num
2   174.1.13.3      Se0/1             14 00:00:12   32   200  0  3
1   174.1.145.5     Se0/0            172 00:20:50   25   200  0  6
0   174.1.145.4     Se0/0            172 00:21:03   23   200  0  6
```

Verify EIGRP routes on spoke routers:

```
Rack1R4#show ip route eigrp
     174.1.0.0/24 is subnetted, 4 subnets
D       174.1.13.0 [90/2681856] via 174.1.145.1, 00:00:36, Serial0/0
     150.1.0.0/24 is subnetted, 4 subnets
D       150.1.5.0 [90/2809856] via 174.1.145.1, 00:20:58, Serial0/0
D       150.1.3.0 [90/2809856] via 174.1.145.1, 00:00:34, Serial0/0
D       150.1.1.0 [90/2297856] via 174.1.145.1, 00:21:01, Serial0/0
```

Note that R4 receives R5 prefixes due to EIGRP split-horizon disabled on R1.

Task 2.5

```
R5:
key chain RIP
 key 1
  key-string CISCO
!
interface FastEthernet0/1.52
 ip rip authentication mode md5
 ip rip authentication key-chain RIP
!
router rip
 version 2
 network 192.10.1.0
```

Task 2.5 Breakdown

The above task illustrates a simple RIPv2 configuration with MD5 authentication. First, a key chain is defined. Next, the key number 1 is defined with the key-string CISCO. Authentication is enabled on the interface with the **`ip rip authentication mode md5`** command. Lastly, the key chain is applied to the interface with the **`ip rip authentication key-chain [name]`** command.

Task 2.5 Verification

Verify the RIP configuration:

```
Rack1R5#show ip protocols | begin rip
Routing Protocol is "rip"
  Sending updates every 30 seconds, next due in 12 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
    FastEthernet0/1.52        2     2                    RIP
  Automatic network summarization is in effect
  Maximum path: 4
  Routing for Networks:
    192.10.1.0
  Routing Information Sources:
    Gateway         Distance      Last Update
    192.10.1.254         120      00:00:20
  Distance: (default is 120)
```

Verify the incoming RIP updates:

```
Rack1R5#debug ip rip
 RIP: received packet with MD5 authentication
 RIP: received v2 update from 192.10.1.254 on FastEthernet0/1.52
      205.90.31.0/24 via 0.0.0.0 in 7 hops
      220.20.3.0/24 via 0.0.0.0 in 7 hops
      222.22.2.0/24 via 0.0.0.0 in 7 hops
```

Task 2.6

R1 and R3:

```
router eigrp 1024
 redistribute ospf 1 metric 10000 10 255 1 1500
```

R3:

```
router ospf 1
 redistribute eigrp 1024 subnets
```

☞ **Quick Note**
Since OSPF area 38 is a stub area, redistribution of EIGRP into OSPF on R1 is not needed since it will have no effect on reachability

R5:

```
router eigrp 1024
 redistribute rip metric 10000 10 255 1 1500
!
router rip
 redistribute eigrp 1024 metric 1
```

Task 2.6 Verification

Use the following TCL script to test connectivity between internal routers, as well as connectivity to backbone RIP networks, received from BB2.

```
foreach i {
174.1.145.1
150.1.1.1
174.1.13.1
174.1.31.1
150.1.2.2
174.1.26.2
174.1.23.2
174.1.38.3
150.1.3.3
174.1.13.3
174.1.23.3
174.1.145.4
174.1.45.4
150.1.4.4
174.1.145.5
174.1.45.5
150.1.5.5
192.10.1.5
150.1.6.6
174.1.26.6
174.1.67.6
150.1.7.7
174.1.67.7
174.1.38.8
174.1.24.8
150.1.8.8
174.1.31.9
150.1.9.9
174.1.34.9
174.1.24.10
174.1.34.10
150.1.10.10
222.22.2.1
220.20.3.1
205.90.31.1
} {  ping $i }
```

Note that VLAN3, VLAN4, VLAN7, VLAN1001 and Frame Relay link to BB1 are excluded from IGP, and therefore could not be pinged.

Task 2.7

R1:
```
interface Serial0/0
 delay 1
```

R4:
```
interface Serial0/0
 delay 21500
 !
interface FastEthernet0/1
 delay 6000
!
router eigrp 1024
 variance 4
```

R5:
```
interface Serial0/0
 delay 1
```

Task 2.7 Breakdown

This task can be easily verified by using the **`show ip route <any prefix from BB2>`** command to view the “traffic share count”. Any metric manipulation that achieves the traffic share count of 4 for the FastEthernet link and 1 for the Frame Relay link is acceptable. In this solution, delay was used. Remember that all tasks are not graded by viewing the router’s configuration and this task would be graded by using the **`show ip route`** command.

Task 2.7 Verification

Verify the traffic share counters:

```
Rack1R4#show ip route 222.22.2.0
Routing entry for 222.22.2.0/24
  Known via "eigrp 1024", distance 170, metric 1794560, type external
  Redistributing via eigrp 1024
  Last update from 174.1.145.1 on Serial0/0, 00:00:38 ago
  Routing Descriptor Blocks:
    174.1.145.1, from 174.1.145.1, 00:00:38 ago, via Serial0/0
      Route metric is 7164672, traffic share count is 1
      Total delay is 215110 microseconds,minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
  * 174.1.45.5, from 174.1.45.5, 00:00:38 ago, via FastEthernet0/1
      Route metric is 1794560, traffic share count is 4
      Total delay is 60100 microseconds,minimum bandwidth is 10000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
```

Task 2.8

R1:

```
ipv6 unicast-routing
!
interface FastEthernet0/0.1001
 ipv6 address FEC0:CC1E:1:1::1/64
!
interface Serial0/0
 ipv6 address 2001:CC1E:1:1515::1/64
 ipv6 address FE80::1 link-local
 frame-relay map ipv6 2001:CC1E:1:1515::5 105 broadcast
 frame-relay map ipv6 FE80::5 105
```

R4:

```
ipv6 unicast-routing
!
interface FastEthernet0/1
 ipv6 address 2001:CC1E:1:45::4/64
 ipv6 rip RIPng enable
!
interface FastEthernet 0/0
 ipv6 address FEC0:CC1E:1:4::4/64
 ipv6 rip RIPng enable
```

R5:

```
ipv6 unicast-routing
!
interface FastEthernet0/0
 ipv6 address 2001:CC1E:1:45::5/64
 ipv6 rip RIPng enable
!
interface Serial0/0
 ipv6 address 2001:CC1E:1:1515::5/64
 ipv6 address FE80::5 link-local
 frame-relay map ipv6 2001:CC1E:1:1515::1 501 broadcast
 frame-relay map ipv6 FE80::1 501
!
ipv6 router rip RIPng
```

Task 2.8 Verification

```
Rack1R5#show frame-relay map
Serial0/0 (up): ip 174.1.145.1 dlci 501(0x1F5,0x7C50), static,
              broadcast,
              CISCO, status defined, active
Serial0/0 (up): ip 174.1.145.4 dlci 501(0x1F5,0x7C50), static,
              CISCO, status defined, active
Serial0/0 (up): ipv6 2001:CC1E:1:1515::1 dlci 501(0x1F5,0x7C50),
static,
              broadcast,
              CISCO, status defined, active
Serial0/0 (up): ipv6 FE80::1 dlci 501(0x1F5,0x7C50), static,
              CISCO, status defined, active

Rack1R5#ping 2001:CC1E:1:1515::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:1515::1, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/32 ms
```

Verify that R5 receives IPv6 prefix from R4:

```
Rack1R5#show ipv6 route rip
IPv6 Routing Table - 7 entries
<snip>
R   FEC0:CC1E:1:4::/64 [120/2]
     via FE80::20F:24FF:FE42:EC01, FastEthernet0/0
```

Task 2.9

R1:

```
interface FastEthernet0/0.1001
 ipv6 ospf 1 area 0
!
interface Serial0/0
 ipv6 ospf neighbor FE80::5
 ipv6 ospf 1 area 0
!
```

R5:

```
interface Serial0/0
 ipv6 ospf priority 0
 ipv6 ospf 1 area 0
!
```

Task 2.9 Verification

Verify the OSPFv3 neighbors. Make sure that you give the adjacency time to form.

```
Rack1R1#show ipv6 ospf neighbor

Neighbor ID   Pri   State        Dead Time   Interface ID    Interface
150.1.5.5      0   FULL/DROTHER  00:01:47    3               Serial0/0
```

Check OSPF routes at R5:

```
Rack1R5#show ipv6 route ospf
IPv6 Routing Table - 10 entries
<output omitted>
O   FEC0:CC1E:1:1::/64 [110/65]
     via FE80::1, Serial0/0
```

Task 2.10

R5:

```
ipv6 route ::/0 Null0
!
ipv6 router ospf 1
 default-information originate
```

Task 2.10 Verification

Verify the default route in the routing table of R1:

```
Rack1R1#show ipv6 route ospf
IPv6 Routing Table - 7 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
OE2  ::/0 [110/1], tag 1
     via FE80::5, Serial0/0
```

Task 2.11

```
R5:
ipv6 router rip RIPng
 redistribute ospf 1 metric 1
```

Task 2.11 Verification

```
Rack1R1#show ipv6 route ospf
IPv6 Routing Table - 7 entries
<snip>
OE2  ::/0 [110/1], tag 1
     via FE80::5, Serial0/0
Rack1R1#

Rack1R4#show ipv6 route rip
IPv6 Routing Table - 8 entries
<snip>
R   2001:CC1E:1:1515::/64 [120/2]
     via FE80::20F:24FF:FE42:F0A0, FastEthernet0/1
R   FEC0:CC1E:1:1::/64 [120/2]
     via FE80::20F:24FF:FE42:F0A0, FastEthernet0/1
```

Task 2.11 Breakdown

R1 has a default route pointing towards R5. R4 does not have reachability to VLAN 1001. In order to pass reachability to RIP, there needs to be some sort of redistribution into RIP. There is no need for redistribution into OSPF, due to the default route that R5 is originating into OSPF. There are a couple different options. You could redistribute static, adding the local default into RIP. You could also redistribute from OSPF into RIP. Normally with IPv6 redistribution, you would also need to redistribute connected to have full reachability. Since the section just states that R4 needs reachability to R1’s VLAN 1001 network, adding redistribute connected is not necessary.

3. BGP

Task 3.1

R1:

```
router bgp 65145
 bgp router-id 150.1.1.1
 bgp confederation identifier 100
 bgp confederation peers 65038
 neighbor 150.1.4.4 remote-as 65145
 neighbor 150.1.4.4 update-source Loopback0
 neighbor 150.1.5.5 remote-as 65145
 neighbor 150.1.5.5 update-source Loopback0
 neighbor 174.1.13.3 remote-as 65038
```

R2:

```
router bgp 65267
 bgp router-id 150.1.2.2
 bgp confederation identifier 100
 bgp confederation peers 65038
 neighbor 174.1.23.3 remote-as 65038
 neighbor 174.1.26.6 remote-as 65267
```

R3:

```
router bgp 65038
 bgp router-id 150.1.3.3
 bgp confederation identifier 100
 bgp confederation peers 65145 65267
 neighbor 174.1.13.1 remote-as 65145
 neighbor 174.1.23.2 remote-as 65267
 neighbor 174.1.38.8 remote-as 65038
```

R4:

```
router bgp 65145
 bgp router-id 150.1.4.4
 bgp confederation identifier 100
 neighbor 150.1.1.1 remote-as 65145
 neighbor 150.1.1.1 update-source Loopback0
 neighbor 150.1.5.5 remote-as 65145
 neighbor 150.1.5.5 update-source Loopback0
```

R5:

```
router bgp 65145
 bgp router-id 150.1.5.5
 bgp confederation identifier 100
 neighbor 150.1.1.1 remote-as 65145
 neighbor 150.1.1.1 update-source Loopback0
 neighbor 150.1.4.4 remote-as 65145
 neighbor 150.1.4.4 update-source Loopback0
 neighbor 204.12.1.254 remote-as 54
```

R6:

```
router bgp 65267
 bgp router-id 150.1.6.6
 bgp confederation identifier 100
 neighbor 54.1.2.254 remote-as 54
 neighbor 174.1.26.2 remote-as 65267
 neighbor 174.1.26.2 route-reflector-client
 neighbor 174.1.67.7 remote-as 65267
 neighbor 174.1.67.7 route-reflector-client
```

SW1:

```
router bgp 65267
 bgp router-id 150.1.7.7
 bgp confederation identifier 100
 neighbor 174.1.67.6 remote-as 65267
```

SW2:

```
router bgp 65038
 bgp router-id 150.1.8.8
 bgp confederation identifier 100
 neighbor 174.1.38.3 remote-as 65038
```

Task 3.1 Breakdown

The above BGP setup centers on the confederated AS 100. AS 100 has been divided into three sub-autonomous systems, AS 65038, AS 65145, and AS 65267. Recall from Lab 7 Task 3.1 that BGP confederation still requires either full mesh or route reflection between iBGP peers within the sub-AS.

Inside sub-AS 65145, full iBGP mesh is maintained. However, within AS 65267, only a partial iBGP mesh is maintained. For this reason, R6 has been designated a route-reflector for this sub-AS.

Task 3.1 Verification

Verify the BGP peering. For instance, on R6:

```
Rack1R6#show ip bgp summary  | begin Neighbor
Neighbor    V AS MsgRcvd MsgSent TblVer  InQ OutQ Up/Down  State/PfxRcd
54.1.2.254  4 54      14    10       11    0    0 00:06:39       10
174.1.26.2  4 65267    5     9       11    0    0 00:01:28        0
174.1.67.7  4 65267    9    13       11    0    0 00:06:38        0
```

Verify that R6 acts as route-reflector:

```
Rack1R6#show ip bgp neighbors 174.1.26.2 | include Reflect
  Route-Reflector Client
```

```
Rack1R6#show ip bgp neighbors 174.1.67.7 | include Reflect
  Route-Reflector Client
```

Task 3.2

R5:

```
router bgp 65145
 redistribute static
 neighbor 150.1.4.4 route-map TO_R4 out
 neighbor 150.1.1.1 route-map TO_R1 out
!
ip route 174.1.0.0 255.255.0.0 Null0
!
ip prefix-list AGGREGATE seq 5 permit 174.1.0.0/16
!
route-map TO_R4 deny 10
 match ip address prefix-list AGGREGATE
!
route-map TO_R4 permit 1000
!
route-map TO_R1 deny 10
 match ip address prefix-list AGGREGATE
!
route-map TO_R1 permit 1000
```

R6:

```
router bgp 65267
 redistribute static
 neighbor 174.1.26.2 route-map TO_R2 out
 neighbor 174.1.67.7 route-map TO_SW1 out
!
ip route 174.1.0.0 255.255.0.0 Null0
!
ip prefix-list AGGREGATE seq 5 permit 174.1.0.0/16
!
route-map TO_R2 deny 10
 match ip address prefix-list AGGREGATE
!
```

```
route-map TO_R2 permit 1000
!
route-map TO_SW1 deny 10
 match ip address prefix-list AGGREGATE
!
route-map TO_SW1 permit 1000
```

Task 3.2 Verification

Verify the prefixes advertised to the backbone routers:

```
Rack1R5#show ip bgp neigh 204.12.1.254 advertised-routes  | begin Net
   Network          Next Hop            Metric LocPrf Weight Path
*> 174.1.0.0        0.0.0.0                  0         32768 ?

Total number of prefixes 1

Rack1R6#show ip bgp neigh 54.1.2.254 advertised-routes | begin Net
   Network          Next Hop            Metric LocPrf Weight Path
*> 174.1.0.0        0.0.0.0                  0         32768 ?

Total number of prefixes 1
```

Note that only the summary prefix is advertised. Next verify that the summary is filtered for announcements sent to internal routers:

```
Rack1R6#show ip bgp neigh 174.1.26.2 advertised-routes  | include 174
Rack1R6#

Rack1R2#show ip bgp 174.1.0.0 255.255.0.0
% Network not in table
```

Task 3.3

R5:

```
route-map TO_R4 permit 1000
 set ip next-hop peer-address
!
route-map TO_R1 permit 1000
 set ip next-hop peer-address
```

R6:

```
route-map TO_R2 permit 1000
 set ip next-hop peer-address
!
route-map TO_SW1 permit 1000
 set ip next-hop peer-address
```

☠ Pitfall

When altering the next hop with eBGP, ensure that eBGP multihop is enabled when altering the next hop to an IP address that is not directly connected.

Task 3.3 Verification

Verify that the next-hop is changed:

```
Rack1R6#debug ip bgp updates
Rack1R6#clear ip bgp * soft out
Rack1R6#
  BGP(0): 54.1.2.254 send UPDATE (format) 174.1.0.0/16, next 54.1.2.6,
metric 0, path Local
  BGP(0): 174.1.26.2 send UPDATE (format) 115.0.0.0/8, next 174.1.26.6,
metric 0, path 54
  BGP(0): 174.1.67.7 send UPDATE (format) 115.0.0.0/8, next 174.1.67.6,
metric 0, path 54
  BGP(0): 174.1.26.2 send UPDATE (prepend, chgflags: 0x0) 114.0.0.0/8,
next 174.1.26.6, metric 0, path 54
  BGP(0): 174.1.67.7 send UPDATE (prepend, chgflags: 0x0) 114.0.0.0/8,
next 174.1.67.6, metric 0, path 54
  BGP(0): 174.1.26.2 send UPDATE (format) 28.119.17.0/24, next
174.1.26.6, metric 0, path 54
  BGP(0): 174.1.67.7 send UPDATE (format) 28.119.17.0/24, next
174.1.67.6, metric 0, path 54
  BGP(0): 174.1.26.2 send UPDATE (prepend, chgflags: 0x0)
28.119.16.0/24, next 174.1.26.6, metric 0, path 54
<output omitted>
```

Task 3.4

R3:

```
router bgp 65038
 network 174.1.3.0 mask 255.255.255.0
```

R4:

```
router bgp 65145
 network 174.1.4.0 mask 255.255.255.0
```

R5:

```
router bgp 65145
 neighbor 204.12.1.254 route-map MED out
 neighbor 204.12.1.254 send-community
!
ip prefix VLAN_4 permit 174.1.4.0/24
ip prefix VLANS_3_&_7 permit 174.1.3.0/24
ip prefix VLANS_3_&_7 permit 174.1.7.0/24
!
route-map MED permit 10
 match ip address prefix-list VLAN_4
 set metric 20
 set community no-export
!
route-map MED permit 20
 match ip address prefix-list VLANS_3_&_7
 set metric 10
 set community no-export
!
route-map MED permit 30
```

```
R6:
router bgp 65267
 neighbor 54.1.2.254 route-map MED out
 neighbor 54.1.2.254 send-community
!
ip prefix VLAN_4 permit 174.1.4.0/24
ip prefix VLANS_3_&_7 permit 174.1.3.0/24
ip prefix VLANS_3_&_7 permit 174.1.7.0/24
!
route-map MED permit 10
 match ip address prefix-list VLAN_4
 set metric 10
 set community no-export
!
route-map MED permit 20
 match ip address prefix-list VLANS_3_&_7
 set metric 20
 set community no-export
!
route-map MED permit 30

SW1:
router bgp 65267
 network 174.1.7.0 mask 255.255.255.0
```

Task 3.4 Breakdown

In order to affect inbound traffic flow, either AS-Path or MED should be modified. MED is only compared (by default) between prefixes learned from the same autonomous system. Since AS 100 is dual homed to AS 54, this is an appropriate case to modify the multi-exit discriminator value.

Task 3.4 Verification

Verify that communities are enabled:

```
Rack1R6#show ip bgp neighbors 54.1.2.254 | include Comm
  Community attribute sent to this neighbor
```

Next, verify BGP table at backbone routers, e.g. on BB1. Look for prefixes of VLAN4, VLAN7 and VLAN3:

```
BB1>show ip bgp 174.1.4.0
BGP routing table entry for 174.1.4.0/24, version 578
Paths: (1 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
  Advertised to non peer-group peers:
  172.16.4.3
  100
    54.1.2.6 from 54.1.2.6 (150.1.6.6)
      Origin IGP, metric 10, localpref 100, valid, external, best
      Community: no-export
```

```
BB1>show ip bgp 174.1.7.0
BGP routing table entry for 174.1.7.0/24, version 581
Paths: (2 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
  Not advertised to any peer
  100
    172.16.4.3 from 172.16.4.3 (31.3.0.1)
      Origin IGP, metric 10, localpref 100, valid, internal, best
      Community: no-export
  100
    54.1.2.6 from 54.1.2.6 (150.1.6.6)
      Origin IGP, metric 20, localpref 100, valid, external
      Community: no-export

BB1>show ip bgp 174.1.3.0
BGP routing table entry for 174.1.3.0/24, version 580
Paths: (2 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
  Not advertised to any peer
  100
    172.16.4.3 from 172.16.4.3 (31.3.0.1)
      Origin IGP, metric 10, localpref 100, valid, internal, best
      Community: no-export
  100
    54.1.2.6 from 54.1.2.6 (150.1.6.6)
      Origin IGP, metric 20, localpref 100, valid, external
      Community: no-export
```

Note that BB1 has selected the best path to 174.1.4.0/24 via R6 and the best path to 174.1.7.0/24 & 174.1.3.0/24 via BB3.

Task 3.5

```
R1:
router bgp 65145
 network 174.1.1.0 mask 255.255.255.0 route-map LOCAL_AS
 neighbor 150.1.5.5 send-community
!
route-map LOCAL_AS permit 10
 set community local-AS
```

Task 3.5 Breakdown

The well known community of local-AS is a special case of the community no-export. This community is used to ensure that a prefix is not advertised outside of a sub-AS in a confederation. Had this prefix been set to the community no-export, it would have been advertised between AS 65038, AS 65145, and AS 65267.

Task 3.5 Verification

Verify that the summary prefix on R5:

```
Rack1R5#show ip bgp 174.1.1.0
BGP routing table entry for 174.1.1.0/24, version 17
Paths: (1 available, best #1, table Default-IP-Routing-Table, not
advertised outside local AS)
Flag: 0x880
  Not advertised to any peer
  Local
    150.1.1.1 (metric 1786112) from 150.1.1.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, confed-internal, best
      Community: local-AS
```

Check that R3 does not receive the summary prefix:

```
Rack1R3#show ip bgp 174.1.1.0
% Network not in table
```

4. IP and IOS Features

Task 4.1

```
R2:
interface FastEthernet0/0.26
 no ip proxy-arp

R6:
interface FastEthernet0/1.26
 no ip proxy-arp
```

Task 4.1 Breakdown

When a router received an ARP request on an interface that is for a client not on that network, the router will respond with its own MAC address if it has a route for the destination in the IP routing table. This behavior is known as *proxy* arp. Proxy arp is on by default on all FastEthernet interfaces. To disable this behavior, issue the **`no ip proxy-arp`** interface level command.

Task 4.1 Verification

Verify that proxy ARP is disabled:

```
Rack1R6#show ip interface Fa0/1.26 | include ARP
  Proxy ARP is disabled
  Local Proxy ARP is disabled
```

Task 4.2

```
ip wccp web-cache
!
R4:
interface Serial0/0
 ip wccp web-cache redirect out
```

Task 4.2 Breakdown

Web Cache Communication Protocol (WCCP) allows the transparent redirection of client web requests to one or more web cache engine, which can significantly improve browsing performance over low speed links. The only command needed to enable WCCP is the interface level command **`ip wccp web-cache redirect [in | out]`** command. This command instructs the router in which direction to listen for HTTP requests. In the above example, R4 listens for HTTP requests leaving the Serial0/0 interface. These requests will be redirected to one or more web cache engines that have been configured to communicate with the router via WCCP. By redirecting traffic passing out the serial interface, web traffic from VLAN 4 to VLAN 45 will not be sent to the cache engine.

Task 4.2 Verification

```
Rack1R4#show ip wccp interfaces
WCCP interface configuration:
    Serial0/0
        Output services: 1
        Input services:  0
        Mcast services:  0
        Exclude In:      FALSE
```

Task 4.3

```
R6:
ip sla monitor 1
 type echo protocol ipIcmpEcho 115.0.0.1
 request-data-size 1250
 timeout 25
 threshold 25
 frequency 30
ip sla monitor schedule 1 start-time now
```

Task 4.3 Breakdown

Cisco Service Assurance Agent (SAA), also known as the Response Time Reporter (RTR), is a monitoring application built into IOS that can track service levels through UDP echo, ICMP, DNS lookups, packet loss, etc.

To configure SAA, first issue the **`rtr [number]`** global configuration command. This creates an instance of SAA. Once inside the rtr mode, specify the type of traffic to generate by issuing the **`type`** command. In the above example ICMP echo packets will be sent to the IP address 115.0.0.1. The frequency and timeout have also been specified in the above configuration.

To initiate an SAA instance, issue the **`rtr schedule [number] start-time [time].`**

Further Reading

Cisco Service Assurance Agent User Guide

Measuring Delay, Jitter, and Packet Loss with Cisco IOS SAA and RTTMON

Task 4.3 Verification

```
Rack1R6#show ip sla monitor configuration 1
IP SLA Monitor, Infrastructure Engine-II.
Entry number: 1
Owner:
Tag:
Type of operation to perform: echo
Target address: 115.0.0.1
Request size (ARR data portion): 1250
Operation timeout (milliseconds): 25
Type Of Service parameters: 0x0
Verify data: No
Operation frequency (seconds): 30
Next Scheduled Start Time: Start Time already passed
Group Scheduled : FALSE
Life (seconds): 3600
Entry Ageout (seconds): never
Recurring (Starting Everyday): FALSE
Status of entry (SNMP RowStatus): Active
Threshold (milliseconds): 25
Number of statistic hours kept: 2
Number of statistic distribution buckets kept: 1
Statistic distribution interval (milliseconds): 20
Number of history Lives kept: 0
Number of history Buckets kept: 15
History Filter Type: None
Enhanced History
```

Verify the SLA statistics:

```
Rack1R6#show ip sla monitor statistics 1
Round trip time (RTT)   Index 1
        Latest RTT: NoConnection/Busy/Timeout
Latest operation start time: *09:17:37.899 UTC Tue May 2 2006
Latest operation return code: Timeout
Number of successes: 0
Number of failures: 6
Operation time to live: 3425 sec
```

Note that the operation times out due to the very short timeout.

Task 4.4

R2:

```
interface FastEthernet0/0.26
 standby 1 ip 174.1.26.254
 standby 1 preempt
```

```
R6:
track 1 rtr 1
!
interface FastEthernet0/1.26
 standby 1 ip 174.1.26.254
 standby 1 priority 110
 standby 1 preempt
 standby 1 track 1 decrement 20
```

Task 4.4 Verification

Verify the tracking configuration:

```
Rack1R6#show track
Track 1
  Response Time Reporter 1 state
  State is Down
    1 change, last change 00:02:53
  Latest operation return code: Timeout
  Tracked by:
    HSRP FastEthernet0/1.26 1
```

Verify the HSRP groups:

```
Rack1R6#show standby
FastEthernet0/1.26 - Group 1
  State is Standby
    1 state change, last state change 00:02:26
  Virtual IP address is 174.1.26.254
  Active virtual MAC address is 0000.0c07.ac01
    Local virtual MAC address is 0000.0c07.ac01 (v1 default)
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 0.228 secs
  Preemption enabled
  Active router is 174.1.26.2, priority 100 (expires in 7.220 sec)
  Standby router is local
  Priority 90 (configured 110)
    Track object 1 state Down decrement 20
  IP redundancy name is "hsrp-Fa0/1-1" (default)
```

Note that R6 is in standby mode since the tracking object is down.

5. IP Multicast

Task 5.1

R1:

```
ip multicast-routing
!
interface FastEthernet0/0.1001
 ip pim sparse-dense-mode
!
interface Serial0/0
 ip pim sparse-dense-mode
!
interface Serial0/1
 ip pim sparse-dense-mode
```

R2:

```
ip multicast-routing
!
interface FastEthernet0/0.26
 ip pim sparse-dense-mode
!
interface Multilink1
 ip pim sparse-dense-mode
```

R3:

```
ip multicast-routing
!
interface FastEthernet0/0
 ip pim sparse-dense-mode
!
interface Multilink1
 ip pim sparse-dense-mode
!
interface Serial1/2
 ip pim sparse-dense-mode
```

R4:

```
ip multicast-routing
!
interface FastEthernet0/0
 ip pim sparse-dense-mode
!
interface Serial0/0
 ip pim sparse-dense-mode
```

R5:

```
ip multicast-routing
!
interface Serial0/0
 ip pim sparse-dense-mode
!
interface FastEthernet0/1.52
 ip pim sparse-dense-mode
```

Task 5.1 Verification

Verify with **`show ip pim interface`** on each device.

```
Rack1R1#show ip pim int

Address          Interface                Ver/   Nbr    Query  DR     DR
                                          Mode   Count  Intvl  Prior
174.1.1.1        FastEthernet0/0.1001     v2/SD  0      30     1      174.1.1.1
174.1.145.1      Serial0/0                v2/SD  2      30     1      174.1.145.5
174.1.13.1       Serial0/1                v2/SD  1      30     1      0.0.0.0
Rack1R1#
```

```
Rack1R2#show ip pim int

Address          Interface                Ver/   Nbr    Query  DR     DR
                                          Mode   Count  Intvl  Prior
174.1.26.2       FastEthernet0/0.26       v2/SD  0      30     1      174.1.26.2
174.1.23.2       Multilink1               v2/SD  1      30     1      0.0.0.0
Rack1R2#
```

```
Rack1R3#show ip pim int

Address          Interface                Ver/   Nbr    Query  DR     DR
                                          Mode   Count  Intvl  Prior
174.1.3.3        FastEthernet0/0          v2/SD  0      30     1      174.1.3.3
174.1.23.3       Multilink1               v2/SD  1      30     1      0.0.0.0
174.1.13.3       Serial1/2                v2/SD  1      30     1      0.0.0.0
Rack1R3#
```

```
Rack1R4#show ip pim int

Address          Interface                Ver/   Nbr    Query  DR     DR
                                          Mode   Count  Intvl  Prior
174.1.4.4        FastEthernet0/0          v2/SD  0      30     1      174.1.4.4
174.1.145.4      Serial0/0                v2/SD  1      30     1      174.1.145.4
Rack1R4#
```

```
Rack1R5#show ip pim int

Address          Interface                Ver/   Nbr    Query  DR     DR
                                          Mode   Count  Intvl  Prior
174.1.145.5      Serial0/0                v2/SD  1      30     1      174.1.145.5
192.10.1.5       FastEthernet0/1.52       v2/SD  0      30     1      192.10.1.5
Rack1R5#
```

Task 5.2

R1:

```
interface Loopback0
 ip pim sparse-dense-mode
!
ip pim send-rp-announce Loopback0 scope 16 group-list EVEN interval 5
!
ip access-list standard EVEN
 permit 224.0.0.0 0.255.255.255
 permit 226.0.0.0 0.255.255.255
 permit 228.0.0.0 0.255.255.255
 permit 230.0.0.0 0.255.255.255
 permit 232.0.0.0 0.255.255.255
 permit 234.0.0.0 0.255.255.255
 permit 236.0.0.0 0.255.255.255
 permit 238.0.0.0 0.255.255.255
```

R2:

```
interface Loopback0
 ip pim sparse-dense-mode
!
ip pim send-rp-announce Loopback0 scope 16 group-list ODD interval 5
!
ip access-list standard ODD
 permit 225.0.0.0 0.255.255.255
 permit 227.0.0.0 0.255.255.255
 permit 229.0.0.0 0.255.255.255
 permit 231.0.0.0 0.255.255.255
 permit 233.0.0.0 0.255.255.255
 permit 235.0.0.0 0.255.255.255
 permit 237.0.0.0 0.255.255.255
 permit 239.0.0.0 0.255.255.255
```

R3:

```
interface Loopback0
 ip pim sparse-dense-mode
!
ip pim send-rp-discovery Loopback0 scope 16
!
ip pim rp-announce-filter rp-list R2_RP group-list R2_GROUPS
ip pim rp-announce-filter rp-list R1_RP group-list R1_GROUPS
!
ip access-list standard R1_GROUPS
 permit 224.0.0.0 0.255.255.255
 permit 226.0.0.0 0.255.255.255
 permit 228.0.0.0 0.255.255.255
 permit 230.0.0.0 0.255.255.255
 permit 232.0.0.0 0.255.255.255
 permit 234.0.0.0 0.255.255.255
 permit 236.0.0.0 0.255.255.255
 permit 238.0.0.0 0.255.255.255
!
```

```
ip access-list standard R1_RP
 permit 150.1.1.1
!
ip access-list standard R2_GROUPS
 permit 225.0.0.0 0.255.255.255
 permit 227.0.0.0 0.255.255.255
 permit 229.0.0.0 0.255.255.255
 permit 231.0.0.0 0.255.255.255
 permit 233.0.0.0 0.255.255.255
 permit 235.0.0.0 0.255.255.255
 permit 237.0.0.0 0.255.255.255
 permit 239.0.0.0 0.255.255.255
!
ip access-list standard R2_RP
 permit 150.1.2.2
```

Task 5.2 Verification

On each multicast device, verify the RP to group mappings.

R1:

```
ip mroute 150.1.3.3 255.255.255.255 174.1.13.3
```

```
Rack1R4#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/8
  RP 150.1.1.1 (?), v2v1
    Info source: 150.1.3.3 (?), elected via Auto-RP
         Uptime: 00:08:55, expires: 00:02:02
Group(s) 225.0.0.0/8
  RP 150.1.2.2 (?), v2v1
    Info source: 150.1.3.3 (?), elected via Auto-RP
         Uptime: 00:08:55, expires: 00:02:02
Group(s) 226.0.0.0/8
  RP 150.1.1.1 (?), v2v1
    Info source: 150.1.3.3 (?), elected via Auto-RP
         Uptime: 00:08:55, expires: 00:02:04
Group(s) 227.0.0.0/8
  RP 150.1.2.2 (?), v2v1
    Info source: 150.1.3.3 (?), elected via Auto-RP
         Uptime: 00:08:55, expires: 00:01:59
<output omitted>
```

You may run into RPF failures on R1, since the route to the loopback of R3 is via OSPF, not via the Serial interface to R3. A static mroute can override the routing table, for the purpose of multicast RPF checks.

Output of **`debug ip mpacket`** shown below:

```
IP(0): s=150.1.2.2 (Serial0/1) d=224.0.1.39 id=16232, ttl=14, prot=17,
len=94(90), RPF lookup failed for source
IP(0): s=150.1.2.2 (Serial0/1) d=224.0.1.39 id=16232, ttl=14, prot=17,
len=94(90), not RPF interface
Rack1R1(config)#
IP(0): s=150.1.3.3 (Serial0/1) d=224.0.1.40 id=1140, ttl=15, prot=17,
len=148(144), RPF lookup failed for source
IP(0): s=150.1.3.3 (Serial0/1) d=224.0.1.40 id=1140, ttl=15, prot=17,
len=148(144), not RPF interface
```

R1:

```
ip mroute 150.1.3.3 255.255.255.255 Serial0/1
```

After configuring the mroute, the debug output will not show errors about not being the RPF interface.

```
IP(0): s=150.1.2.2 (Serial0/1) d=224.0.1.39 (Serial0/0) id=16569,
ttl=14, prot=17, len=90(90), mforward

IP(0): s=150.1.2.2 (Serial0/1) d=224.0.1.39 id=16576, ttl=14, prot=17,
len=94(90), mroute olist null

IP(0): s=150.1.3.3 (Serial0/1) d=224.0.1.40 (Serial0/0) id=1324,
ttl=15, prot=17, len=144(144), mforward
```

Task 5.3

R1,R2,R3,R4 and R5:

```
ip pim spt-threshold 128
```

Task 5.4

R1:

```
interface Serial0/0
 ip pim nbma-mode
```

R4:

```
Interface Fa0/0
Ip igmp join-group 226.0.0.4
```

Task 5.4 Verification

To verify, ping 226.0.0.4 from R5:

```
Rack1R5#ping
Protocol [ip]:
Target IP address:  226.0.0.4
Repeat count [1]: 1000
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Interface [All]: Serial0/0
Time to live [255]:
Source address: 192.10.1.5
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 226.0.0.4, timeout is 2 seconds:
Packet sent with a source address of 192.10.1.5
....
Reply to request 56 from 174.1.145.4, 80 ms
Reply to request 57 from 174.1.145.4, 80 ms
Reply to request 58 from 174.1.145.4, 100 ms
Reply to request 59 from 174.1.145.4, 100 ms
Reply to request 60 from 174.1.145.4, 100 ms
Reply to request 61 from 174.1.145.4, 100 ms
<output omitted>
```

Verify the multicast routing table on R1. Notice that the Serial0/0 interface is listed in BOTH the incoming and outgoing interface list:

```
Rack1R1#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running,A-Candidate for MSDP Advertisement,
       U-URD,I-Received Source Specific Host Report,Z-Multicast Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 226.0.0.4), 00:00:11/00:03:25, RP 150.1.1.1, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0, 174.1.145.4, Forward/Sparse-Dense, 00:00:04/00:03:25

(192.10.1.5, 226.0.0.4), 00:00:11/00:02:55, flags: T
  Incoming interface: Serial0/0, RPF nbr 174.1.145.5
  Outgoing interface list:
    Serial0/0, 174.1.145.4, Forward/Sparse-Dense, 00:00:04/00:03:25
<output omitted>
```

Task 5.5

R1:

```
interface FastEthernet0/0.1001
 ip directed-broadcast
!
interface Serial0/1
 ip multicast helper-map 239.39.39.39 174.1.1.255 100
!
access-list 100 permit udp any any eq 3434
!
ip forward-protocol udp 3434
```

R2:

```
interface FastEthernet0/0.26
 ip multicast helper-map broadcast 239.39.39.39 100
!
access-list 100 permit udp any any eq 3434
!
ip forward-protocol udp 3434
```

Task 5.5 Breakdown

The **`ip multicast helper-map`** is used to translate UDP broadcast packets into multicast packets, or translate UDP multicast packets into broadcast packets. This type of configuration is usually required when legacy applications (typically stock ticker applications) only send out traffic to the local IP broadcast address 255.255.255.255.

In order to transport this traffic from one portion of the network to the other, the only other options would be to translate the broadcast packets to unicast packets using the ip helper-address command, or to bridge IP traffic between various network segments. Neither of these two solutions scale. Therefore, broadcast to multicast conversion can take advantage of an already in place multicast transit network (typical of trading environment), and drop the packets off at the destination as either a multicast, or reconvert the traffic back to broadcast.

In order to convert a UDP broadcast to a multicast for transit, and then back to a broadcast for final delivery, as in this scenario, the helper-map must be configured on both the ingress device and egress device for the traffic flow.

> **Note**
>
> Although the first hop device is the ingress (entry) point and the last hop device is the egress (exit) point, the helper map is configured on the incoming interface of both devices.

The above scenario describes the case where the legacy application is located on VLAN 26 attached to R2. This application is sending broadcast traffic (255.255.255.255) to UDP port 3434. The destination of this traffic is on VLAN 1001 attached to R1. Therefore, the ingress point will be R2, while the egress point is R1. The incoming interfaces of both of these devices are Fa0/0.26 on R2 and S0/1 on R1 (R2 incoming from the source and R1 incoming from the IP cloud).

The syntax for broadcast to multicast conversion is:

```
ip multicast helper-map broadcast [group] [acl]
```

The syntax for multicast to broadcast conversion is:

```
ip multicast helper-map [group] [directed_broadcast] [acl]
```

Where group is the multicast group address to translate to, acl is the access list which matches the udp port or ports to listen for, and directed_broadcast is the IP directed broadcast address of the final destination interface.

Note

In order for incoming traffic to hit the multicast helper-map it must be processed switched. In order to process switch traffic on a per port basis use the **`ip forward-protocol udp [port]`** command. In this specific scenario port 3434 must be matched on both R1 and R2.

Finally, in order to transmit the multicast to broadcast conversion, the target interface must support directed broadcast transmissions. To enable the transmission of these directed broadcast packets issue the **`ip directed-broadcast`** interface level command.

Task 5.5 Verification

Option 1:
Create an IP SLA monitor on R6:

```
R6:
ip sla monitor 2
 type udpEcho dest-ipaddr 255.255.255.255 dest-port 3434 control
disable
 timeout 1
 frequency 1
ip sla monitor schedule 2 life forever start-time now
```

Note: Some IOS versions will not allow the use of the broadcast address for the destination of a SLA monitor.

Option 2:
If your IOS version will not allow the broadcast as a destination, you can enable broadcast domain lookup on R6 and add UDP 53 to the forward protocol and access lists on R1 and R2. Make sure that you remove after testing.

```
R1 and R2:
access-list 100 permit udp any any eq 53
ip forward-protocol udp 53
```

```
R6:
Ip domain-lookup

Rack1R6#ping r1

Translating "r1"...domain server (255.255.255.255)
```

Verification for both options:

Debug multicast packets on R2:

```
Rack1R2#debug ip mpacket

 IP(0): s=174.1.26.6 (FastEthernet0/0.26) d=239.39.39.39 (Multilink1)
id=0, prot=17, len=44(44), mforward
  IP(0): s=174.1.26.6 (FastEthernet0/0) d=239.39.39.39 (Multilink1)
id=0, prot=17, len=44(44), mforward
```

Verify the directed broadcast packets on R1:

```
Rack1R1#debug ip packet detail 100

IP: tableid=0, s=174.1.26.6 (Serial0/1), d=174.1.1.255
(FastEthernet0/0), routed via RIB
IP: tableid=0, s=174.1.26.6 (Serial0/1), d=174.1.1.255
(FastEthernet0/0), routed via RIB
```

☠ Pitfall

The IOS documentation for the feature has changed over the last few major releases of IOS. The current 12.4 documentation is in sync with our solution as opposed to placing the multicast helper command on the outbound interface facing the “client”.

6. QoS

Task 6.1

R1, R4, and R5:

```
interface Serial0/0
 frame-relay traffic-shaping
 frame-relay class FRTS
!
map-class frame-relay FRTS
 frame-relay cir 128000
 frame-relay bc 16000
```

Task 6.1 Breakdown

This is a fairly straightforward configuration, configure a map class for CIR and apply to the interface. Since we are not given a specific time interval or bc, it doesn’t matter what value you use.

Task 6.1 Verification

Verify the shaping configuration on every router:

```
Rack1R1#show frame-relay pvc 104

PVC Statistics for interface Serial0/0 (Frame Relay DTE)

DLCI = 104, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0

  input pkts 149           output pkts 302          in bytes 8838
  out bytes 24176          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 69        out bcast bytes 3604
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 00:14:13, last time pvc status changed 00:14:13
  cir 128000    bc 16000     be 0       byte limit 2000   interval 125
  mincir 64000     byte increment 2000  Adaptive Shaping none
  pkts 10        bytes 781       pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/40, 0 drop, 0 dequeued
```

Note per-VC queueing strategy and queue depth.

Task 6.2

```
R1:
map-class frame-relay FRTS
 frame-relay holdq 400
```

Task 6.2 Breakdown

The Frame Relay hold queue is the amount of buffers that are dedicated to the traffic shaping queue. When the hold queue is full, all other packets attempting to enter the output queue are dropped.

Task 6.2 Verification

Verify new queueing parameters:

```
Rack1R1#show frame-relay pvc 104 | beg Queueing
  Queueing strategy: fifo
  Output queue 0/400, 0 drop, 0 dequeued
```

Task 6.3

```
R1:
 class-map match-all AUDIO
  match access-group name AUDIO
!
 policy-map LLQ
  class AUDIO
   priority 128 8000
!
interface Serial0/1
 bandwidth 1536
 service-policy output LLQ
!
ip access-list extended AUDIO
 permit udp any any eq 7070

R3:
 class-map match-all AUDIO
  match access-group name AUDIO
!
 policy-map LLQ
  class AUDIO
   priority 128 8000
!
interface Serial1/2
 bandwidth 1536
 service-policy output LLQ
!
ip access-list extended AUDIO
 permit udp any any eq 7070
```

Task 6.3 Verification

Verify the policy-map configuration (note the burst size is in bytes, not bits):

```
Rack1R1#show policy-map interface s0/1
 Serial0/1

  Service-policy output: LLQ

    Class-map: AUDIO (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name AUDIO
      Queueing
        Strict Priority
        Output Queue: Conversation 264
        Bandwidth 128 (kbps) Burst 8000 (Bytes)
        (pkts matched/bytes matched) 0/0
        (total drops/bytes drops) 0/0

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Task 6.4

```
R4:
policy-map WRED
  class class-default
   fair-queue
   random-detect
   random-detect precedence 5 60 90 5
!
map-class frame-relay FRTS
 service-policy output WRED
```

Task 6.4 Breakdown

The above task dictates how to configure Weighted Random Early Detection (WRED) inside the Modular QoS CLI. In order to configure WRED inside the MQC, the class in question must either have a **bandwidth** reservation configured, or be the default class with **fair-queue** enabled. The min and max levels determine the queue depths that will have packets discarded. The drop probability denominator gives the chance of the packet getting dropped by the router.

Task 6.4 Verification

Verify the WRED configuration:

```
Rack1R4#show frame-relay pvc 401 | begin service policy
  service policy WRED
 Serial0/0: DLCI 401 -

  Service-policy output: WRED

    Class-map: class-default (match-any)
      42 packets, 2388 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
        Flow Based Fair Queueing
        Maximum Number of Hashed Queues 16
        (total queued/total drops/no-buffer drops) 0/0/0
         exponential weight: 9

  class    Transmitted   Random drop  Tail drop Minimum Maximum  Mark
           pkts/bytes    pkts/bytes   pkts/bytes thresh  thresh  prob
      0    0/0           0/0          0/0            20      40   1/10
      1    0/0           0/0          0/0            22      40   1/10
      2    0/0           0/0          0/0            24      40   1/10
      3    0/0           0/0          0/0            26      40   1/10
      4    0/0           0/0          0/0            28      40   1/10
      5    0/0           0/0          0/0            60      90   1/5
      6    43/2420       0/0          0/0            32      40   1/10
      7    0/0           0/0          0/0            34      40   1/10
   rsvp    0/0           0/0          0/0            36      40   1/10

  Output queue size 0/max total 400/drops 0
```

7. Security

Task 7.1

```
R5:
no ip source-route
!
interface FastEthernet0/1.52
 ip access-group APPEASE_MANAGER in
 no ip proxy-arp
 no cdp enable
!
interface FastEthernet0/1.53
 ip access-group APPEASE_MANAGER in
 no ip proxy-arp
 no cdp enable
```

```
!
ip access-list extended APPEASE_MANAGER
 deny   tcp any 174.1.0.0 0.0.255.255 eq www
 deny   tcp any 174.1.0.0 0.0.255.255 eq telnet
 deny   tcp any eq www 174.1.0.0 0.0.255.255
 deny   tcp any eq telnet 174.1.0.0 0.0.255.255
 deny   tcp any eq www 150.1.0.0 0.0.255.255
 deny   tcp any eq telnet 150.1.0.0 0.0.255.255
 deny   tcp any 150.1.0.0 0.0.255.255 eq www
 deny   tcp any 150.1.0.0 0.0.255.255 eq telnet
 deny   icmp any any echo
 permit ip any any
```

Task 7.1 Verification

Verify that CDP is disabled on required interfaces:

```
Rack1R5#show cdp interface
FastEthernet0/0 is up, line protocol is up
  Encapsulation ARPA
  Sending CDP packets every 60 seconds
  Holdtime is 180 seconds
FastEthernet0/1 is up, line protocol is up
  Encapsulation 802.1Q Virtual LAN, Vlan ID  1.
  Sending CDP packets every 60 seconds
  Holdtime is 180 seconds
Serial0/1 is up, line protocol is up
  Encapsulation HDLC
  Sending CDP packets every 60 seconds
  Holdtime is 180 seconds
```

Note that the subinterfaces for VLANs 52 and 53 are not listed. Next verify that proxy ARP is disabled:

```
Rack1R5#show ip interface Fa0/1.52 | include ARP
  Proxy ARP is disabled
  Local Proxy ARP is disabled
```

Verify that source-routing is disabled:

```
Rack1R5#show running-config | include source-route
no ip source-route
```

Finally, verify the access list:

```
Rack1R5#show ip access-lists APPEASE_MANAGER
Extended IP access list APPEASE_MANAGER
    10 deny tcp any 174.1.0.0 0.0.255.255 eq www
    20 deny tcp any 174.1.0.0 0.0.255.255 eq telnet
    30 deny tcp any 150.1.0.0 0.0.255.255 eq www
    40 deny tcp any 150.1.0.0 0.0.255.255 eq telnet
    50 deny tcp any eq www 174.1.0.0 0.0.255.255
    60 deny tcp any eq telnet 174.1.0.0 0.0.255.255
    70 deny tcp any eq www 150.1.0.0 0.0.255.255
    80 deny tcp any eq telnet 150.1.0.0 0.0.255.255
    90 deny icmp any any echo
    100 permit ip any any (41 matches)
```

Task 7.2

R5:

```
 class-map match-all FROM_BB3
  match input-interface FastEthernet0/1
 class-map match-all FROM_BB2
  match input-interface FastEthernet0/1
!
 policy-map TO_BB2
  class FROM_BB3
   drop
 policy-map TO_BB3
  class FROM_BB2
   drop
!
interface FastEthernet0/1.52
 service-policy output TO_BB2
!
interface FastEthernet0/1.53
 service-policy output TO_BB3
```

Task 7.2 Breakdown

The above configuration stops traffic from transiting between two interfaces. By matching the input interface for traffic and dropping it, traffic cannot come in FastEthernet 0/1.52 and go out FastEthernet 0/1.53, or vice versa.

Task 7.2 Verification

To verify traffic filtering, temporarily remove the access-group from FastEthernet0/1.52, and announce network 204.12.1.0/24 to BB2 via RIP:

R5:

```
router rip
 redistribute connected
```

Next do a ping from BB2 to 204.12.1.254:

```
BB2>ping 204.12.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 204.12.1.254, timeout is 2 seconds:
..
```

And verify the policy-map on interface FastEthernet 0/1.53:

```
Rack1R5#show policy-map interface Fa0/1.53
 FastEthernet0/1.53

  Service-policy output: TO_BB3

    Class-map: FROM_BB2 (match-all)
      8 packets, 944 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: input-interface FastEthernet0/1
      drop

    Class-map: class-default (match-any)
      1 packets, 77 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Task 7.3

```
R5:
class-map match-all FROM_BB3
  match not access-group name TO_SMTP_SERVER
 class-map match-all FROM_BB2
  match not access-group name FROM_SMTP_SERVER
!
ip access-list extended FROM_SMTP_SERVER
 permit tcp host 192.10.1.100 eq smtp 204.12.1.0 0.0.0.255
ip access-list extended TO_SMTP_SERVER
 permit tcp 204.12.1.0 0.0.0.255 host 192.10.1.100 eq smtp
```

Task 7.3 Breakdown

The above configuration adds an exception to the previous task. Traffic is only dropped when it is moving between FastEthernet0/1.52 and FastEthernet0/1.53 if it is not a traffic flow from the above referenced SMTP server and its clients in VLAN 53.

Task 7.3 Verification

Verify the access-lists:

```
Rack1R5#show ip access-lists
<output omitted>

Extended IP access list FROM_SMTP_SERVER
    10 permit tcp host 192.10.1.100 eq smtp 204.12.1.0 0.0.0.255
Extended IP access list TO_SMTP_SERVER
    10 permit tcp 204.12.1.0 0.0.0.255 host 192.10.1.100 eq smtp
```

Verify the modified class-maps:

```
Rack1R5#show class-map
 Class Map match-any class-default (id 0)
   Match any

 Class Map match-all FROM_BB3 (id 1)
   Match input-interface FastEthernet0/1
   Match not access-group name TO_SMTP_SERVER

 Class Map match-all FROM_BB2 (id 2)
   Match input-interface FastEthernet0/1
   Match not access-group name FROM_SMTP_SERVER
```