Lab Guide

Overview

This guide presents the instructions and other information concerning the activities for this course. You can find the recommended solutions in the Answer Key.

Outline

This guide includes these activities:

- Lab 4-1: Establishing Basic Connectivity for BGP
- Lab 4-2: Configuring BGP
- Lab 4-3: Filtering BGP Updates and Path Determination
- Lab 5-1: Establishing Basic Connectivity for MPLS Layer 3 VPNs
- Lab 5-2: Configuring the MPLS Core
- Lab 5-3: Creating VPNs and Enabling VPN Routing
- Lab 5-4: Adding a Backup Link in VPNA
- Lab 6-1: Establishing Basic Connectivity for Multicast
- Lab 6-2: Configuring Dense Mode IP Multicast Routing
- Lab 6-3: Configuring PIM-SM
- Lab 7-1: Classification and Marking
- Lab 7-2: Class-Based Shaper
- Lab 7-3: Class-Based Policer
- Lab 7-4: Avoiding and Managing Congestion
- Lab 8-1: Implementing NAT
- Lab 8-2: Implementing FHRP
- Lab 8-3: Implementing NTP
- Lab 8-4: Implementing DHCP for IPv4 and IPv6

Note Your instructor will provide you with CIERSASSESS-1 during Module 2 and CIERSASSESS-2 during Module 9.

Lab 4-1: Establishing Basic Connectivity for BGP

Complete this lab activity to provide the underlying basic connectivity that is needed for the advanced BGP scenarios in this module.

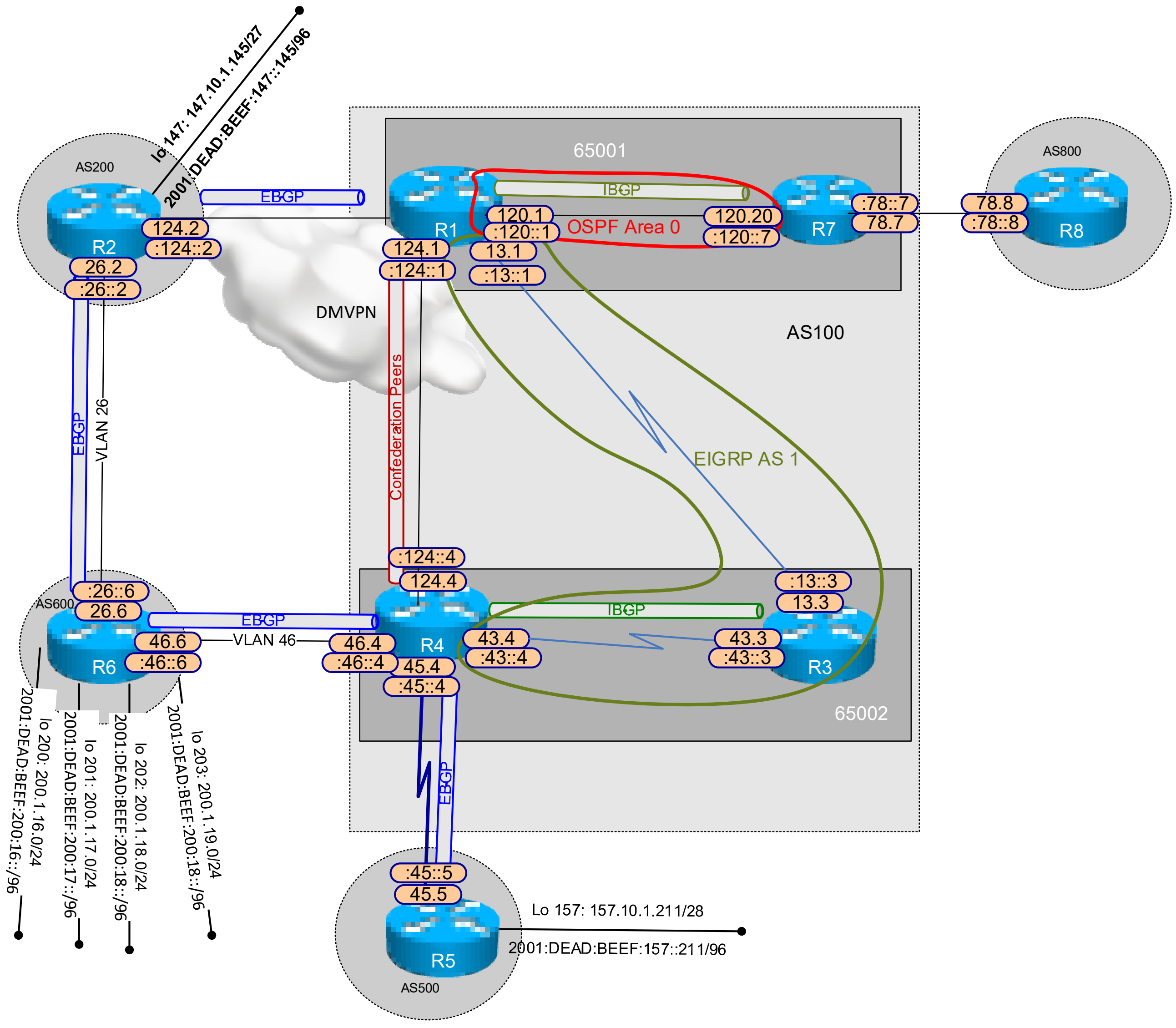
Activity Objective

This activity provides the opportunity to exercise setting up DMVPN, switched Ethernet, and back-to-back serial connections, which are required to support BGP. You will also configure OSPF and EIGRP to enable BGP peering, next-hop reachability, and synchronization requirements. Where the IPv4 addresses are unspecified, they are /24 subnets of 172.16.0.0. Where the IPv6 addresses are unspecified, they are 2005:DEAD:BEEF::/80. After completing this activity, you will be able to meet this objective:

- Establish the baseline Layer 2 and Layer 3 topologies to support BGP

Visual Objective

The figure illustrates what you will accomplish in this activity.



Note This figure supports all three BGP labs and not all Loopback interfaces are shown.

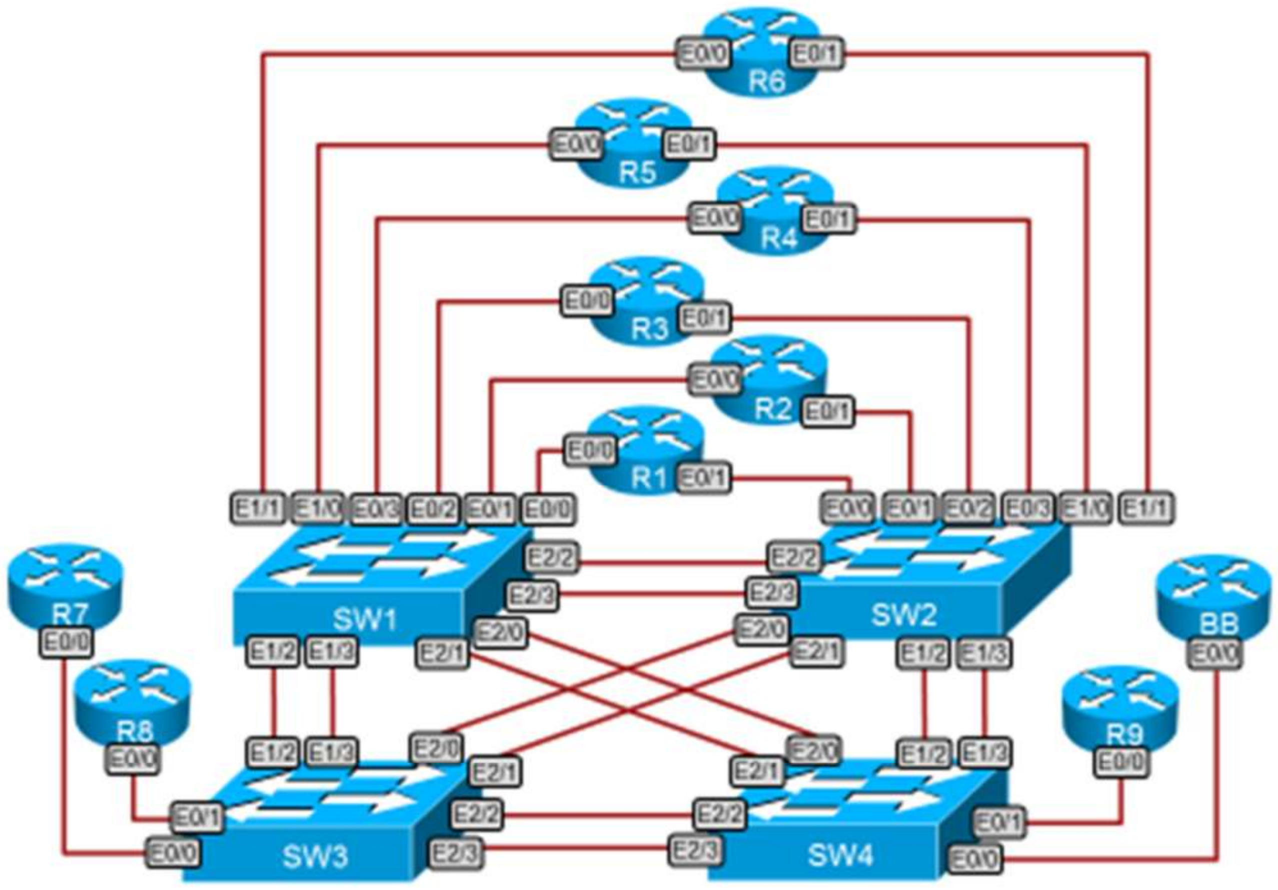
Useful Commands

This section lists some potentially useful commands.

Protocol	Common Commands
DMVPN	**ip nhrp map** *ip-address nbma-address* **ip nhrp map multicast** *nbma-address* **ip nhrp network-id** *number* **ip nhrp nhs** *nhs-address* [*net-address [netmask]*] **show ip nhrp** **show frame-relay map**
Ethernet switching	**switchport trunk encapsulation** {**isl** | **dot1q**} **switchport mode** {**trunk** | **access**} **switchport access vlan** *vlan-id* **no switchport** **show interface trunk** **show vlan** **show interface status**

Task 1: Configure Ethernet Switching

In this task, you will implement the Ethernet switching configuration between R1, R2, R4, R6, R7, and R8 that will be used in the BGP topology.



Activity Procedure

Complete these steps:

Step 1 Create the necessary VLANs, trunks, and access.

Router	Switch	VLAN
R1	SW1	17, 124
R2	SW1	26, 124
R4	SW1	124
R4	SW2	46
R6	SW1	26, 46
R7	SW3	17, 78
R8	SW3	78

Step 2 Use VTP transparent mode.

Step 3 Where trunking is required, use the IEEE standard trunking protocol. Use access mode on the first link between SW1 and SW2.

Step 4 Trunks should only allow the nessessary VLANs.

Note The specific interface might vary depending on the training partner.

Task 2: Configure the Serial Connections

In this task, you will configure the serial connections between R1, R3, R4, and R5 that will be used in the BGP topology.

Activity Procedure

Complete these steps:

Step 1 Configure a back-to-back serial link between R1 and R3 using the default encapsulation for subnet 172.16.13.0/24 and prefix 2005:DEAD:BEEF:13::/80.

Step 2 Configure a back-to-back serial link between R4 and R5 for subnet 172.16.45.0/24. and prefix 2005:DEAD:BEEF:45::/80.

Step 3 Configure a Layer 2 authentication between R4 and R5 with a password of C1sc0. Make sure the password is not sent.

Step 4 Configure the back-to-back serial interfaces between R3 and R4 using the default encapsulation for subnet 172.16.43.0/24 and prefix 2005:DEAD:BEEF:43::/80.

Task 3: Configure the DMVPN Connections

In this task, you will configure the DMVPN connections between R1, R2, and R4 that will be used in the BGP topology.

Activity Procedure

Complete these steps:

Step 1 Configure DMVPN connections as shown in the Visual Objective diagram between R1, R2, and R4. All IPv4 and IPv6 addresses shown on the diagram should be associated with the tunnel interface of the routers.

Step 2 Configure the physical or subinterface IPv4 addressing area as follows:

i. **R1:** 10.0.0.1/24

ii. **R2:** 10.0.0.2/24

iii. **R4:** 10.0.0.4/24.

Step 3 Once you have verified the configuration, shut down the tunnel interface on R4

Task 4: Implement the Required IGPs

In this task, you will implement the specific IGP configuration that will be used in the BGP topology.

Activity Procedure

Complete these steps:

Step 1 Configure OSPF and EIGRP as the Visual Objective diagram shows for both IPv4 and IPv6.

Step 2 OSPF must have *only* subnet 172.16.120.0/24 and 2005:DEAD:BEEF:120::/80 as internal routes.

Step 3 EIGRP includes *only* subnets 172.16.13.0/24 and 172.16.43.0/24 for IPv4, and 2005:DEAD:BEEF:13::/80 and 2005:DEAD:BEEF:43::/80 for IPv6.

Step 4 Do not redistribute EIGRP or connected routes into OSPF.

Step 5 Verify that R1 can reach 172.16.43.4 and 2005:DEAD:BEEF:43::4 and R4 can reach 172.16.13.1 and 2005:DEAD:BEEF:13::1.

Step 6 On R4, shut down the tunnel interface that is associated with the IP address 172.16.124.4.

Note Shutting down this interface is critical to the lab operations.

Lab 4-2: Configuring BGP

Complete this lab activity to establish a base BGP configuration with synchronization. The lab allows you to practice configuration skills as well as expert-level task analysis skills for advanced BGP topics.

Activity Objective

Configure BGP as shown on the Visual Objective diagram for Lab 4-1. After completing this activity, you will be able to meet these objectives:

- Create the necessary BGP autonomous systems and peer them as is required.
- Advertise networks into BGP using network statements.

Useful Commands

This section lists some potentially useful commands.

Routing Protocol	Common Commands
BGP configuration	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number* **neighbor** *ip-address* | *peer-group-name* **next-hop-self** **network** *network-number* [**mask** *network-mask*] **bgp confederation identifier** *as-number* **bgp confederation peers** *as-number* [... *as-number*] **synchronization**
BGP verification	**show ip bgp summary** **show bgp ipv6 unicast summary** **show ip bgp** **show bgp ipv6 unicast** **debug bgp** **show ip bgp** [*ip-address*] **show bgp ipv6 unicast** [*ipv6-address*]

Task 1: Implement BGP Confederations

In this task, you will enable BGP on R1, R3, R4, R7, and R8.

Activity Procedure

Complete these steps:

Step 1 Create AS 100 as a confederation of AS 65001 and AS 65002.

Step 2 Place R1 and R7 in AS 65001.

Step 3 Place R3 and R4 in AS 65002.

Step 4 Configure peering between AS 65001 and AS 65002 only between R1 and R4. Use addresses 172.16.13.1/2005:DEAD:BEEF:13::1 and 172.16.43.4/2005:DEAD:BEEF:43::4.

Task 2: Implement EBGP Neighbors to AS 100

In this task, you will enable BGP on R2, R5, and R8. You will peer these routers with AS 100, and advertise routes.

Activity Procedure

Complete these steps:

Step 1 Configure AS 200 on R2 and advertise Loopback 147 into BGP using a network statement.

Step 2 Peer R2 and R1.

Step 3 Configure AS 500 on R5 and advertise Loopback 157 into BGP using a network statement.

Step 4 Peer R5 and R4.

Step 5 Configure AS 800 on R8.

Step 6 Peer R8 and R7.

Step 7 Verify that all BGP speakers have prefixes from R2 Loopback 147 and R5 Loopback 157 in their routing tables from BGP. Make sure that you have reachability between these subnets using an extended **ping** with a **source** option (a source ping) and an extended **traceroute** with a **source** option (a source trace) from R5.

Note You will enable BGP on R6 in Lab 4-3.

Task 3: Use the Synchronization Method in AS 65001

In this task, you will use the synchronization method in AS 65001.

Activity Procedure

Complete these steps:

Step 1 Enable synchronization on R1 and R7.

Step 2 Implement a solution that will ensure that both subnets 147.10.1.128/27 and 157.10.1.208/28 have best indicators in the local RIBs of both routers.

Step 3 Confirm that R8 still has the routes within its RIB.

Lab 4-3: Filtering BGP Updates and Path Determination

Complete this lab activity to establish BGP filters and to influence BGP path determination. The lab allows you to practice configuration skills as well as expert-level task analysis skills for BGP filters and path determination.

Activity Objective

Use BGP filtering and path determination features to implement the required policies for the autonomous systems in the lab.

Useful Commands

This section lists some potentially useful commands.

Routing Protocol	Common Commands
BGP configuration	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number* **neighbor** *ip-address* | *peer-group-name* **next-hop-self** **clear ip bgp *** **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
BGP verification	**show ip bgp** **show ip bgp** [*ip-address*]

Task 1: Configure and Connect AS 600

In this task, you will configure AS 600 on R6 and connect it to its EBGP peers R2 and R4.

Activity Procedure

Complete these steps:

Step 1 Configure AS 600 on R6 and peer it as shown in the Lab 4-1 Visual Objective diagram.

Step 2 Originate only these networks from AS 600 without using network statements or any static routes.

- Loopback200: 200.1.16.0/24 and 2001:DEAD:BEEF:200:16::/96
- Loopback201: 200.1.17.0/24 and 2001:DEAD:BEEF:200:17::/96
- Loopback202: 200.1.18.0/24 and 2001:DEAD:BEEF:200:18::/96
- Loopback203: 200.1.19.0/24 and 2001:DEAD:BEEF:200:19::/96

Step 3 Make AS 600 nontransit for any potential prefixes, and do not use any prefix filtering.

Task 2A: Filter Updates

In this task, you will filter updates to AS 500 from AS 100.

Activity Procedure

Complete these steps:

Step 1 Allow R4 to announce only the 200.1.16.0/24 and 2001:DEAD:BEEF:200:16::/96 prefixes and the 200.1.17.0/24 and 2001:DEAD:BEEF:200:17::/96 prefixes to AS 500. Use a minimum number of statements to perform this task.

Step 2 Allow router R4 to announce routes that are originating from AS 200 as well.

Task 2B: Filter Updates

In this task, you will filter updates from AS 800 to AS 100.

Activity Procedure

Complete these steps:

Step 1 Add the following loopback interfaces to R8:

- Loopback 100: 208.1.16.8/24
- Loopback 101: 208.1.17.8/24
- Loopback 102: 208.1.18.8/24
- Loopback 103: 208.1.19.8/24

Step 2 Inject the new loopback interfaces on R8 into BGP with an origin code of "i".

Step 3 On R7, enable soft reconfiguration inbound relative to R8.

Step 4 Have R8 only send 208.1.16.0/24 and 208.1.17.0/24, but no filtering is allowed on R8. Verify that only those two routes are displayed on R7 for the received routes from R8.

Task 3: Implement Preferred Paths

In this task, you will implement preferred paths in the lab topology.

Activity Procedure

Complete this step:

Step 1 Make sure that router R4 selects R1 for forwarding outbound traffic to prefixes that originate from AS 600.

Task Analysis

Your Cisco CCIE® lab strategy should include task analysis. As you read and review the lab tasks, consider these questions:

1. Do you fully understand the requirements?
   - What is the synchronization method? Do you need to research it before starting? Should you skip it for now?
   - How does the underlying Layer 3 connectivity interact with BGP?
   - Why are source pings specified? Does this make your task easier or harder?
   - Why would you want to add dynamic routes to OSPF? Which routes?
   - Can you break down this task into manageable subproblems, such as these?
     - Peering
     - Route advertisement
     - Optimization
2. Which options are specifically ruled out?
   - You are not permitted to add additional connected routes into OSPF.
   - No static routes, no additional routing protocols, no policy-based routing, and no tunnels are allowed.
   - Are your general practice approaches ruled out?
3. How does the context affect configuration?
   - BGP requires underlying Layer 3 connectivity for peering and next-hop reachability. Do you have it? If not, what are the workaround options?
   - Do the link types or topology present special problems for peering or packet delivery?
   - Is there a linkage to a subsequent lab task? Do other lab tasks depend on this one being performed in a particular way?

Approaches for Common Issues

Here are suggested approaches for addressing common challenges with this lab:

1. Loopback 147 is not being advertised from R2 to R1. (Or Loopback 157 is not being advertised from R5 to R3.)
   - Is the loopback network in the local RIB of the originating router?
     - Run the **show ip bgp** command. The prefix should be in the table with a next hop of 0.0.0.0.
     - Use the **show run | begin router bgp** command to verify your network statement. Is it an exact match for a routing table entry?

— Do you have a good peering?

  - Use the **show ip bgp summary** command to look at the status value at the end of the configured neighbor. Active or idle would indicate a bad peering.
  - Can you **ping** the peering address?
  - Run the **debug ip bgp** command, and then the **clear ip bgp *** command. Do the peers go to the established state? If not, what errors are shown?
  - Verify your configuration for typos.

2. Loopback 147 is seen on R4, but not on R5. (Or Loopback 157 is seen on R1, but not on R2.)

   — Have you verified the peerings?

   — Are the routes eligible for advertisement? Use the **show ip bgp** command on the sending router. Is there a best indicator?

   — Are the routes eligible for advertisement? On R1, run the **show ip bgp 157.10.1.208** command. On R3, run the **show ip bgp 147.101.128** command.

     - Does the next hop show "inaccessible"? If so, what configuration options are available to fix this?
     - Does the output say "not synchronized"?
       - Try disabling synchronization to see if that fixes it.
       - Fix synchronization by making sure that you have a matching route from a source other than BGP.
       - Do the BGP and OSPF source routers have the same RID?

   — Are there any filters that are blocking the routes?

3. You have two good routes on all of the BGP routers, but you cannot ping.

   — Did you remember to do a source ping? Why is this necessary here?

   — Try a source trace. Where does it fail?

   — Does R3 have both loopback routes in its forwarding table? Why is this necessary in this context?

   — Are there forwarding plane filters in place?

Lab 5-1: Establishing Basic Connectivity for MPLS Layer 3 VPNs

Complete this lab activity to practice the underlying basic connectivity needed for the MPLS scenarios in this module.

Activity Objective

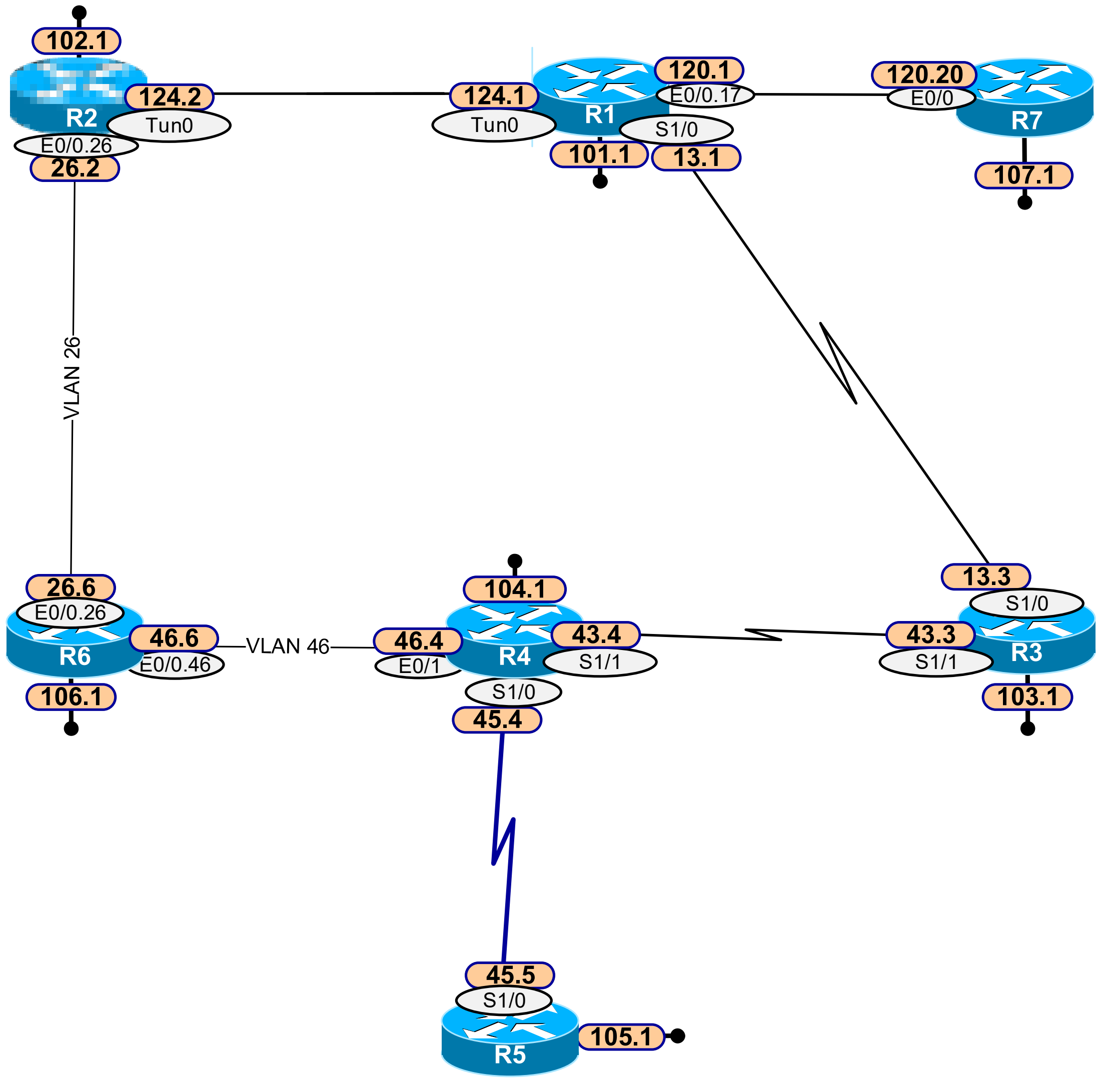
This activity assumes that the basic Layer 2 and Layer 3 tasks in the previous BGP labs have been completed. No changes to the Ethernet, DMVPN, or IPv4 addressing configuration are required. You will remove any currently configured routing protocols.

Where unspecified, the IPv4 addresses are /24 subnets of 172.16.0.0. After completing this activity, you will be able to meet this objective:

- Establish the baseline Layer 2 and Layer 3 topologies to support MPLS Layer 3 VPN scenarios

Visual Objective

The figure illustrates what you will accomplish in this activity.



Note Only the interfaces, links, and IP addresses that are relevant to this lab are shown.

Task 1: Prepare the Equipment for the Lab

In this task, you will remove any currently configured routing protocols and verify the required starting topology.

Activity Procedure

Complete these steps:

Step 1 Remove any currently configured routing protocols from the devices shown in the Visual Objective diagram. This can be quickly accomplished by entering the commands **no ip routing** and then **ip routing** in global configuration mode.

Step 2 Make sure that the tunnel interface on R4 is shut down.

Step 3 Verify same-subnet reachability on each of the links shown in the diagram.

Note Route maps, access lists, prefix lists, and other configurations do not need to be removed.

Lab 5-2: Configuring the MPLS Core

Complete this lab activity to establish OSPF, BGP, and MPLS in the core of the network.

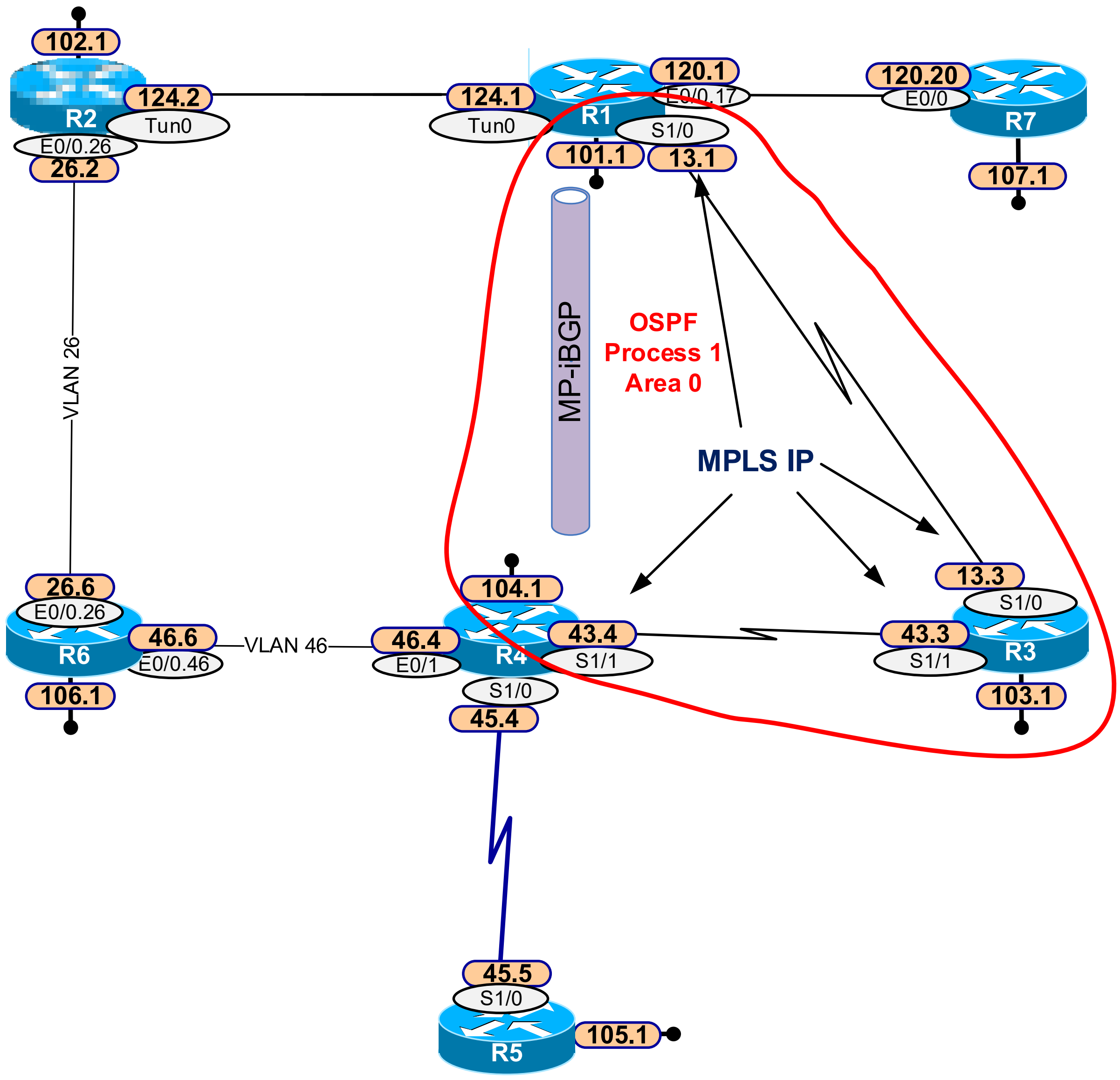
Activity Objectives

Configure OSPF, BGP, and LDP as shown in the Visual Objective diagram. After completing this activity, you will be able to meet these objectives:

- Enable OSPF in the core
- Create an MP-BGP peering between PE routers
- Enable MPLS on core links
- Disable TTL propagation

Visual Objective

The figure illustrates what you will accomplish in Lab 5-2.



Useful Commands

The table describes the commands that are used in this activity.

Technology	Common Commands
BGP	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number* **neighbor** {**ip-address** | **ipv6-address** [%] | **peer-group-name**} **update-source interface-type interface-number** **address-family vpnv4** **neighbor** {**ip-address** | **peer-group-name** | **ipv6-address**} **activate** **neighbor** {**ip-address** | **ipv6-address** | **peer-group-name**} **send-community** [**both** | **standard** | **extended**] **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
MPLS	**mpls ldp router-id** [**vrf** *vrf-name*] **interface** [**force**] **mpls ip** **show mpls ldp neighbor** **show mpls forwarding-table** **debug mpls packets**

Task 1: Enable OSPF in the Core

In this task, you will enable OSPF to support BGP and LDP in the core.

Activity Procedure

Complete these steps:

Step 1 On R1, add only the subnets 172.16.101.0/24 and 172.16.13.0/24 into OSPF process ID 1 Area 0.

Step 2 On R3, add only subnets 172.16.103.0/24, 172.16.13.0/24, and 172.16.43.0/24 into OSPF process ID 1 Area 0.

Step 3 On R4, add only the subnets 172.16.104.0/24 and 172.16.43.0/24 into OSPF process ID 1 Area 0.

Step 4 Make sure R1, R3 and R4 advertise their loopbacks interfaces with their configured mask.

Step 5 Verify the configuration and reachability among the five networks.

Task 2: Implement MP-BGP Between R1 and R4

In this task, you will enable MP-BGP on PE routers R1 and R4.

Activity Procedure

Complete these steps:

Step 1 Configure an MP-IBGP peer relationship between R1 loopback 101 and R4 loopback 104. Use AS number 1.

Step 2 Activate the neighbors for the VPNv4 address family and enable each peer to send extended communities to the other peers.

Note Use **show ip bgp neighbor** to verify that the VPNv4 address family is enabled.

Task 3: Enable MPLS in the Core

In this task, you will enable LDP on the links connecting R3 to R1 and R4 and examine its operation.

Activity Procedure

Complete these steps:

Step 1 Configure loopbacks 101, 103, and 104 to serve as LDP router IDs.

Step 2 Enable LDP on the links that connect R3 to R1 and R4.

Step 3 Verify that R3 has two LDP neighbors.

Step 4 Disable TTL propagation.

Step 5 On R1, enter the command **show mpls forwarding-table**. What label does R1 use when sending traffic to 172.16.104.1?

________________________.

Step 6 Enter the same **show** command on R4. What label does R4 use to send traffic to 172.16.101.1?

________________________.

Step 7 Enable **debug mpls packets** on R3. Ping from R4 to 172.16.101.1 using a source of loopback 104. Was the traffic routed by R3 or was it label-switched?

________________________.

Step 8 Allow the debug to run for a few minutes. Every 60 seconds, packets marked CoS 6 are switched by R3. What is this traffic? Disable the debug.

________________________.

Lab 5-3: Creating VPNs and Enabling VPN Routing

Complete this lab activity to practice MPLS Layer 3 VPN configuration and analysis skills.

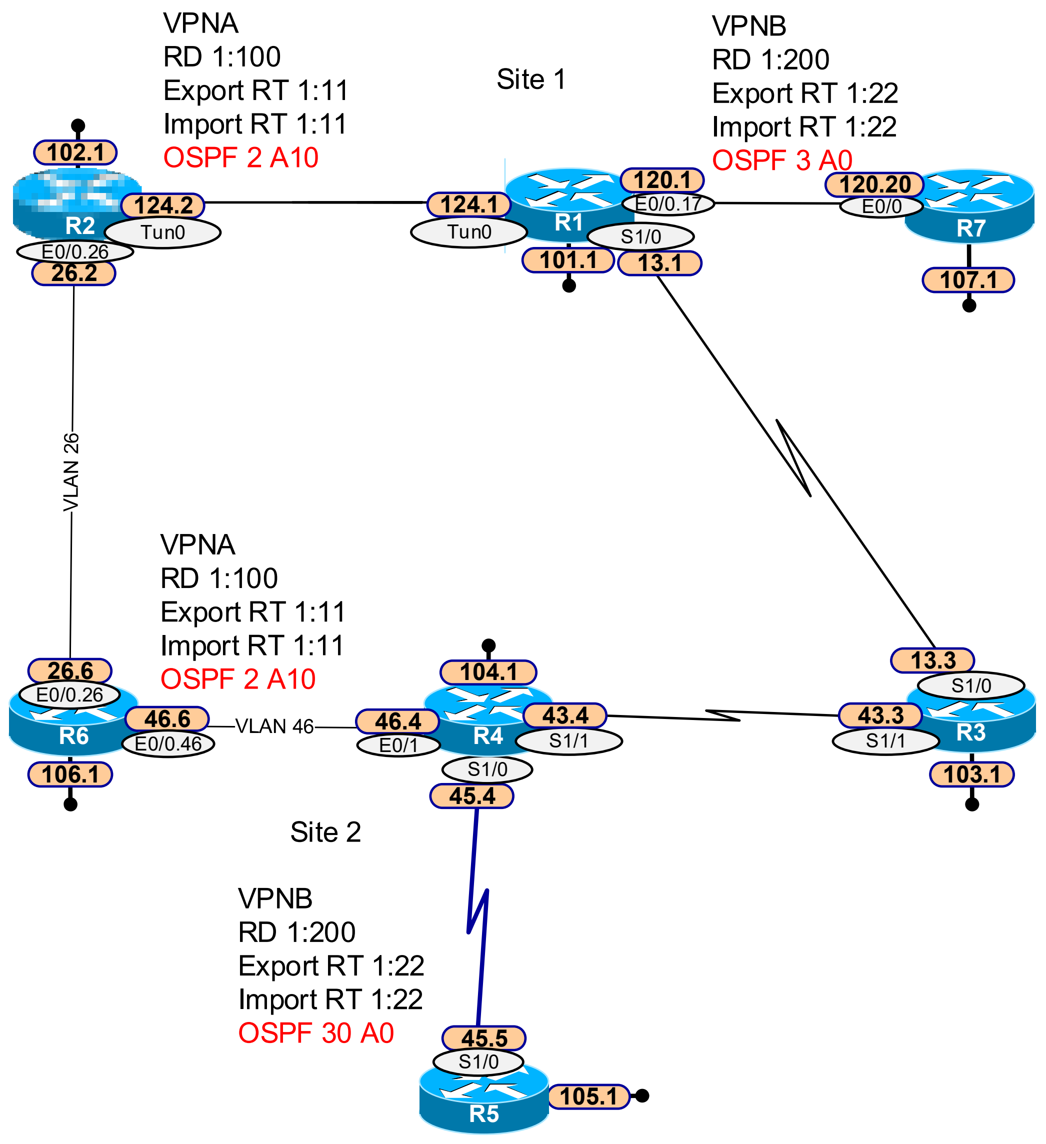
Activity Objectives

In this activity, you will create customer VRFs on PE routers R1 and R4. After completing this activity, you will be able to meet these objectives:

- Create VRFs on the PE routers
- Enable OSPF for PE-to-CE customer routing at each site
- Redistribute OSPF and MP-BGP to achieve full customer reachability across the MPLS connection

Visual Objective

The figure illustrates what you will accomplish in Tasks 1 and 2 of Lab 5-3.



Useful Commands

The table describes the commands that are used in this activity.

Technology	Common Commands
MPLS L3 VPNs	**ip vrf** *vrf-name* **ip vrf forwarding** *vrf-name* **rd route-distinguisher** **route-target** {**import** | **export** | **both**} **route-target-ext-community** **show ip vrf** [**brief** | **detail** | **interfaces** | **id**] [*vrf-name*] [**output-modifiers**]
OSPF and BGP	**address-family ipv4 vrf** *vrf-name* **redistribute match internal match external** **show ip bgp vpnv4** {**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*} [**network-address** [**mask**] **router ospf process-id** [**vrf** *vpn-name*] **show ip route** [**vrf** *vpn-name*]

Task 1: Create VRFs for VPNA and VPNB on the PE Routers

In this task, you will create the necessary VRFs on R1 and R4 and associate the required connected interfaces. Use the Visual Objective diagram as an aid.

Activity Procedure

Complete these steps:

Step 1 On R1, create a VRF called "VPNA" using the following settings:

- Route distinguisher value 1:100
- Export route target 1:11
- Import route target 1:11
- Include the tunnel interface in this VRF.

Step 2 On R1, create a VRF called "VPNB" using the following settings:

- Route distinguisher value 1:200
- Export route target 1:22
- Import route target 1:22
- Include the subinterface to R7 in this VRF.

Step 3 On R4, create a VRF called "VPNA" using the following settings:

- Route distinguisher value 1:100
- Export route target 1:11
- Import route target 1:11
- Include interface E0/1 in this VRF.

Step 4 On R4, create a VRF called "VPNB" using the following settings:

- Route distinguisher value 1:200
- Export route target 1:22
- Import route target 1:22
- Include interface S1/0 in this VRF.

Step 5 Use the command **show ip vrf** to verify the correct configurations.

Task 2: Enable OSPF for PE-to-CE Routing

In this task, you will enable OSPF between the PE routers R1 and R4 and the CE routers, R2, , R5, R6, and R7.

Activity Procedure

Complete these steps:

Step 1 On R1, create OSPF process 2 for VRF VPNA. Configure a unique router ID and add a network statement for subnet 172.16.124.0/24 in Area 10. Configure the tunnel interface as an OSPF point-to-point network type.

Step 2 Create an OSPF process on R2 and add *only* the subnets 172.16.102.0/24 and 172.16.124.0/24 into Area 10. *Do not include subnet 172.16.26.0/24 at this time.*

Step 3 On R1, create OSPF process 3 for VRF VPNB. Configure a unique router ID and add a network statement for subnet 172.16.120.0/24 in Area 0.

Step 4 Create an OSPF process on R7 and add subnets 172.16.120.0/24 and 172.16.107.0/24 into Area 0.

Step 5 On R4, create OSPF process 2 for VRF VPNA. Configure a unique router ID and add a network statement for subnet 172.16.46.0/24 in Area 10.

Step 6 Create an OSPF process on R6 and add *only* the subnets 172.16.106.0/24 and 172.16.46.0/24 into Area 10. *Do not include subnet 172.16.26.0/24 at this time.*

Step 7 On R4, create an OSPF process 30 for VRF VPNB. *Note that this is a different process ID than was used for this VPN on R1.* Configure a unique router ID and add a network statement for subnet 172.16.45.0/24 in Area 0.

Step 8 Create an OSPF process on R5 and add subnets 172.16.105.0/24 and 172.16.45.0/24 into Area 0.

Step 9 Use the command **show ip ospf interface brief** on R1 and R4 to verify your configuration.

Task 3: Mutually Redistribute OSPF and MP-BGP

In this task, you will use MP-BGP to convey OSPF routing information between sites. This will achieve full unicast reachability within each VPN.

Activity Procedure

Complete these steps:

Step 1 On R1 and R4, redistribute BGP 1 into both VPN OSPF processes.

Step 2 On R1 and R4, redistribute the VPN OSPF processes into the correct BGP VRF address family.

Step 3 Verify successful redistribution into BGP with the command **show ip bgp vpnv4 all**. You should see routes from the local site in the BGP table for each VPN.

Step 4 Verify routing in VPNA. Check that the routing table on R2 includes OSPF interarea routes to subnets 172.16.106.0/24 and 172.16.46.0/24. Ping and trace to 172.16.106.1. Are the core links visible in the output?

Step 5 In the space below, enter the OSPF cost from R2 to subnet 172.16.106.0/24.

______________________________.

Step 6 Verify routing in VPNB. Check that the routing table on R7 contains OSPF E2 routes for 172.16.105.0/24 and 172.16.45.0/24. Why are these routes external when the routes in VPNA were interarea routes?

____.

Step 7 On R1, enter the command **show ip route vrf VPNB**. Is the route to subnet 172.16.45.0/24 a BGP route or an OSPF route? Which routing source has a lower administrative distance?

____.

Step 8 On R1, enter the command **show ip ospf database external**. You should see a Type 5 LSA for subnet 172.16.45.0/24. Why did OSPF fail to put this route in the forwarding table?

____.

Step 9 Configure a domain ID so that VPNB routes from other sites are interarea routes in the VRF forwarding table.

Lab 5-4: Adding a Backup Link in VPNA

Complete this lab activity to practice configuration of an OSPF sham link.

Activity Objective

In this activity, you will add a backup link in VPNA. After completing this activity, you will be able to meet these objectives:

- Add a backup link
- Optimize paths between sites using a sham link

Useful Commands

The table describes the commands that are used in this activity.

Routing Protocol	Common Commands
OSPF	**area** *area-id* **sham-link** *source-address destination-address*

Task 1: Add Subnet 172.16.26.0/24 to OSPF

In this task, you will activate the backup link in VPNA and observe the result.

Activity Procedure

Complete these steps:

Step 1 On R2 and R6, add subnet 172.16.26.0/24 into OSPF area 10 and verify an OSPF adjacency between R2 and R6.

Step 2 Change the OSPF cost to 1500 on each interface in subnet 172.16.26.0/24.

Step 3 Change the OSPF cost of the tunnel interface to be 1.

Step 4 How have the routes in VPNA changed? What is the OSPF cost from R2 to subnet 172.16.106.0/24? How has it changed?

____.

Task 2: Create a Sham Link Between PE Routers R1 and R4

In this task, you will configure loopbacks in the VPNA VRF with host routes and a sham link in OSPF Area 10 between them.

Activity Procedure

Complete these steps:

Step 1 On R1, add Loopback 0 to the VPNA VRF. Redistribute it as a connected route into BGP.

Step 2 On R4, add Loopback 0 to the VPNA VRF. Redistribute it as a connected route into BGP.

Step 3 Create a sham link between these two loopback addresses for OSPF Area 10. Verify your configuration with the commands **show ip ospf sham-link** and **show ip ospf neighbor**.

Step 4 How has this action affected the routes in VPNA? What is the OSPF cost between R2 and subnet 172.16.106.0/24?

Lab 6-1: Establishing Basic Connectivity for Multicast

Complete this lab activity to provide the underlying basic connectivity that is needed for the IP multicast scenario. This activity provides the opportunity to exercise setting up DMVPN, switched Ethernet, and OSPF to support multicast.

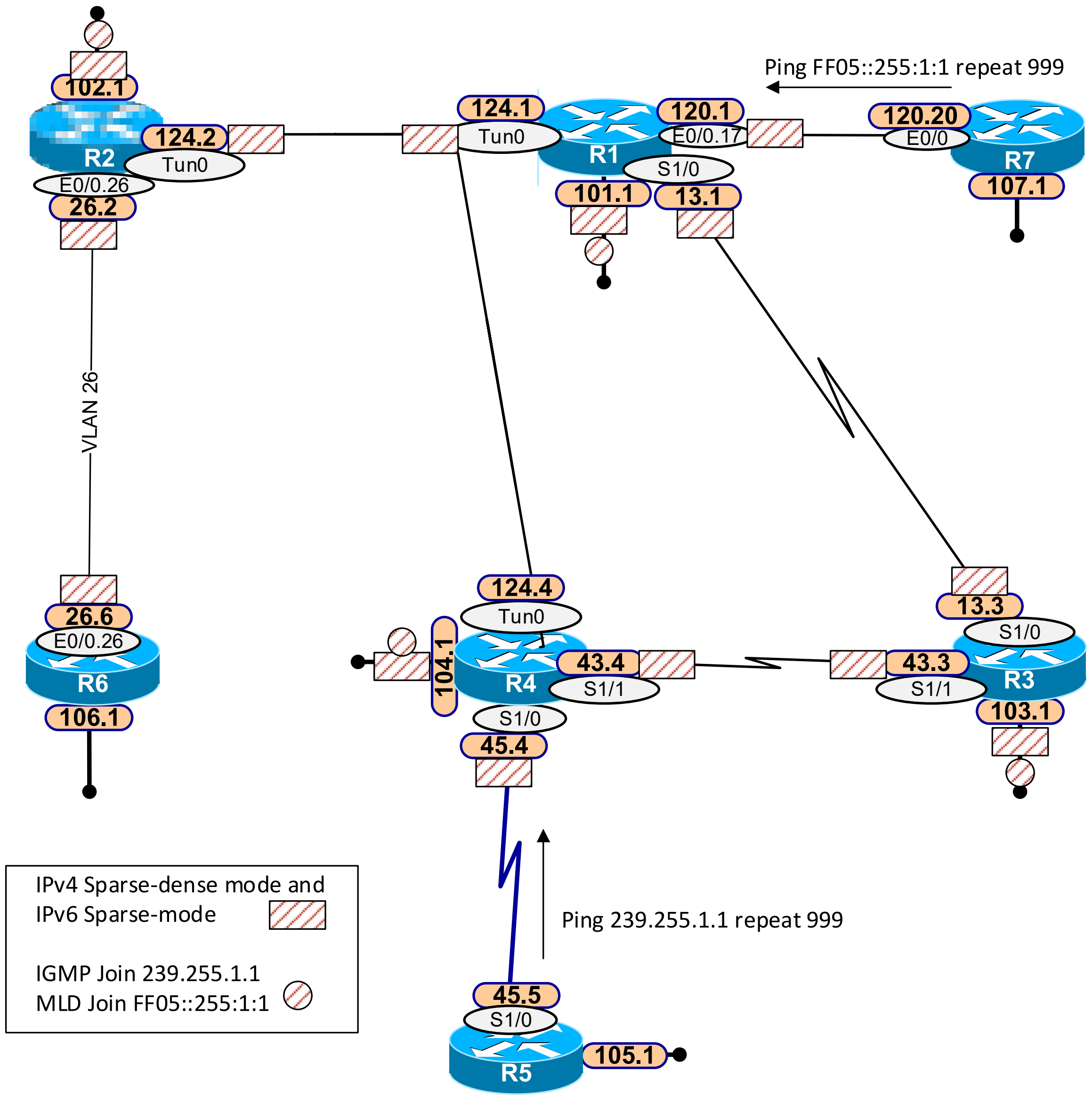
Activity Objective

This lab builds on the previous MPLS lab to develop a base for deploying IPv4 and IPv6 multicast. After completing this activity, you will be able to meet this objective:

- Establish baseline Layer 2 and Layer 3 topology to support IP multicast

Visual Objective

The figure illustrates what you will accomplish in this activity.



Task 1: Activate the Required Links

In this task, you will establish the active links that are used in the base topology.

Activity Procedure

Complete these steps:

Step 1 On R4, shut down the interface that connects to R6.

Step 2 On R4, activate the tunnel interface.

Task 2: Remove the MPLS Configuration and Implement the Required IGP

In this task, you will remove the MPLS configuration and migrate the lab network topology to OSPF only.

Activity Procedure

Complete these steps:

Step 1 Remove the VPNA and VPNB VRFs from both R1 and R4. Re-enter the IPv4 addresses on the R1 tunnel interface and the R4 interfaces E0/1 and S1/0. Remove the command **mpls ip** from the interfaces connecting R1 and R4 to R3.

Step 2 Disable IP routing and then re-enable it.

Step 3 On R1, R2, R3, R4, R5, R6, and R7, enable OSPF Area 0 for all interfaces for both IPv4 and IPv6.

- **Bonus challenge:** Configure one OSPF process for both IPv4 and IPv6.

Step 4 Use the point-to-multipoint network type on the DMVPN tunnel interfaces and set the OSPF cost to 1 on the R1 tunnel interface.

Step 5 Add the following IPv6 addresses:

- R1 Loopback 101: 2001:DEAD:BEEF:101::1/128
- R2 Loopback 102: 2001:DEAD:BEEF:102::1/128
- R3 Loopback 103: 2001:DEAD:BEEF:103::1/128
- R4 Loopback 104: 2001:DEAD:BEEF:104::1/128
- R5 Loopback 105: 2001:DEAD:BEEF:105::1/128
- R6 Loopback 106: 2001:DEAD:BEEF:106::1/128
- R7 Loopback 107: 2001:DEAD:BEEF:107::1/128

Activity Verification

You have completed this task when you verify that all addresses are reachable from all of the devices in the diagram.

Lab 6-2: Configuring Dense Mode IPv4 Multicast Routing

Complete this lab activity to establish IP multicast connectivity. This lab allows you to practice configuration skills and expert-level task analysis skills for PIM-DM topics.

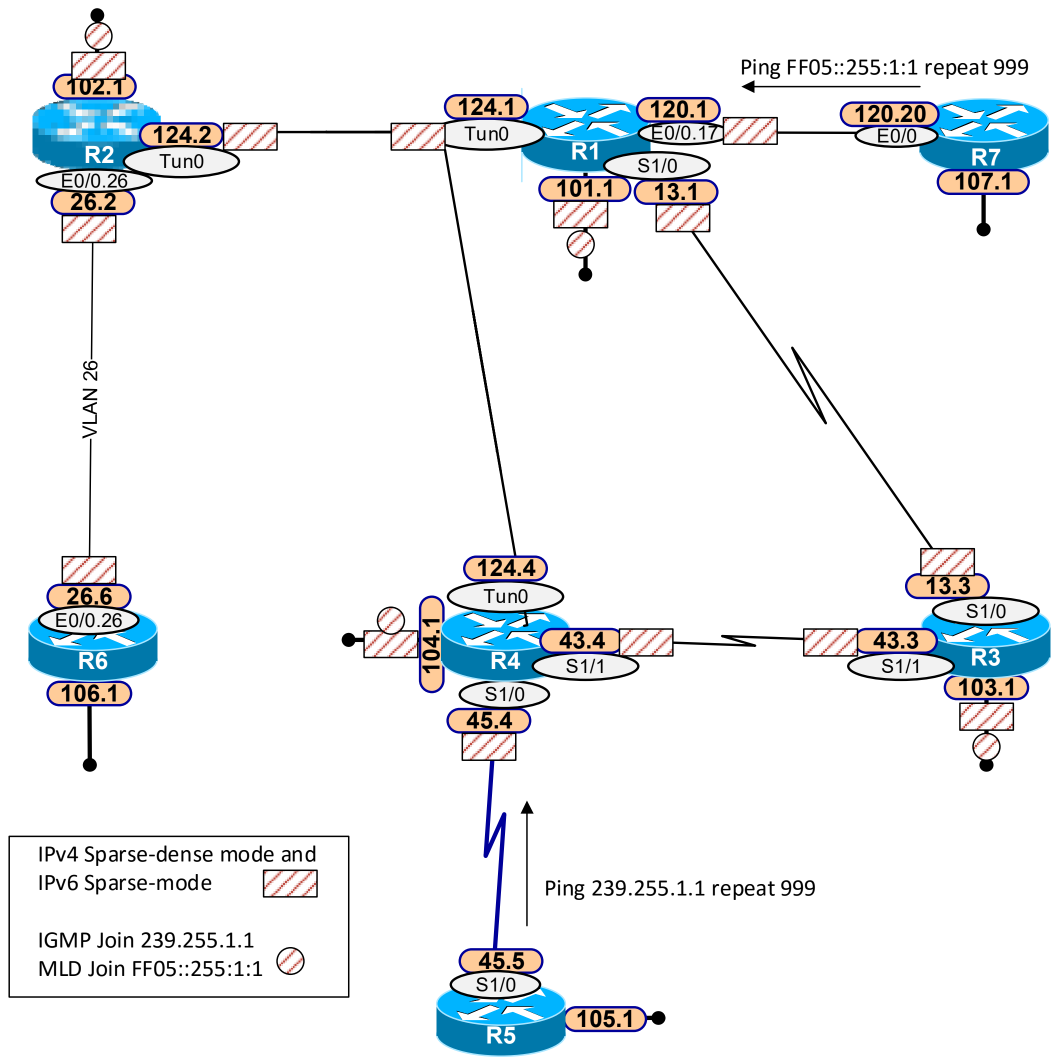
Activity Objective

Configure IP multicast as shown in the Visual Objective diagram. After completing this activity, you will be able to meet these objectives:

- Configure and troubleshoot PIM-DM
- Demonstrate successful multicast traffic flows across a network topology
- Enhance PIM-DM pruning operations in a network

Visual Objective

The figure illustrates what you will accomplish in this activity.



You will enable IPv4 multicast routing on all routers except for R5 and R7. You will enable the PIM sparse-dense mode on R1, R2, R3, R4, and R6 as shown.

Useful Commands

This section lists some potentially useful commands.

Command	Description
ip multicast-routing	In global configuration mode, enables IP multicast routing
ip pim sparse-dense-mode	In interface configuration mode, enables PIM to operate in sparse or dense mode, depending on the group
ip igmp join-group *group-address* [**source** *source-address*]	In interface configuration mode, enables a router to join a multicast group
ip mroute *source-address mask* {*rpf-address* | *interface-type interface-number*} [*distance*]	In global configuration mode, configures an IP multicast static route
ip pim state-refresh origination-interval [*interval*]	In interface configuration mode, configures the origination of the PIM Dense Mode State Refresh control message. Optionally, you can configure the number of seconds between control messages by using the **interval** argument. The default interval is 60 seconds. The interval range is from 4 to 100 seconds.
show ip pim neighbors	Lists the PIM neighbors that are discovered by the router
show ip mroute	Displays the contents of the IP multicast routing table
show ip mroute static	Displays the static routes in the IP multicast routing table
show ip pim interface count	Displays packet count information about interfaces that are configured for PIM
debug ip pim	Displays PIM packets that are received and transmitted, as well as PIM related event
debug ip mpacket	Displays IP multicast packets that are received and transmitted
mtrace source [destination] [group]	Traces the path from a source to a destination branch for a multicast distribution tree for a given group

Task 1: Configure PIM Sparse-Dense Mode

In this task, you will enable IPv4 multicast routing at the global and interface levels.

Activity Procedure

Complete these steps:

Step 1 Enter the command **ip multicast-routing** in global configuration mode on R1, R2, R3, R4, and R6.

Step 2 On the interfaces that are designated by the block symbol on the diagram, enter the command **ip pim sparse-dense-mode**. Note that these interfaces include the specified loopback interfaces but do not include the tunnel interface on R4.

Activity Verification

Like OSPF, PIM automatically discovers neighbors. Verify the basic IP multicast configuration with the **show ip pim neighbor** command. R1 should see R2 and R3 as PIM neighbors. The only neighbor for R4 is R3, because PIM is not supposed to be configured on the 172.16.124.4 interface of R4.

Note It is necessary to enable PIM on the R4 interface that is connected to R5 so that R4 will accept IP multicast from R5, which is acting as the multicast server.

Task 2: Configure and Verify Multicast Clients

In this task, you will configure IP multicast routing clients and verify connectvity.

Activity Procedure

Complete these steps:

Step 1 Configure the designated loopback interfaces on R1, R2, R3, and R4 to join the group 239.255.1.1.

Step 2 From R5, start an extended **ping** to address 239.255.1.1 with a repeat count of 999. Source the ping from the IP address 172.16.45.5.

Caution At this point, it is likely that only R3 and R4 will respond to the extended ping. You will probably need to investigate the state of the source tree on R3 and R1.

Step 3 While the ping is running, execute these commands on R3 and then on R1:

- **show ip mroute 239.255.1.1**
- **debug ip mpacket**
- **show ip pim interface count**
- **mtrace 172.16.45.5**

Step 4 Use the output from the commands above to answer these questions:

A) Is R3 forwarding the traffic to R1? ____________.
What is your evidence?
____.

____.

B) Is the traffic arriving at R1? __________.
If so, on what interface? _____________.

How do you know?
____

____.

C) What is the RPF interface on R1? ____________.

Why is R1 not responding to the multicast pings from R5?

____.

____

Step 5 Repair the source tree so that R1, R2, R3, and R4 all respond to the multicast ping from R5. Your solution must not affect unicast routing.

Task 3: Enhance Dense Mode Pruning

In this task, you will examine and enhance dense mode pruning operations in the lab network.

Activity Procedure

Complete these steps:

Step 1 On R2, examine the (S,G) state for group 239.255.1.1 using the **show ip mroute 239.255.1.1** command. What is the state of the interface that is connected to R6?

____

____

Step 2 On R6, enable the **debug ip pim** and **debug ip mpacket** commands. Examine the (S,G) entry for 239.255.1.1 using the **show ip mroute | include 172.16.45.5** command several times, observing the expire timer, and answer these questions:

A) What are the maximum and minimum values of the expire timer?

____

____

B) What happens when the expire timer reaches its minimum value?

____

____

C) What event appears to cause the expire timer to be refreshed?

____

____

Step 3 Enhance dense mode pruning by refreshing the state every 30 seconds without flooding unnecessary user traffic. Verify proper operation by using debugging and observing timer values on R6.

Note If the instructor stopped between the dense mode and sparse mode lectures, stop here and wait to complete Lab 6-3 until after the spare mode lecture. If the instructor has completed the entire multicast lecture, you can continue into the next section of these labs.

Lab 6-3: Configuring PIM-SM

Complete this lab activity to practice your skills in configuring, analyzing, and troubleshooting PIM-SM. This lab allows you to practice configuration skills and expert-level task analysis skills for PIM-SM topics.

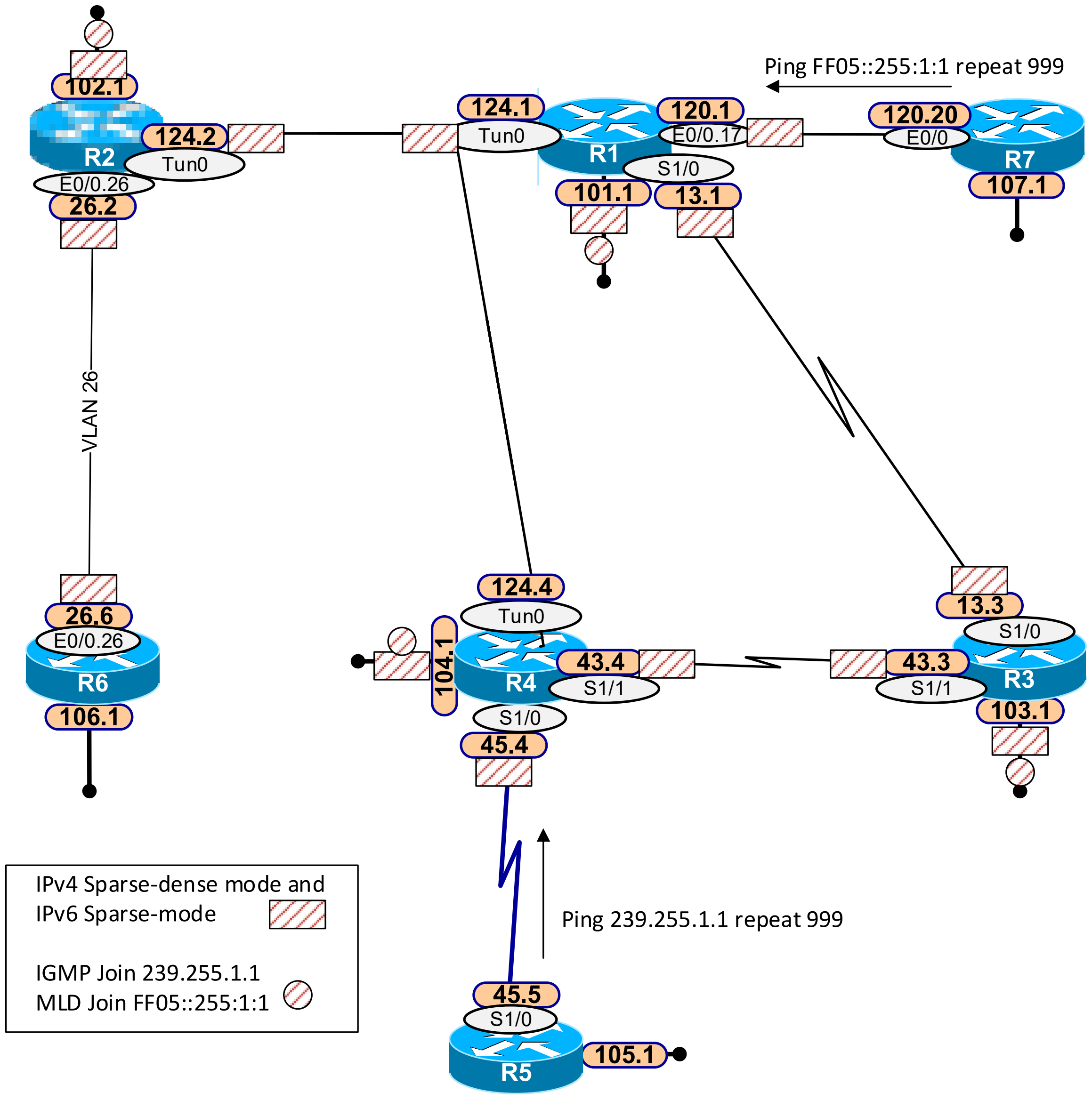
Activity Objective

Configure IPv4 and IPv6 multicast as shown in the Visual Objective diagram. After completing this activity, you will be able to meet these objectives:

- Configure and troubleshoot IP multicast performance
- Demonstrate successful multicast traffic flows across a network topology

Visual Objective

The figure illustrates what you will accomplish in this activity.



Useful Commands

This section lists some potentially useful commands.

Command	Description
ip pim [**vrf** *vrf-name*] **rp-candidate** *interface-type interface-number* **[bidir]** [**group-list** *access-list*] [**interval** *seconds*] [**priority** *value*]	Configures a router to advertise itself to the BSR as a PIMv2 candidate RP
ip pim [**vrf** *vrf-name*] **bsr-candidate** *interface-type interface-number* [*hash-mask-length* [*priority*]]	In global configuration mode, configures a router to announce its candidacy as a BSR
no ip pim dm-fallback	In global configuration mode, prevents PIM-DM fallback and blocks all multicast traffic for groups that are not specifically configured
ip pim rp-address *ip-address*	Configures the address of a PIM RP for multicast groups
ip pim nbma-mode	In interface configuration mode, configures a multiaccess WAN interface to be in NBMA mode
ip pim spt-threshold infinity	In global configuration mode, configures when a PIM leaf router should join the shortest path source tree for the specified group
show ip pim rp [**mapping**]	Displays the mappings for the PIM group to the active RP
show ip rpf *source-address*	Displays RPF information for a multicast group
debug ip pim [**group**]	Displays PIM packets that are received and transmitted, as well as PIM-related events
debug ip mpacket [**detail**]	Displays IP multicast packets that are received and transmitted
ipv6 multicast-routing [**vrf** *vrf-name*]	Enables IPv6 multicast routing
ipv6 mld join-group [*group-address*] [**include** | **exclude**] {*source-address* | **source-list** *acl*}	In interface configuration mode, configures MLD reporting for a specified group and source
ipv6 pim [**vrf** *vrf-name*] **bsr candidate rp** *ipv6-address* [**group-list** *access-list-name*] [**priority** *priority-value*] [**interval** *seconds*] [**scope** *scope-value*] [**bidir**]	In global configuration mode, configures the candidate RP to send PIM RP advertisements to the BSR
ipv6 pim [**vrf** *vrf-name*] **bsr candidate bsr** *ipv6-address [hash-mask-length]* [**priority** *priority-value*] [**scope**] [**accept-rp-candidate** *acl-name*]	In global configuration mode, configures a device to be a candidate BSR

Task 1: Enable IPv6 Multicast

In this task, you will enable IPv6 multicast and add MLD joins on R1, R2, R3, and R4.

Activity Procedure

Complete these steps:

Step 1 Enable IPv6 multicast routing on R1, R2, R3, and R4.

Step 2 Configure R1, R2, R3, and R4 to join the multicast group of FF05::255:1:1 on Loopback 10*X*, where *X* is the router number.

Task 2: Configure and Troubleshoot BSR

In this task, you will configure, analyze, and troubleshoot PIM-SM with Auto-RP.

Activity Procedure

Complete these steps:

Step 1 Remove any **state-refresh** or **static mroute** commands that were entered in previous tasks on R1, R2, R3, R4, and R6.

Step 2 Configure R1 as a candidate RP and BSR using Loopback 101.

Note R1 should act as an RP only for groups that are in the range 239.255.0.0/16 and FF05::255:0:0/96.

Step 3 Use the **show ip pim rp mapping** command to determine which routers have learned the RP information. If any of the multicast routers have not learned the RP information, investigate the problem using the commands **debug ip mpacket** and **mtrace 172.16.101.1**. Briefly describe your findings.

____

____

____

Step 4 On R5, enter the command **ping 239.255.1.1 repeat 999 source 172.16.45.5**. If there are responses, determine whether the traffic is being distributed in dense mode or sparse mode. Configure the routers so that they will not use dense mode distribution for user mode traffic when an RP is not dynamically learned.

Note As needed, refer to the answer key for detailed instructions and explanations.

Step 5 On R4, enter the **ip pim sparse-dense mode** command on the interface that is associated with the IP address 172.16.124.4. Verify that routers R1, R3, and R4 are responding to the multicast ping from R5. Verify they are all using the dynamically learned RP address 172.16.101.1.

Step 6 On R7, enter the command **ping ff05::255:1:1 source e0/0.17 repeat 999**. At this point, you should receive responses from R1, R2, R3, and R4.

Task 3: Enable Spoke-to-Spoke Multicast

In this task, you will enable spoke-to-spoke multicast for the lab topology.

Activity Procedure

Complete these steps:

Step 1 On R1, implement a method that will permit spoke-to-spoke IP multicast without adding additional interfaces.

Step 2 Verify that R2 now responds to the multicast pings from R5.

Task 4: Analyze and Control SPT Cutover

In this task, you will analyze and control the SPT cutover for PIM-SM with the lab topology.

Activity Procedure

Complete these steps:

Note You may find it helpful to configure R3 to send logging information to the buffer on R3. Use the commands **no logging console** and **logging buffered 100000** in the global configuration mode of R3. Use the commands **clear logging** and **show logging** as necessary.

Step 1 As needed, stop the multicast ping from R5 using the **Ctrl-Shift-6** key sequence. On R3, clear the mroute table using the **clear ip mroute *** command, and enter the command **debug ip pim**. On R5, enter the **ping 239.255.1.1 repeat 4 source 172.16.45.5** command. Use the ping and debug output to answer these questions.

A) Which interface or interfaces does R3 use to respond to the multicast ping?

____

____

B) On R3, examine the debug entries that are associated with group 239.255.1.1. Can you identify a PIM join message that was sent to 172.16.43.4?

____

C) Can you identify a PIM prune message that was sent to 172.16.13.1 with the RPT-bit set?

____

Step 2 On R3, implement a technique that will have R3 remain on the shared tree even when the source tree is available, and clear the mroute table.

Step 3 On R5, enter the **ping 239.255.1.1 repeat 4 source 172.16.45.5** command. Does R3 respond? If so, which interface does it use?

____

____

Step 4 On R3, enter the command **show ip mroute 239.255.1.1**. Do you have (S,G) state? If not, why not?

____

____

Module 7: Router MQC QoS

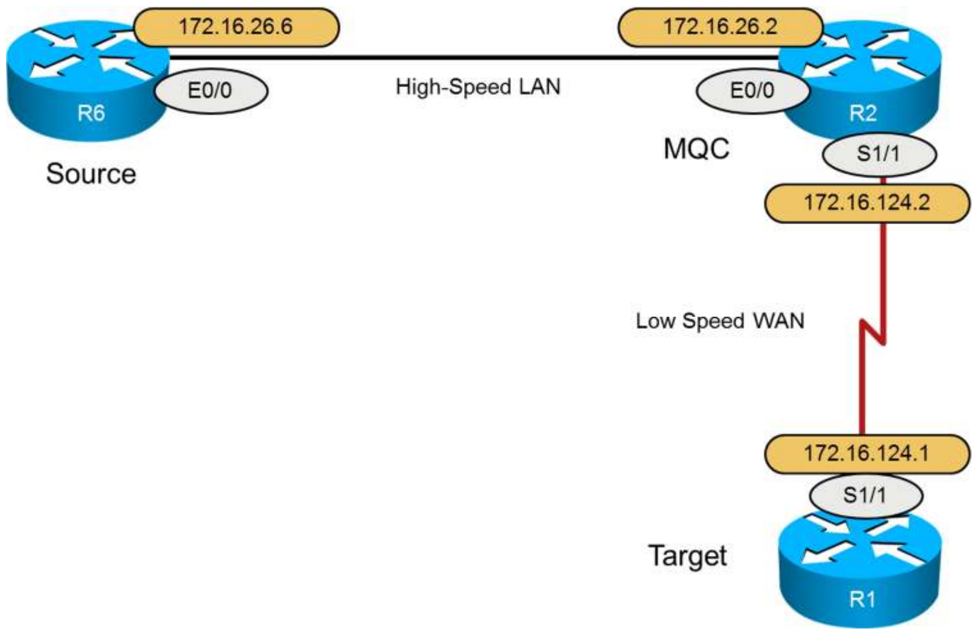
The "Router MQC QoS Task Analysis and Configuration" module of *Cisco Internetworking Expert Routing and Switching Part 1* uses a set of lab activities to reinforce the concepts that are covered in the module.

Activity Objective

These activities help learners to exercise their MQC configuration and verification skills.

Visual Objective

A simple three-router topology is used to demonstrate MQC operations in the "Router MQC QoS Task Analysis and Configuration" module.



The topology includes one Ethernet link and one serial link. Since the Ethernet link is a significantly higher bandwidth link than the serial interface, traffic will originate on the higher-speed link and then be forwarded over the lower-speed link. This is an ideal topology for applying QoS traffic shaping, congestion management, and congestion avoidance mechanisms.

To match the visual objectives, perform the following steps (this assumes completion of the multicast labs):

Step 1 Shut down the R1 S1/0 and tunnel interfaces.

Step 2 Apply IPv4 and IPv6 addresses from the tunnel interface to R1 S1/1 (you will get an error about an overlapping IPv6 address with the tunnel interface, but it still should accept the command), and then enable R1 S1/1.

Step 3 Shut down the R2 tunnel interface and apply the IPv4 and IPv6 addresses from the tunnel interface to S1/1. Then enable S1/1.

Step 4 Enable OSPF on S1/1 of R1 and R2.

Step 5 Shut down the R6 E0/0.46 interface.

Useful Commands

The table describes the commands that are used in this activity.

Command	Description
class-map *class-map-name*	Specifies a class map name.
policy-map *policy-map-name*	Specifies a policy map name
service-policy {input | output} *policy-map-name*	Attaches a traffic policy to an interface, and allows you to specify the direction in which the traffic policy should be applied
show policy-map interface	Displays the statistics and the configurations of the input and output policies that are attached to an interface
show policy-map	Displays the configuration of all classes for a specified policy map or all classes for all existing policy maps

Lab 7-1: Classification and Marking

In this activity, you will examine the operation of the MQC as a tool for classification and marking. You will set up four IP SLA sessions on R6 in order to generate different types of traffic for you to base your QoS policies on. The rest of the QoS depends on these IP SLA sessions.

Activity Procedure

Complete these steps:

Step 1 On R6, configure IP SLA for the following sessions:

- Session 1: UDP echo to port 3000 using the minimum frequency and the R1 IPv4 address as a target.
- Session 2: TCP-connect traffic to port 2999 using the minimum frequency and the R1 IPv6 address as a target.
- Session 3: ICMP echo to the R1 IPv6 address
- Session 4. ICMP jitter to the R1 IPv4 address

Step 2 Configure R1 as a responder.

Step 3 On R6 E0/0, create a policy to match on the following:

- IPv4 UDP port 3000: AF41
- IPv6 TCP port 2999: AF42
- ICMPv6: AF43
- ICMPv4: EF

Step 4 Enable all the IP SLA sessions to start now and live forever.

Step 5 Verify your policy.

Lab 7-2: Class-Based Shaper

In this activity, you will examine the operation of the MQC class-based shaper. You will configure a nested policy to shape the interface and shape specific types of traffic.

Activity Procedure

Complete these steps:

Step 1 On R2, configure a policy on ingress from R6 to match on the markings made in Lab 7-1. Set the QoS groups as follows:

- EF: Group 1
- AF41: Group 2
- AF42: Group 3
- AF43: Group 4

Step 2 On the R2 egress interface to R1, create a policy to shape all traffic to 768 kbps.

Step 3 Make sure the Tc is 10 seconds.

Step 4 Create a policy that will shape all traffic from Group 2 to 25 percent of the 768 kbps, all traffic from Group 3 to 15 percent of the 768 kbps, and Group 4 to 10 percent of the 768 kbps.

Step 5 Verify the operation of the class-based shaper.

Lab 7-3: Class-Based Policer

In this activity, you will examine the class-based policer. The policer can be used to enforce the SLA from a customer on the ingress or it can be used to modify the marking of traffic based on thresholds. Here, you will enforce the policy that you created on R2 on the ingress of R1.

Activity Procedure

Complete these steps:

Step 1 On R1, create a policy that will police the traffic from R2 so that all traffic will be policed at 768 kbps.

Step 2 On R1, create a child policy that will police the traffic as follows:

- All traffic marked as AF43 to the minimum rate
- All traffic marked as AF42 to twice the minimum rate
- All traffic marked as AF41 to four times the minimum rate
- All traffic marked as EF to eight times the minimum rate

Step 3 Perform the following ping from R6:

```
ping 172.16.124.1 repeat 1000 size 1000 timeout 1
```

You should see an output similar to the following:

```
R6#ping 172.16.124.1 rep 1000 tim 1
Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 172.16.124.1, timeout is 1 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!
Success rate is 98 percent (982/1000), round-trip min/avg/max = 5/8/13 ms
```

Step 4 Verify that you are seeing drops in your policy on R1.

Lab 7-4: Avoiding and Managing Congestion

In this activity, you will examine how you can divide bandwidth between different types of traffic, and you will configure WRED for traffic that you are willing to drop.

Activity Procedure

Step 1 On R2, add to your existing policy that shapes the different types of traffic a queue that matches all traffic. Reserve 50 percent of the bandwidth for this new queue.

Step 2 Add the following policy within the new queue that matches all traffic:

- EF (Group 1): 50 percent
- Pre-emptive
- Burst of 1000 bytes
- Class of 3 gets 25 percent
- Class of 2 gets 15 percent
- Class of 1 gets 5 percent

Step 3 For the class of 1, make sure to use congestion avoidance using default thresholds for DSCP values.

Step 4 Ping from R2 to R1 with a class selector of 1.

Step 5 Verify that you get hits on the policy, including within the class of 1.

Module 8: Cisco Network Services

In this set of labs, you will examine the Cisco Network Services, as detailed in the CCIE Routing & Switching v5.0 blueprint.

This set of labs is based on the same topology that was configured in Lab 4-1. It is assumed that you have completed all of the Module 7 labs.

Lab 8-1: Implementing NAT

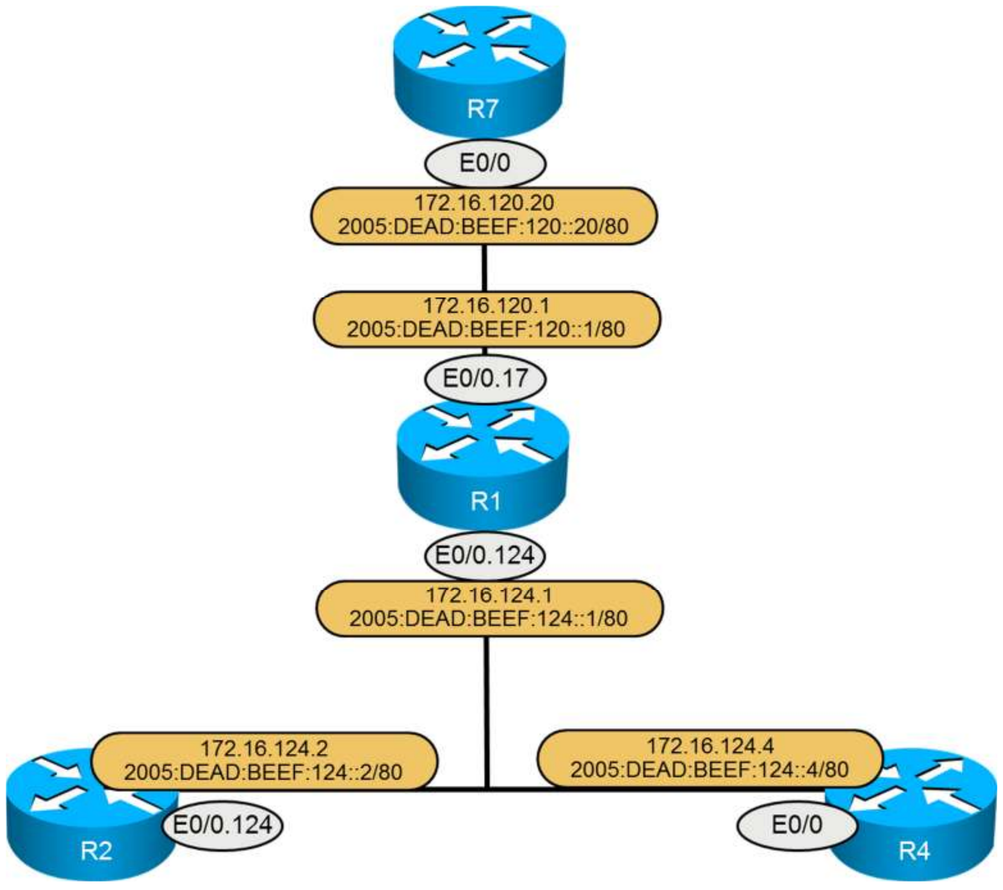
Activity Objective

This activity builds on previous labs. You do need to modify the configuration from Module 7. You will then configure NAT for IPv4. After completing this activity, you will be able to meet this objective:

- Establish additional connectivity between the devices using all four switches

Visual Objective

The figure illustrates what you will accomplish in this activity.



Task 1: Configure the Required Router Interfaces

In this task, you will need to change the current configuration and move addressing.

Activity Procedure

Complete these steps:

Step 1 Make sure the tunnel interfaces are shut down on R1, R2, and R4.

Step 2 Shut down the R1 and R2 S1/1 interfaces.

Step 3 Apply the IPv4 and IPv6 address from the tunnel interfaces to the E0/0.124 interface of R1 and R2, and the E0/0 interface of R4.

Step 4 Have R1 inject a default route to R7.

Step 5 Configure OSPF for IPv4 and IPv6 on the R2 E0/0.124 and R4 E0/0 interfaces only.

Task 2: Configure NAT and Verify Connectivity

In this task, you will configure NAT on R1 to hide the real IPv4 address of R7 from R2 and R4.

Step 1 Configure NAT on R1 so that any IPv6 address from R7 will be hidden from R2 and R4. You are not allowed to create a pool of addresses for this task, and you cannot introduce any new IPv4 addresses.

Step 2 Verify that R7 can ping the addresses of R2 and R4 on the shared network between R1, R2, and R4.

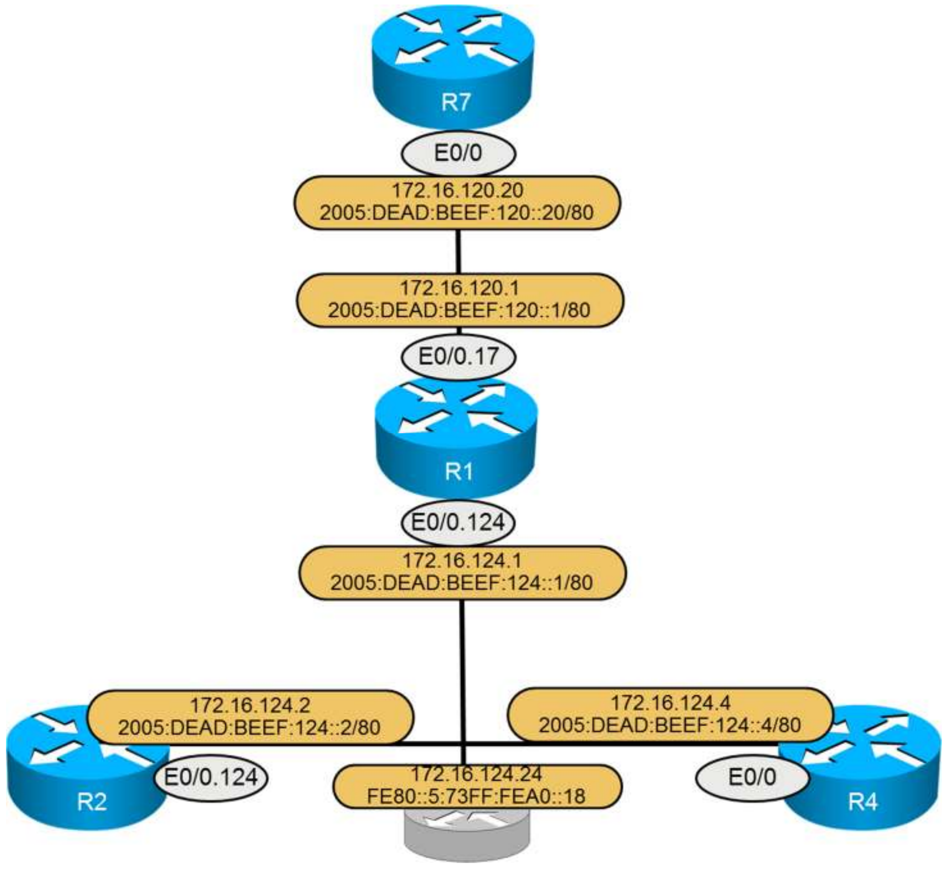
Lab 8-2: Implementing FHRP

Activity Objective

You will now configure FHRP for IPv4 and IPv6. In this section you will configure HSRP for IPv4 and IPv6, GLBP for IPv4 and IPv6, and VRRP for IPv4.

Visual Objective

The figure illustrates what you will accomplish in this activity.



Task 1: Configure HSRP for IPv4 and IPv6 and Verify Connectivity Router Interfaces

In this task, you will add HSRP for IPv4 and IPv6 on R2 and R4. You will make the virtual router addresses into the default route for R1.

Activity Procedure

Complete these steps:

Step 1 Add HSRP on the routers R2 and R4 with a set of virtual addresses of 172.16.124.24 and FE80::5:73FF:FEA0::18. Do not set the virtual IPv6 address directly.

Step 2 Make sure the virtual MAC address for the virtual IPv4 address is 0000.0C9F.F00C. Do not use the **standby mac-address** command.

Step 3 For IPv4, make sure that R2 is the active router with a priority of 1 above the default. Make sure that it waits a minimum of 30 seconds before taking over the active role.

Step 4 If the interface to R6 goes down on R2, have R4 take over with no delay. Use the default decrement.

Step 5 For IPv6, make sure that R4 is the active router with a priority of 2 above the default.

Step 6 On R1, make the virtual address your default route.

Step 7 On R2 and R4, make R1 the default for IPv6.

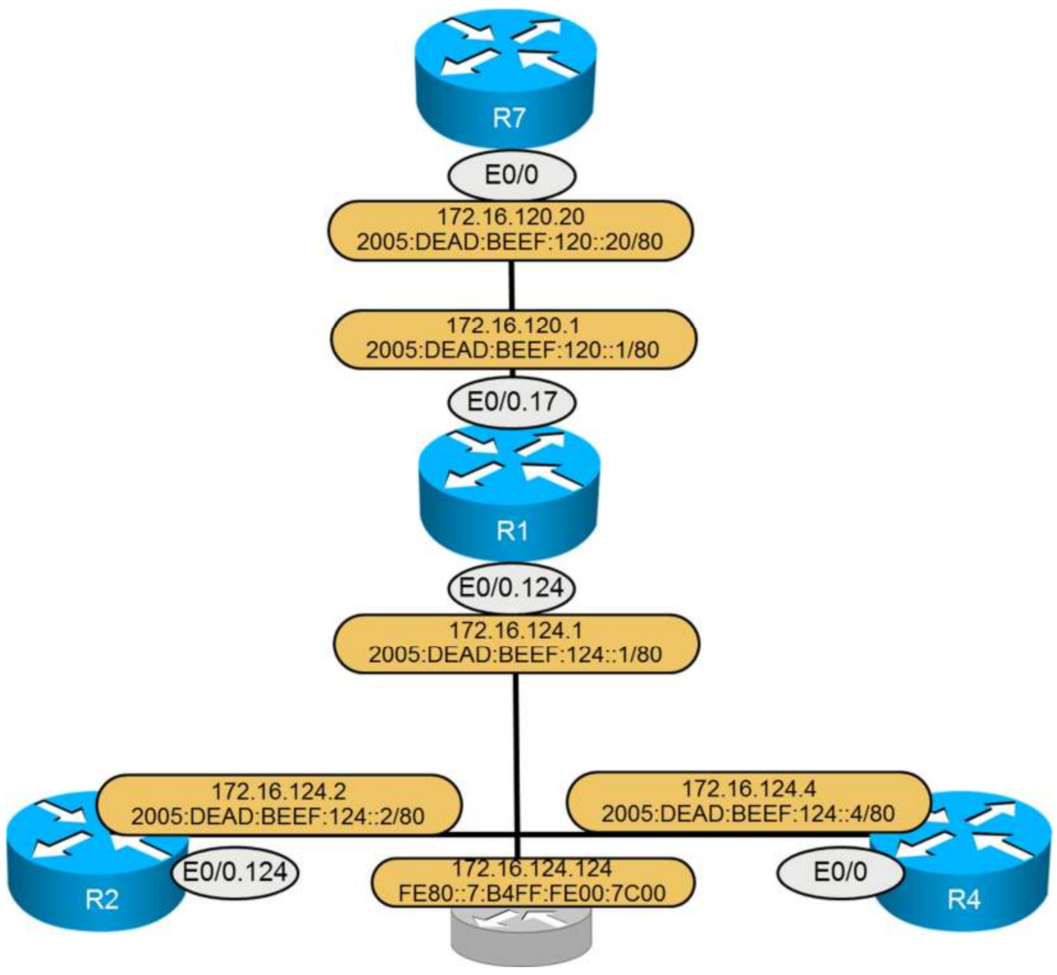
Step 8 Verify that R7 can ping the virtual addresses.

Task 2: Configure GLBP for IPv4 and IPv6 and Verify Connectivity

In this task, you will configure R2 and R4 for GLBP for IPv4 and IPv6. You will also change the default route for R1 to match the new virtual router addresses.

Visual Objective

The figure illustrates what you will accomplish in this activity.



Activity Procedure

Complete these steps:

Step 1 On R2 and R4, configure GLBP for IPv4 with a virtual IPv4 address of 172.16.124.124.

Step 2 Make sure that R2 is the AVG, as long as it is up.

Step 3 Have R4 stop acting as an AVF if the interface to its upstream router fails; use the default decrement.

Step 4 Configure GLBP for IPv6 with the virtual address of FE80::7:B4FF:FE00:7C00 without setting the address directly. Make sure that R4 is the AVG.

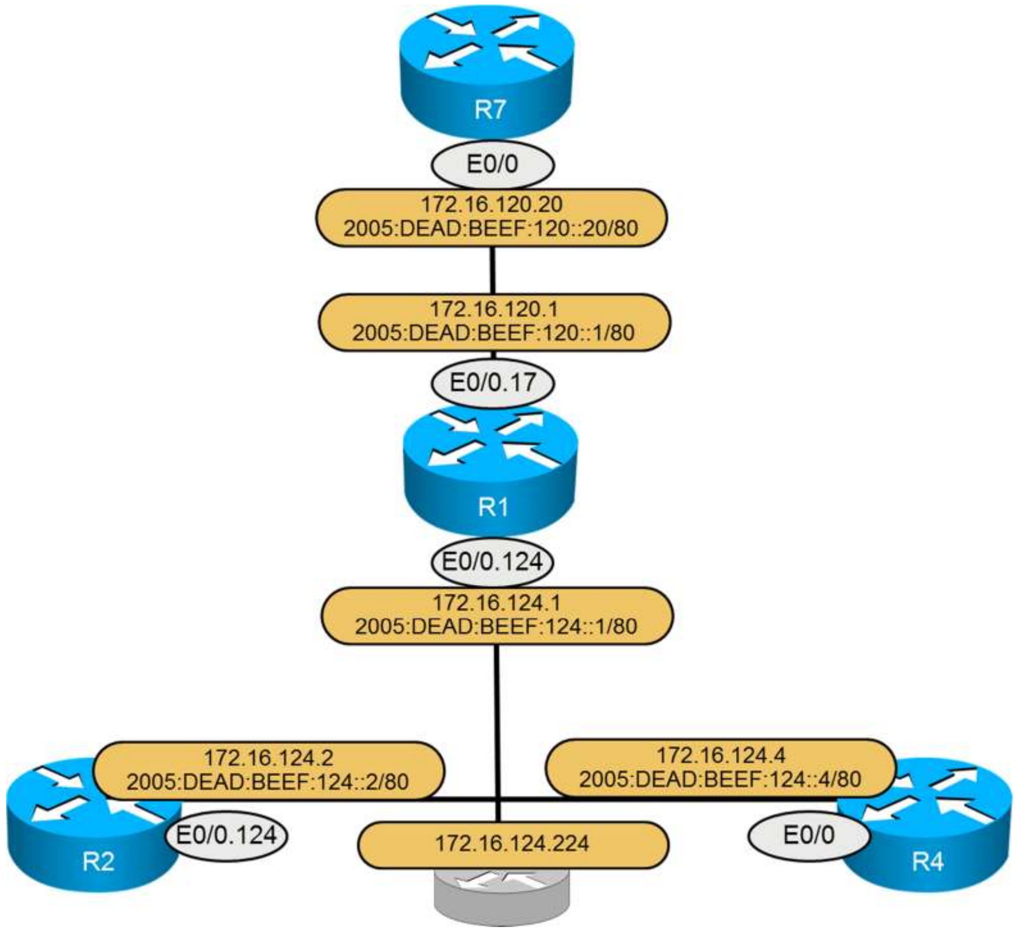
Step 5 Change the default gateway on R1 to match the virtual address from GLBP and verify that R7 can still ping the Loopback 10x addresses of R2 and R4.

Task 3: Configure VRRP for IPv4 and Verify Connectivity

In this task, you will configure R2 and R4 for VRRP for IPv4. You will also change the default route for R1 to match the new virtual router addresses.

Visual Objective

The figure illustrates what you will accomplish in this activity.



Activity Procedure

Complete these steps:

Step 1 On R2 and R4, configure VRRP for IPv4 with a virtual IPv4 address of 172.16.124.224.

Step 2 Make sure that R2 is the master for this group with a priority of 1 more than default.

Step 3 Make sure the virtual MAC address is 0000.5e00.01e0.

Step 4 Make sure that the master sends hellos every 250 ms. Do not configure timers on R4, have it learn from R2.

Step 5 Change the default gateway on R1 to match the virtual address from VRRP and verify that R7 can still ping the Loopback 10x addresses of R2 and R4.

Lab 8-3: Implementing NTP

Activity Objective

In this section, you will configure NTP to run over IPv4 and IPv6.

Useful Commands

This section lists some potentially useful commands.

Category	Commands
Configuration	**ntp master ntp server** **ntp authenticate** **ntp authentication-key** **ntp trusted-key**
Verification	**show ntp associations** **show ntp information** **show ntp packets** **show ntp status** **debug ntp events** **debug ntp packets**

Task 1: Configure NTP on R7

Activity Procedure

In this task, you will configure R7 to be the NTP server that will feed time to R1. R1 will feed the other routers with authentication.

Complete these steps:

Step 1 Configure R7 as the NTP master.

Step 2 Have R1 pull time from R7 using the IPv6 address of 2001:DEAD:BEEF:107::1.

Step 3 Have R2 and R4 get their time from their peer R1, without configuring the peer address. Make sure that NTP is secure between R1, R2, and R4 with a key of 42 and string of C1sc0.

Lab 8-4: Implementing DHCP for IPv4 and IPv6

Complete this lab activity to practice DHCP configuration for both IPv4 and IPv6.

Activity Objective

After completing this activity, you will be able to meet this objective:

- Deploy DHCP for both IPv4 and IPv6

Useful Commands

This section lists some potentially useful commands.

Category	Command
Configuration	**ip dhcp pool** **ip address dhcp** **ipv6 dhcp pool** **ipv6 dhcp server** **ipv6 address dhcp**
Verification	**show ip dhcp binding** **show ip dhcp pool** **show ip dhcp pool server** **show ipv6 dhcp binding** **show ipv6 dhcp pool** **show ipv6 dhcp interface**

Task 1: Configure DHCP for IPv4 and IPv6

In this task, you will configure R1 as the DHCP server for IPv4 and IPv6, and serve addressing to R7.

Activity Procedure

Complete these steps:

Step 1 Make R1 the DHCP server for R7 to issue its addresses for both IPv4 and IPv6. Make sure that R7 gets the correct IPv4 address as listed on the visual objective from previous tasks.

Answer Key

Here are the recommended solutions for the Lab Guide activities.

Lab 4-1 Answer Key: Establishing Basic Connectivity for BGP

You need to complete two tasks to establish basic connectivity.

Task 1: Configure Ethernet Switching

Step 1 Create the necessary VLANs, trunks, and access.

Router	Switch	VLAN
R1	SW1	17, 124
R2	SW1	26, 124
R4	SW1	124
R4	SW2	46
R6	SW1	26, 46
R7	SW3	17, 78
R8	SW3	78

Step 2 Use VTP transparent mode.

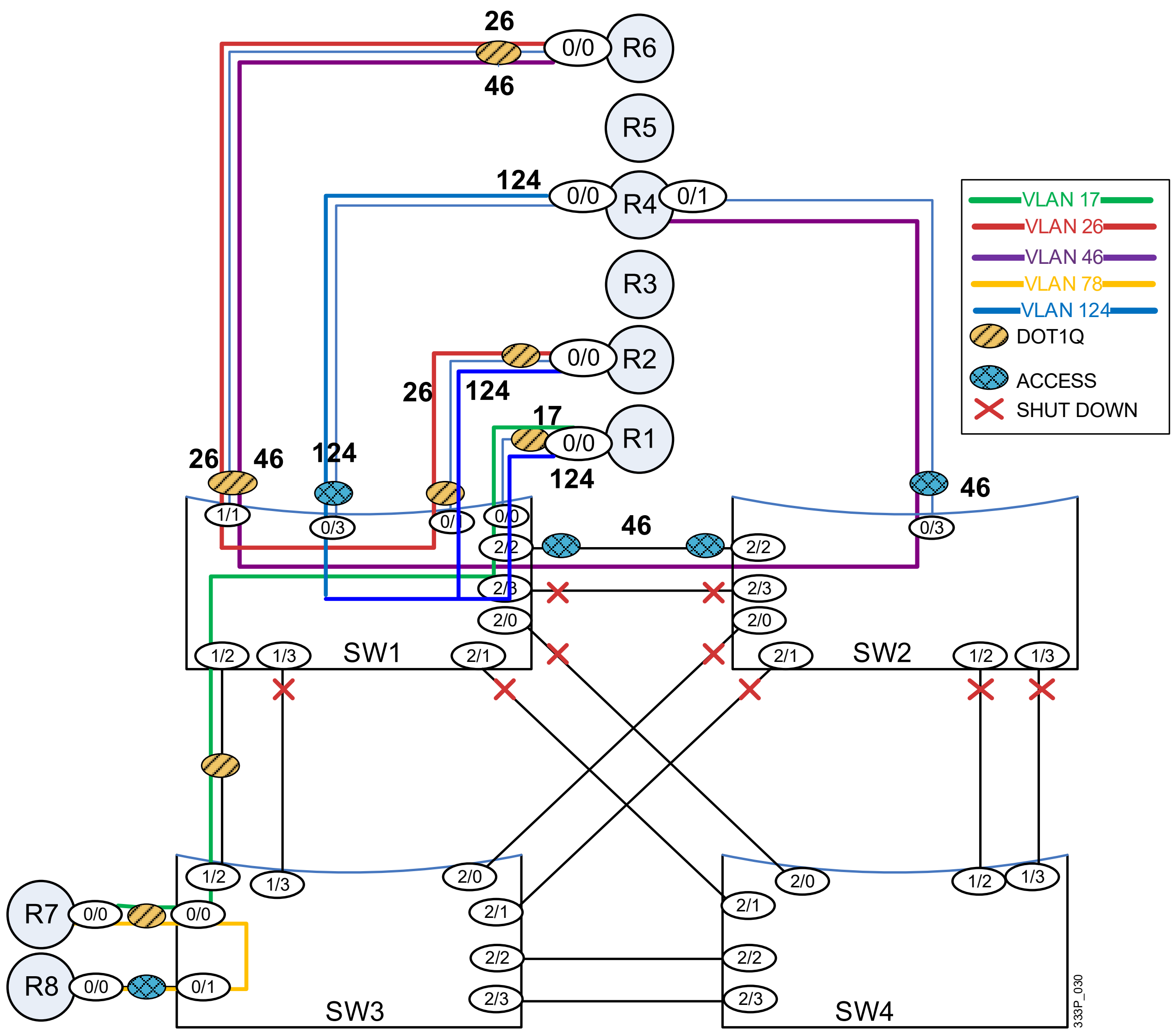
Step 3 Where trunking is required, use the IEEE standard trunking protocol. Use access mode on the first link between SW1 and SW2.

Step 4 Trunks should only allow the nessessary VLANs.

Solution to Task 1:

You must coordinate the provided Layer 1 topology and the required Layer 3 connectivity by creating the necessary VLANs and properly configuring the indicated switch and router ports. Many learners find it helpful to copy the relevant portions of the Layer 1 diagram and use colored pencils to indicate the path of each VLAN between routed ports in the same Ethernet subnets. The result might look something like the Layer 2 diagram that is shown.

Layer 2 Diagram



Remember that if you do not have access to colored pencils, you can create this diagram with a single color and just label the links per their function. R1, R2, R6, and R7 have two VLANs that are associated with their E0/0 interfaces, so they are configured as routers-on-a-stick using dot1q encapsulation. The SW1 ports E0/0, E0/1, and E1/1 and the SW3 port E0/0 have to be configured as trunks to match up with the routers. The other router-to-switch connections are all access ports. The only other interesting step is Step 3, "Use access mode on the first link between SW1 and SW2." The first link between SW1 and SW2 is over the E2/2 interfaces. As for which VLAN needs to go across this link, VLAN 46 appears on R4 E0/1, which connects to SW2, and R6 has a trunk with VLANs 26 and 46. Therefore, VLAN 46 must cross the ports between SW1 and SW2.

```
SW1#sh vlan

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Et0/2, Et1/0, Et1/3, Et2/0
                                                Et2/1, Et2/3
17   VLAN0017                         active
26   VLAN0026                         active
```

```
46   VLAN0046                         active    Et2/2
124  VLAN0124                         active    Et0/3
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        0      0
17   enet  100017     1500  -      -      -        -    -        0      0
26   enet  100026     1500  -      -      -        -    -        0      0
46   enet  100046     1500  -      -      -        -    -        0      0
124  enet  100124     1500  -      -      -        -    -        0      0
1002 fddi  101002     1500  -      -      -        -    -        0      0
1003 tr    101003     1500  -      -      -        -    -        0      0

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1004 fdnet 101004     1500  -      -      -        ieee -        0      0
1005 trnet 101005     1500  -      -      -        ibm  -        0      0

Primary Secondary Type              Ports
------- --------- ----------------- ------------------------------------------

SW1#show interfaces trunk

Port        Mode             Encapsulation  Status        Native vlan
Et0/0       on               802.1q         trunking      1
Et0/1       on               802.1q         trunking      1
Et1/1       on               802.1q         trunking      1
Et1/2       on               802.1q         trunking      1

Port        Vlans allowed on trunk
Et0/0       17,124
Et0/1       26,124
Et1/1       26,46
Et1/2       17

Port        Vlans allowed and active in management domain
Et0/0       17,124
Et0/1       26,124
Et1/1       26,46
Et1/2       17

Port        Vlans in spanning tree forwarding state and not pruned
Et0/0       17,124
Et0/1       26,124
Et1/1       26,46

Port        Vlans in spanning tree forwarding state and not pruned
Et1/2       17
hostname SW1
!
vtp domain CCIE
vtp mode transparent
!
vlan 17,26,46,124
!
interface Ethernet0/0
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 17,124
 switchport mode trunk
 duplex auto
!
interface Ethernet0/1
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 26,124
 switchport mode trunk
 duplex auto
!
interface Ethernet0/3
 switchport access vlan 124
 switchport mode access
 duplex auto
!
interface Ethernet1/1
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 26,46
 switchport mode trunk
 duplex auto
!
interface Ethernet1/2
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 17
 switchport mode trunk
 duplex auto
!
interface Ethernet2/2
 switchport access vlan 46
 switchport mode access
 duplex auto
!
SW2#show vlan

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Et0/0, Et0/1, Et0/2, Et1/0
                                                Et1/1, Et1/2, Et1/3, Et2/0
                                                Et2/1, Et2/3
46   VLAN0046                         active    Et0/3, Et2/2
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        0      0
46   enet  100046     1500  -      -      -        -    -        0      0
1002 fddi  101002     1500  -      -      -        -    -        0      0
1003 tr    101003     1500  -      -      -        -    -        0      0
1004 fdnet 101004     1500  -      -      -        ieee -        0      0
1005 trnet 101005     1500  -      -      -        ibm  -        0      0

Primary Secondary Type              Ports
------- --------- ----------------- ------------------------------------------

SW2#show run
!
hostname SW2
!
vtp domain CCIE
vtp mode transparent
!
vlan 46
!
interface Ethernet0/3
 switchport access vlan 46
 switchport mode access
 duplex auto
!
interface Ethernet2/2
 switchport access vlan 46
 switchport mode access
 duplex auto
```

```
!
SW3#show vlan
```

```
VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Et0/2, Et0/3, Et1/0, Et1/1
                                                Et1/3, Et2/0, Et2/1, Et2/2
                                                Et2/3
17   VLAN0017                         active
78   VLAN0078                         active    Et0/1
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        0      0
17   enet  100017     1500  -      -      -        -    -        0      0
78   enet  100078     1500  -      -      -        -    -        0      0
1002 fddi  101002     1500  -      -      -        -    -        0      0
1003 tr    101003     1500  -      -      -        -    -        0      0
1004 fdnet 101004     1500  -      -      -        ieee -        0      0
1005 trnet 101005     1500  -      -      -        ibm  -        0      0

Primary Secondary Type              Ports
------- --------- ----------------- ------------------------------------------

SW3#show interfaces trunk

Port        Mode             Encapsulation  Status        Native vlan
Et0/0       on               802.1q         trunking      1
Et1/2       on               802.1q         trunking      1

Port        Vlans allowed on trunk
Et0/0       17,78
Et1/2       17

Port        Vlans allowed and active in management domain
Et0/0       17,78
Et1/2       17

Port        Vlans in spanning tree forwarding state and not pruned
Et0/0       17,78
Et1/2       17
SW3#show run
!
hostname SW3
!
vtp domain CCIE
vtp mode transparent
!
vlan 17,78
!
interface Ethernet0/0
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 17,78
 switchport mode trunk
 duplex auto
!
interface Ethernet0/1
 switchport access vlan 78
 switchport mode access
 duplex auto
!
interface Ethernet1/2
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 17
 switchport mode trunk
 duplex auto
```

```
R1#show running-config | begin 0/0
interface Ethernet0/0
 no ip address
!
interface Ethernet0/0.17
 encapsulation dot1Q 17
 ip address 172.16.120.1 255.255.255.0
 ipv6 address 2005:DEAD:BEEF:120::1/80
 ipv6 ospf 1 area 0
!
interface Ethernet0/0.124
 encapsulation dot1Q 124
 ip address 10.0.0.1 255.255.255.0
!
```

Task 2: Configure the Serial Connections

Step 1 Configure a back-to-back serial link between R1 and R3 using the default encapsulation for subnet 172.16.13.0/24 and prefix 2005:DEAD:BEEF:13::/80.

Step 2 Configure a back-to-back serial link between R4 and R5 for subnet 172.16.45.0/2. and prefix 2005:DEAD:BEEF:45::/80.

Step 3 Configure Layer 2 authentication between R4 and R5 with a password of C1sc0. Make sure that the password is not sent.

Step 4 Configure the back-to-back serial interfaces between R3 and R4 using the default encapsulation for subnet 172.16.43.0/24 and prefix 2005:DEAD:BEEF:43::/80.

Solution to Task 2:

There is nothing too special with the back-to-back connections in Step 1 and Step 4. The only one that is interesting is connection between R4 and R5. Step 3 is trying to lead you to use PPP. The second sentence, “Make sure that the password is not sent,” indicates that you should use CHAP, which does not send the password like PAP does. CHAP sends the hostname and a key and then sends a hash of the key and password. CHAP never sends the password itself.

```
R4#show interfaces serial 1/0
Serial1/0 is up, line protocol is up
  Hardware is M4T
  Internet address is 172.16.45.4/24
  MTU 1500 bytes, BW 1544 Kbit/sec, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation PPP, LCP Open
  Open: IPCP, IPV6CP, CDPCP, crc 16, loopback not set
  Keepalive set (10 sec)
  Restart-Delay is 0 secs
  Last input 00:00:03, output 00:00:03, output hang never
  Last clearing of "show interface" counters 00:22:57
  Input queue: 0/75/0/0 (size/max/drshow intops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     436 packets input, 22057 bytes, 0 no buffer
     Received 0 broadcasts (0 IP multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     435 packets output, 22247 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 unknown protocol drops
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up

R4#show ppp interface serial 1/0
PPP Serial Context Info
-------------------
```

```
Interface        : Se1/0
PPP Serial Handle: 0x61000001
PPP Handle       : 0x6C000001
SSS Handle       : 0x98000001
AAA ID           : 12
Access IE        : 0xEF000001
SHDB Handle      : 0x0
State            : Up
Last State       : Binding
Last Event       : LocalTerm

PPP Session Info
----------------
Interface        : Se1/0
PPP ID           : 0x6C000001
Phase            : UP
Stage            : Local Termination
Peer Name        : R5
Peer Address     : 172.16.45.5
Control Protocols: LCP[Open] CHAP+ IPCP[Open] IPV6CP[Open] CDPCP[Open]
Session ID       : 1
AAA Unique ID    : 12
SSS Manager ID   : 0x98000001
SIP ID           : 0x61000001
PPP_IN_USE       : 0x11

Se1/0 LCP: [Open]
Our Negotiated Options
Se1/0 LCP:    AuthProto CHAP (0x0305C22305)
Se1/0 LCP:    MagicNumber 0xBBCC1515 (0x0506BBCC1515)
Peer's Negotiated Options
Se1/0 LCP:    AuthProto CHAP (0x0305C22305)
Se1/0 LCP:    MagicNumber 0xBBCC1507 (0x0506BBCC1507)

Se1/0 IPCP: [Open]
Our Negotiated Options
Se1/0 IPCP:    Address 172.16.45.4 (0x0306AC102D04)
Peer's Negotiated Options
Se1/0 IPCP:    Address 172.16.45.5 (0x0306AC102D05)

Se1/0 IPV6CP: [Open]
Our Negotiated Options
Se1/0 IPV6CP:    Interface-Id A8BB:CCFF:FE00:0400 (0x010AA8BBCCFFFE000400)
Peer's Negotiated Options
Se1/0 IPV6CP:    Interface-Id A8BB:CCFF:FE00:0500 (0x010AA8BBCCFFFE000500)

Se1/0 CDPCP: [Open]
Our Negotiated Options
  NONE
Peer's Negotiated Options
  NONE
R4#show run
!
hostname R4
!
username R5 password 0 C1sc0
!
interface Serial1/0
 ip address 172.16.45.4 255.255.255.0
 encapsulation ppp
 ipv6 address 2005:DEAD:BEEF:45::4/80
 ppp authentication chap
 serial restart-delay 0
!
```

Note that the interfaces that are shown may differ depending on the equipment in your lab, and you may have to set the clock rate on the DCE end of the connection.

Task 3: Configure the DMVPN Connections

Step 1 Configure DMVPN connections as shown in the Visual Objective diagram between R1, R2, and R4. All IPv4 and IPv6 addresses that are shown on the diagram should be associated with the tunnel interface of the routers.

Step 2 Configure the physical or subinterface IPv4 addressing area as follows:

- R1 10.0.0.1/24
- R2 10.0.0.2/24
- R4 10.0.0.4/24.

Step 3 Once verified, shut down the tunnel interface on R4.

Solution to Task 3:

DMVPN may or may not involve security. In this case, it does not. There is no requirement for IPsec in order to make the DMVPN function. You need to map the spoke routers to the hub router for both IPv4 and IPv6. The underlying path does not have to support IPv6 for the tunnel to carry IPv6 traffic; this is a tunnel, after all. The IPv6 address of the hub is mapped to its IPv4 address, just as the IPv4 address is on the spoke routers. Give a common network ID to all the spokes and the hub.

Note The network IDs for IPv4 and IPv6 are independent of each other. They could be the same or different, there is no effect. They have to be common within the protocol (either IPv4 or IPv6).

```
R1#show dmvpn detail
Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
        N - NATed, L - Local, X - No Socket
        # Ent --> Number of NHRP entries with same NBMA peer
        NHS Status: E --> Expecting Replies, R --> Responding, W --> Waiting
        UpDn Time --> Up or Down Time for a Tunnel
==========================================================================

Interface Tunnel0 is up/up, Addr. is 172.16.124.1, VRF ""
   Tunnel Src./Dest. addr: 10.0.0.1/MGRE, Tunnel VRF ""
   Protocol/Transport: "multi-GRE/IP", Protect ""
   Interface State Control: Disabled
   nhrp event-publisher : Disabled
Type:Hub, Total NBMA Peers (v4/v6): 2

# Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb    Target Network
----- --------------- --------------- ----- -------- ----- ----------------
    1 10.0.0.2           172.16.124.2    UP 00:36:15    D    172.16.124.2/32
    1 10.0.0.4           172.16.124.4    UP 00:36:10    D    172.16.124.4/32

Interface Tunnel0 is up/up, Addr. is 2005:DEAD:BEEF:124::1, VRF ""
   Tunnel Src./Dest. addr: 10.0.0.1/MGRE, Tunnel VRF ""
   Protocol/Transport: "multi-GRE/IP", Protect ""
   Interface State Control: Disabled
   nhrp event-publisher : Disabled
Type:Hub, Total NBMA Peers (v4/v6): 2
    1.Peer NBMA Address: 10.0.0.2
        Tunnel IPv6 Address: 2005:DEAD:BEEF:124::2
        IPv6 Target Network: 2005:DEAD:BEEF:124::2/128
        # Ent: 2, Status: UP, UpDn Time: 00:36:12, Cache Attrib: D
    2.Peer NBMA Address: 10.0.0.2
        Tunnel IPv6 Address: 2005:DEAD:BEEF:124::2
        IPv6 Target Network: FE80::A8BB:CCFF:FE00:200/128
        # Ent: 0, Status: UP, UpDn Time: 00:36:12, Cache Attrib: D
    3.Peer NBMA Address: 10.0.0.4
        Tunnel IPv6 Address: 2005:DEAD:BEEF:124::4
        IPv6 Target Network: 2005:DEAD:BEEF:124::4/128
        # Ent: 2, Status: UP, UpDn Time: 00:35:21, Cache Attrib: D
    4.Peer NBMA Address: 10.0.0.4
        Tunnel IPv6 Address: 2005:DEAD:BEEF:124::4
        IPv6 Target Network: FE80::A8BB:CCFF:FE00:400/128
        # Ent: 0, Status: UP, UpDn Time: 00:35:21, Cache Attrib: D


Crypto Session Details:
--------------------------------------------------------------------------------
-

Pending DMVPN Sessions:

R2#show dmvpn detail
Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
        N - NATed, L - Local, X - No Socket
        # Ent --> Number of NHRP entries with same NBMA peer
        NHS Status: E --> Expecting Replies, R --> Responding, W --> Waiting
        UpDn Time --> Up or Down Time for a Tunnel
==========================================================================

Interface Tunnel0 is up/up, Addr. is 172.16.124.2, VRF ""
   Tunnel Src./Dest. addr: 10.0.0.2/MGRE, Tunnel VRF ""
   Protocol/Transport: "multi-GRE/IP", Protect ""
   Interface State Control: Disabled
   nhrp event-publisher : Disabled

IPv4 NHS:
172.16.124.1  RE priority = 0 cluster = 0
Type:Spoke, Total NBMA Peers (v4/v6): 1

# Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb    Target Network
----- --------------- --------------- ----- -------- ----- -----------------
    1 10.0.0.1           172.16.124.1    UP 00:36:56    S    172.16.124.1/32

Interface Tunnel0 is up/up, Addr. is 2005:DEAD:BEEF:124::2, VRF ""
   Tunnel Src./Dest. addr: 10.0.0.2/MGRE, Tunnel VRF ""
   Protocol/Transport: "multi-GRE/IP", Protect ""
   Interface State Control: Disabled
   nhrp event-publisher : Disabled

IPv6 NHS:
2005:DEAD:BEEF:124::1  RE priority = 0 cluster = 0
Type:Spoke, Total NBMA Peers (v4/v6): 1
    1.Peer NBMA Address: 10.0.0.1
        Tunnel IPv6 Address: 2005:DEAD:BEEF:124::1
        IPv6 Target Network: 2005:DEAD:BEEF:124::/80
        # Ent: 2, Status: UP, UpDn Time: 00:36:53, Cache Attrib: S
    2.Peer NBMA Address: 10.0.0.1
        Tunnel IPv6 Address: FE80::A8BB:CCFF:FE00:100
        IPv6 Target Network: FE80::A8BB:CCFF:FE00:100/128
        # Ent: 0, Status: NHRP, UpDn Time: never, Cache Attrib: S


Crypto Session Details:
-------------------------------------------------------------------------------

Pending DMVPN Sessions:

R2#show dmvpn interface tunnel 0
Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
        N - NATed, L - Local, X - No Socket
        # Ent --> Number of NHRP entries with same NBMA peer
        NHS Status: E --> Expecting Replies, R --> Responding, W --> Waiting
        UpDn Time --> Up or Down Time for a Tunnel
==========================================================================

Interface: Tunnel0, IPv4 NHRP Details
Type:Spoke, NHRP Peers:1,
 # Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb
 ----- --------------- --------------- ----- -------- -----
     1 10.0.0.1           172.16.124.1    UP 00:37:54     S

Interface: Tunnel0, IPv6 NHRP Details
Type:Spoke, Total NBMA Peers (v4/v6): 1
    1.Peer NBMA Address: 10.0.0.1
        Tunnel IPv6 Address: 2005:DEAD:BEEF:124::1
        IPv6 Target Network: 2005:DEAD:BEEF:124::/80
        # Ent: 1, Status: UP, UpDn Time: 00:37:51, Cache Attrib: S

R2#ping 172.16.124.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.124.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
R2#show ip route
Codes: L - local, 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, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is not set

      2.0.0.0/32 is subnetted, 1 subnets
C        2.2.2.2 is directly connected, Loopback0
      10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C        10.0.0.0/24 is directly connected, Ethernet0/0.124
L        10.0.0.2/32 is directly connected, Ethernet0/0.124
      147.10.0.0/16 is variably subnetted, 2 subnets, 2 masks
C        147.10.1.128/27 is directly connected, Loopback147
L        147.10.1.145/32 is directly connected, Loopback147
      157.10.0.0/28 is subnetted, 1 subnets
B        157.10.1.208 [20/0] via 172.16.124.1, 00:38:33
      172.16.0.0/16 is variably subnetted, 4 subnets, 2 masks
C        172.16.26.0/24 is directly connected, Ethernet0/0.26
L        172.16.26.2/32 is directly connected, Ethernet0/0.26
C        172.16.124.0/24 is directly connected, Tunnel0
L        172.16.124.2/32 is directly connected, Tunnel0
B     200.1.16.0/24 [20/0] via 172.16.26.6, 00:38:33
B     200.1.17.0/24 [20/0] via 172.16.26.6, 00:38:33
B     200.1.18.0/24 [20/0] via 172.16.26.6, 00:38:33
B     200.1.19.0/24 [20/0] via 172.16.26.6, 00:38:33
R2#ping 2005:DEAD:BEEF:124::1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2005:DEAD:BEEF:124::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/2 ms
R2#show ipv6 route 2005:DEAD:BEEF:124::1
Routing entry for 2005:DEAD:BEEF:124::/80
  Known via "connected", distance 0, metric 0, type connected
  Route count is 1/1, share count 0
  Routing paths:
    directly connected via Tunnel0
      Last updated 00:41:51 ago
R1#show running-config interface tunnel 0
Building configuration...

Current configuration : 295 bytes
!
interface Tunnel0
 ip address 172.16.124.1 255.255.255.0
 no ip redirects
 ip mtu 1416
 ip nhrp map multicast dynamic
 ip nhrp network-id 1
 ipv6 address 2005:DEAD:BEEF:124::1/80
 ipv6 nhrp map multicast dynamic
 ipv6 nhrp network-id 1
 tunnel source 10.0.0.1
 tunnel mode gre multipoint
R2#show running-config interface tunnel 0
Building configuration...

Current configuration : 477 bytes
!
interface Tunnel0
 ip address 172.16.124.2 255.255.255.0
 no ip redirects
 ip mtu 1416
 ip nhrp map 172.16.124.1 10.0.0.1
 ip nhrp map multicast 10.0.0.1
 ip nhrp network-id 1
 ip nhrp nhs 172.16.124.1
 ipv6 address 2005:DEAD:BEEF:124::2/80
 ipv6 nhrp map multicast dynamic
 ipv6 nhrp map multicast 10.0.0.1
 ipv6 nhrp map 2005:DEAD:BEEF:124::1/80 10.0.0.1
 ipv6 nhrp network-id 1
 ipv6 nhrp nhs 2005:DEAD:BEEF:124::1
 tunnel source 10.0.0.2
 tunnel mode gre multipoint
```

Task 4: Implement the Required IGPs

Step 1 Configure OSPF and EIGRP as the Visual Objective diagram shows for both IPv4 and IPv6.

Step 2 OSPF must have *only* subnet 172.16.120.0/24 and 2005:DEAD:BEEF:120::/80 as internal routes.

Step 3 EIGRP includes *only* subnets 172.16.13.0/24 and 172.16.43.0/24 for IPv4, and 2005:DEAD:BEEF:13::/80 and 2005:dead:beef:43::/80 for IPv6.

Step 4 Do not redistribute EIGRP or connected routes into OSPF.

Step 5 Verify that R1 can reach 172.16.43.4 and 2005:DEAD:BEEF:43::4, and that R4 can reach 172.16.13.1 and 2005:DEAD:BEEF:13::1.

Step 6 On R4, shut down the tunnel interface that is associated with the IP address 172.16.124.4.

Solution to Task 4:

The requirements in this section provide a foundation of limited Layer 3 connectivity and a platform on which to demonstrate several features of BGP. EIGRP is configured to permit a peering between R1 and R4 in the absence of a direct link. OSPF is provided to satisfy the synchronization requirement. Next-hop reachability issues will arise because the border networks of AS 100 are not advertised into its IGPs. Shutting down the R4 tunnel interface introduces a challenging interaction between underlying Layer 3 connectivity and the assumptions made by BGP.

```
R1#show ip ospf neighbor

Neighbor ID     Pri   State          Dead Time   Address         Interface
7.7.7.7           1   FULL/BDR       00:00:31    172.16.120.7    Ethernet0/0.17
R1#show ipv6 ospf neighbor

            OSPFv3 Router with ID (172.16.124.1) (Process ID 1)

Neighbor ID     Pri   State          Dead Time   Interface ID    Interface
7.7.7.7           1   FULL/BDR       00:00:30    15              Ethernet0/0.17
R1#show ip eigrp neighbors
EIGRP-IPv4 Neighbors for AS(1)
H   Address              Interface            Hold Uptime   SRTT   RTO  Q  Seq
                                                (sec)       (ms)       Cnt Num
0   172.16.13.3            Se1/0                12 00:44:51   15   100  0  2
R1#show ipv6 eigrp neighbors
EIGRP-IPv6 Neighbors for AS(1)
H   Address                Interface          Hold Uptime   SRTT   RTO  Q  Seq
                                                  (sec)         (ms)   Cnt Num
0   Link-local address:    Se1/0                11 00:44:56   10   100  0  4
    FE80::A8BB:CCFF:FE00:300

R1#show run
!
hostname R1
!
interface Ethernet0/0
 no ip address
!
interface Ethernet0/0.17
 encapsulation dot1Q 17
 ip address 172.16.120.1 255.255.255.0
 ipv6 address 2005:DEAD:BEEF:120::1/80
 ipv6 ospf 1 area 0
!
interface Serial1/0
 ip address 172.16.13.1 255.255.255.0
 ipv6 address 2005:DEAD:BEEF:13::1/80
 ipv6 eigrp 1
 serial restart-delay 0
!
router eigrp 1
 network 172.16.13.1 0.0.0.0
!
router ospf 1
 router-id 1.1.1.1
 redistribute bgp 65001 subnets
 network 172.16.120.1 0.0.0.0 area 0
!
ipv6 router eigrp 1
!
ipv6 router ospf 1
!
```

Lab 4-2 Answer Key: Configuring BGP

Task 1: Implement BGP Confederations

Step 1 Create AS 100 as a confederation of AS 65001 and AS 65002.

Step 2 Place R1 and R7 in AS 65001.

Step 3 Place R3 and R4 in AS 65002.

Step 4 Configure peering between AS 65001 and AS 65002 only between R1 and R4. Use addresses 172.16.13.1/2005:DEAD:BEEF:13::1 and 172.16.43.4/2005:DEAD:BEEF:43::4.

Solution to Task 1:

The full-mesh requirement is in place in IBGP to avoid looped advertisements within an AS. By default, BGP detects loops by looking for its own AS number in the AS path. But within a single AS, the path does not normally change. If it was permitted to advertise a route that was learned from one IBGP peer to another IBGP, peer loops could not be detected. Therefore, each IBGP router must peer with every other IBGP router.

Route reflectors permit less than a full mesh by adding the originator ID and the cluster list as added loop-prevention mechanisms. Confederations permit a relaxation of the full-mesh requirement by breaking up a given AS into subautonomous systems. Each sub-AS must be fully meshed, but a full mesh between routers in different subautonomous systems is not required. When updates are advertised between subautonomous systems, the AS path is prepended with the sub-AS number. This replicates the AS path method of loop detection within a single AS. Routers that are external to the confederation peer with the original AS and know nothing of the confederation.

Here are the peering configurations for R1 and R4 for this section:

```
R1#show run | section bgp
!
router bgp 65001
 bgp router-id 1.1.1.1
 bgp log-neighbor-changes
 bgp confederation identifier 100
 bgp confederation peers 65002
 neighbor 2005:DEAD:BEEF:43::4 remote-as 65002
 neighbor 2005:DEAD:BEEF:43::4 ebgp-multihop 2
 neighbor 2005:DEAD:BEEF:120::7 remote-as 65001
 neighbor 2005:DEAD:BEEF:124::2 remote-as 200
 neighbor 172.16.43.4 remote-as 65002
 neighbor 172.16.43.4 ebgp-multihop 2
 neighbor 172.16.120.7 remote-as 65001
 !
 address-family ipv4
  synchronization
  no neighbor 2005:DEAD:BEEF:43::4 activate
  no neighbor 2005:DEAD:BEEF:120::7 activate
  neighbor 172.16.43.4 activate
  neighbor 172.16.120.7 activate
  neighbor 172.16.120.7 next-hop-self
 exit-address-family
 !
 address-family ipv6
  neighbor 2005:DEAD:BEEF:43::4 activate
  neighbor 2005:DEAD:BEEF:43::4 next-hop-self
  neighbor 2005:DEAD:BEEF:120::7 activate
  neighbor 2005:DEAD:BEEF:120::7 next-hop-self
 exit-address-family
R4#show run | section bgp
!
```

```
router bgp 65002
 bgp router-id 4.4.4.4
 bgp log-neighbor-changes
 bgp confederation identifier 100
 bgp confederation peers 65001
 neighbor 2005:DEAD:BEEF:13::1 remote-as 65001
 neighbor 2005:DEAD:BEEF:13::1 ebgp-multihop 2
 neighbor 2005:DEAD:BEEF:43::3 remote-as 65002
 neighbor 172.16.13.1 remote-as 65001
 neighbor 172.16.13.1 ebgp-multihop 2
 neighbor 172.16.43.3 remote-as 65002
 !
 address-family ipv4
  no neighbor 2005:DEAD:BEEF:13::1 activate
  no neighbor 2005:DEAD:BEEF:43::3 activate
  neighbor 172.16.13.1 activate
  neighbor 172.16.43.3 activate
exit-address-family
 !
 address-family ipv6
  neighbor 2005:DEAD:BEEF:13::1 activate
  neighbor 2005:DEAD:BEEF:43::3 activate
exit-address-family
```

Notice that the BGP process ID is the sub-AS number; and AS 100 is configured as the confederation identifier. The **bgp confederation peers** statement lists the subautonomous system peers. Even though R1 and R4 are configured with different AS numbers, the peering between R1 and R4 is like an IBGP peering in these five ways:

1. The next hop is not changed when updates are sent between confederation peers.
2. The local-preference attribute is sent between peers in different subautonomous systems.
3. The routes that are learned from a sub-AS peer are placed in the routing table with an administrative distance of 200 instead of 20.
4. The AS path length value is not affected; confederation subautonomous system numbers are not counted.
5. The multi-exit discriminator (MED) is not effective between subautonomous systems.

In every other way, however, the peering between R1 and R4 is an EBGP peering. For example, the command **ebgp-multihop** is required to increase the TTL.

Verification

Verify your peerings with the command **show ip bgp summary**. Here is the expected result on R1:

```
R1#show ip bgp summary
BGP router identifier 1.1.1.1, local AS number 65001
BGP table version is 11, main routing table version 11
0 network entries using 0 bytes of memory
0 path entries using 0 bytes of memory
0/0 BGP path/bestpath attribute entries using 0 bytes of memory
0 BGP AS-PATH entries using 0 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 2524 total bytes of memory
BGP activity 0/0 prefixes, 0/0 paths, scan interval 60 secs

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd
172.16.43.4     4     65002      98      96       11    0    0 01:24:55      0
172.16.120.7    4     65001      96      98       11    0    0 01:24:14      0
172.16.124.2    4       200      97      97       11    0    0 01:24:03      0
R1#show bgp ipv6 unicast summary
```

```
BGP router identifier 1.1.1.1, local AS number 65001
BGP table version is 11, main routing table version 11
0 network entries using 0 bytes of memory
0 path entries using 0 bytes of memory
0/0 BGP path/bestpath attribute entries using 0 bytes of memory
0 BGP AS-PATH entries using 0 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 2240 total bytes of memory
BGP activity 0/0 prefixes, 0/0 paths, scan interval 60 secs

Neighbor        V   AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd
2005:DEAD:BEEF:43::4
                4      65002      98      97       11    0    0 01:25:09      0
2005:DEAD:BEEF:120::7
                4      65001      96     100       11    0    0 01:24:31      0
2005:DEAD:BEEF:124::2
                 4       200      95      97       11    0    0 01:24:13      0
```

Note that the State/PfxRcd value is zero, indicating the number of prefixes that are learned from this neighbor. This value is expected at this point. A failed peering would be indicated by a state, such as “active” or “idle.”

Task 2: Implement EBGP Neighbors to AS 100

Step 1 Configure AS 200 on R2 and advertise Loopback 147 into BGP using a network statement.

Step 2 Peer R2 and R1.

Step 3 Configure AS 500 on R5 and advertise Loopback 157 into BGP using a network statement.

Step 4 Peer R5 and R4.

Step 5 Configure AS 800 on R8.

Step 6 Peer R8 and R7.

Step 7 Verify that all BGP speakers have prefixes from R2 Loopback 147 and R5 Loopback 157 in their routing tables from BGP. Make sure that you have reachability between these subnets using an extended **ping** with a **source** option (a source ping) and an extended **traceroute** with a **source** option (a source trace) from R5.

Solution to Task 2:

The challenge here is to remember that AS 200 and AS 500 are peering with AS 100, not with the sub-AS numbers. Also remember that the prefix and mask length that are used in the network statement must precisely match values in the routing table. Here is the configuration for R2. R5 would be similar.

```
R2#show run | section bgp
!
router bgp 200
 bgp router-id 2.2.2.2
 bgp log-neighbor-changes
 neighbor 2005:DEAD:BEEF:124::1 remote-as 100
 neighbor 172.16.124.1 remote-as 100
 !
 address-family ipv4
  network 147.10.1.128 mask 255.255.255.224
  no neighbor 2005:DEAD:BEEF:124::1 activate
  neighbor 172.16.124.1 activate
 exit-address-family
 !
 address-family ipv6
```

```
  network 2001:DEAD:BEEF:147::/96
  neighbor 2005:DEAD:BEEF:124::1 activate
 exit-address-family
```

Verification

You can quickly verify that your peering is up by running the command **show ip bgp** or **show bgp ipv6 unicast** on a downstream router. If the required routes have not been successfully advertised, you will need to analyze the process and separately troubleshoot the peering and route origination configurations. If your network statement is working, then you should see the advertised network in the output of **show ip bgp** with a 0.0.0.0 next hop or **show bgp ipv6 unicast** with a :: next hop on the originating router.

If you have done only the configuration mentioned so far, you are likely to get these results:

Lines have been removed from the following output for brevity:

```
R1#show ip bgp

     Network          Next Hop            Metric LocPrf Weight Path
 *>  147.10.1.128/27  172.16.124.2             0             0 200 i
 *   157.10.1.208/28  172.16.45.5              0    100      0 (65002) 500 i
R1#show bgp ipv6 unicast

     Network          Next Hop            Metric LocPrf Weight Path
 *>  2001:DEAD:BEEF:147::/96
                       2005:DEAD:BEEF:124::2
                                                0             0 200 i
 *   2001:DEAD:BEEF:157::/96
                       2005:DEAD:BEEF:45::5
                                                0    100      0 (65002) 500 i
R2#show ip bgp

     Network          Next Hop            Metric LocPrf Weight Path
 *>  147.10.1.128/27  0.0.0.0                  0         32768 i
R2#show bgp ipv6 unicast

     Network          Next Hop            Metric LocPrf Weight Path
 *>  2001:DEAD:BEEF:147::/96
                       ::                       0         32768 i
R3#show ip bgp

     Network          Next Hop            Metric LocPrf Weight Path
 * i 157.10.1.208/28  172.16.45.5              0    100      0 500 i
R3#show bgp ipv6 unicast

     Network          Next Hop            Metric LocPrf Weight Path
  * i 2001:DEAD:BEEF:157::/96
                       2005:DEAD:BEEF:45::5
                                                0    100      0 500 i
R4#show ip bgp

     Network          Next Hop            Metric LocPrf Weight Path
 *   147.10.1.128/27  172.16.124.2             0    100      0 (65001) 200 i
 *>  157.10.1.208/28  172.16.45.5              0             0 500 i
R4#show bgp ipv6 unicast

     Network          Next Hop            Metric LocPrf Weight Path
 *   2001:DEAD:BEEF:147::/96
                       2005:DEAD:BEEF:124::2
                                                0    100      0 (65001) 200 i
 *>  2001:DEAD:BEEF:157::/96
                       2005:DEAD:BEEF:45::5
                                                0             0 500 i
R5#show ip bgp

     Network          Next Hop            Metric LocPrf Weight Path
 *>  157.10.1.208/28  0.0.0.0                  0         32768 i
R5#show bgp ipv6 unicast

     Network          Next Hop            Metric LocPrf Weight Path
 *>  2001:DEAD:BEEF:157::/96
                       ::                       0         32768 i
R7#show ip bgp

     Network          Next Hop            Metric LocPrf Weight Path
 * i 147.10.1.128/27  172.16.120.1             0    100      0 200 i
R7#show bgp ipv6 unicast

     Network          Next Hop            Metric LocPrf Weight Path
 * i 2001:DEAD:BEEF:147::/96
                       2005:DEAD:BEEF:124::2
                                                0    100      0 200 i
```

Best indicators (>) are present for 147.10.1.128/27 and 2001:DEAD:BEEF:147::/96 only on R2 and R1. Best indicators are present for 157.10.1.208/28 and 2001:DEAD:BEEF:157::/96 only on R5 and R4. The problem in each case is next-hop reachability. For example, R4 has the route 147.10.1.128 with a next hop of 172.16.124.2 in its BGP routing table. However, the 172.16.124.0 network is not in its IP routing table. R1 has the route 157.10.1.208/24 with a next hop of 172.16.45.5 in its BGP routing table, but the next-hop network is not in its IP routing table.

To avoid advertising invalid routes, BGP will not use or advertise a route unless the next hop is in the IP routing table. The scenario restrictions ruled out adding these border networks into the AS 100 IGPs. The obvious alternative is to use the **neighbor next-hop-self** command so that the border routers, R1 and R4, change the next hop to their internal peering addresses.

This can be accomplished as follows:

```
R1#show run | section bgp
!
router bgp 65001
 bgp router-id 1.1.1.1
 bgp log-neighbor-changes
 bgp confederation identifier 100
 bgp confederation peers 65002
 neighbor 2005:DEAD:BEEF:43::4 remote-as 65002
 neighbor 2005:DEAD:BEEF:43::4 ebgp-multihop 2
 neighbor 2005:DEAD:BEEF:120::7 remote-as 65001
 neighbor 2005:DEAD:BEEF:124::2 remote-as 200
 neighbor 172.16.43.4 remote-as 65002
 neighbor 172.16.43.4 ebgp-multihop 2
 neighbor 172.16.120.7 remote-as 65001
 neighbor 172.16.124.2 remote-as 200
 !
 address-family ipv4
  synchronization
  no neighbor 2005:DEAD:BEEF:43::4 activate
  no neighbor 2005:DEAD:BEEF:120::7 activate
  no neighbor 2005:DEAD:BEEF:124::2 activate
  neighbor 172.16.43.4 activate
  neighbor 172.16.43.4 next-hop-self
  neighbor 172.16.120.7 activate
  neighbor 172.16.120.7 next-hop-self
  neighbor 172.16.124.2 activate
 exit-address-family
 !
 address-family ipv6
  neighbor 2005:DEAD:BEEF:43::4 activate
  neighbor 2005:DEAD:BEEF:43::4 next-hop-self
  neighbor 2005:DEAD:BEEF:120::7 activate
  neighbor 2005:DEAD:BEEF:120::7 next-hop-self
  neighbor 2005:DEAD:BEEF:124::2 activate
 exit-address-family
R4#show run | section bgp
!
router bgp 65002
 bgp router-id 4.4.4.4
 bgp log-neighbor-changes
 bgp confederation identifier 100
 bgp confederation peers 65001
 neighbor 2005:DEAD:BEEF:13::1 remote-as 65001
 neighbor 2005:DEAD:BEEF:13::1 ebgp-multihop 2
 neighbor 2005:DEAD:BEEF:43::3 remote-as 65002
 neighbor 2005:DEAD:BEEF:45::5 remote-as 500
 neighbor 172.16.13.1 remote-as 65001
 neighbor 172.16.13.1 ebgp-multihop 2
 neighbor 172.16.43.3 remote-as 65002
 neighbor 172.16.45.5 remote-as 500
 !
 address-family ipv4
  no neighbor 2005:DEAD:BEEF:13::1 activate
  no neighbor 2005:DEAD:BEEF:43::3 activate
  no neighbor 2005:DEAD:BEEF:45::5 activate
  neighbor 172.16.13.1 activate
  neighbor 172.16.13.1 next-hop-self
  neighbor 172.16.43.3 activate
  neighbor 172.16.43.3 next-hop-self
  neighbor 172.16.45.5 activate
 exit-address-family
 !
 address-family ipv6
  neighbor 2005:DEAD:BEEF:13::1 activate
  neighbor 2005:DEAD:BEEF:13::1 next-hop-self
  neighbor 2005:DEAD:BEEF:43::3 activate
  neighbor 2005:DEAD:BEEF:43::3 next-hop-self
  neighbor 2005:DEAD:BEEF:45::5 activate
 exit-address-family
```

This configuration will provide reachable next hops for both prefixes on all routes; all of the routes will have best indicators in the local RIBs and 147.10.1.128/27 and 157.10.1.208/24 will be in the routing tables of all of the BGP routers. You might conclude, therefore, that a ping from R5 to 147.10.1.145, sourced from 157.10.1.211, would be successful. However, you are likely to find that it is not. If you try a similar **source trace** at this point, you are likely to see this result:

```
R5#trace ip 147.10.1.145 source 157.10.1.211

Type escape sequence to abort.
Tracing the route to 147.10.1.145

  1 172.16.45.4 52 msec 100 msec 44 msec
  2 172.16.43.3 36 msec 96 msec 80 msec
  3 172.16.43.4 72 msec 120 msec 164 msec
  4 172.16.43.3 140 msec 160 msec 148 msec
  5 172.16.43.4 132 msec 180 msec 92 msec
  6 172.16.43.3 172 msec 120 msec 248 msec
  7 172.16.43.4 260 msec 180 msec 220 msec
...

R5#traceroute ipv6
Target IPv6 address: 2001:DEAD:BEEF:147::145
Source address: 2001:dead:beef:157::211
Insert source routing header? [no]:
Numeric display? [no]:
Timeout in seconds [3]:
Probe count [3]:
Minimum Time to Live [1]:
Maximum Time to Live [30]:
Priority [0]:
```

```
Port Number [0]:
Type escape sequence to abort.
Tracing the route to 2001:DEAD:BEEF:147::145

  1 2005:DEAD:BEEF:45::4 9 msec 9 msec 8 msec
  2 2005:DEAD:BEEF:43::3 17 msec 17 msec 11 msec
  3 2005:DEAD:BEEF:43::4 17 msec 17 msec 17 msec
  4 2005:DEAD:BEEF:43::3 26 msec 24 msec 25 msec
  5 2005:DEAD:BEEF:43::4 25 msec 25 msec 24 msec
  6 2005:DEAD:BEEF:43::3 34 msec 33 msec 34 msec
  7 2005:DEAD:BEEF:43::4 33 msec 34 msec 33 msec
  8 2005:DEAD:BEEF:43::3 41 msec 42 msec 41 msec
  9 2005:DEAD:BEEF:43::4 43 msec 40 msec 42 msec
 10 2005:DEAD:BEEF:43::3 49 msec 50 msec 51 msec
 11 2005:DEAD:BEEF:43::4 50 msec 49 msec 50 msec
 12 2005:DEAD:BEEF:43::3 59 msec 57 msec 54 msec
 13 2005:DEAD:BEEF:43::4 58 msec 55 msec 58 msec
 14 2005:DEAD:BEEF:43::3 67 msec 67 msec 66 msec
 15 2005:DEAD:BEEF:43::4 67 msec 67 msec 66 msec
 16 2005:DEAD:BEEF:43::3 75 msec 75 msec 75 msec
 17 2005:DEAD:BEEF:43::4 75 msec 74 msec 71 msec
 18 2005:DEAD:BEEF:43::3 83 msec 79 msec 84 msec
 19 2005:DEAD:BEEF:43::4 83 msec 83 msec 84 msec
 20 2005:DEAD:BEEF:43::3 87 msec 92 msec 92 msec
 21 2005:DEAD:BEEF:43::4 91 msec 92 msec 88 msec
 22 2005:DEAD:BEEF:43::3 99 msec 111 msec 98 msec
 23 2005:DEAD:BEEF:43::4 99 msec 100 msec 100 msec
 24 2005:DEAD:BEEF:43::3 108 msec 108 msec 109 msec
 25 2005:DEAD:BEEF:43::4 108 msec 108 msec 105 msec
 26 2005:DEAD:BEEF:43::3 117 msec 116 msec 112 msec
 27 2005:DEAD:BEEF:43::4 117 msec 117 msec 112 msec
 28 2005:DEAD:BEEF:43::3 125 msec 125 msec 121 msec
 29 2005:DEAD:BEEF:43::4 126 msec 124 msec 121 msec
 30 2005:DEAD:BEEF:43::3 129 msec 133 msec 134 msec
Destination not found inside max hopcount diameter.
```

There is a routing loop between R4 and R3. When the packet that is destined to 147.10.1.145 arrives at R4 from R5, it matches the BGP route with a next hop of 172.16.13.1. But R4 is not directly connected to the next-hop network and must do a recursive lookup. R4 finds the next hop toward 172.16.13.1 is 172.16.43.3 and sends the packet to R3. On R3, however, the BGP next hop toward 172.16.13.1 is 172.16.43.4, so the packet is sent back to R4, forming the loop.

The command **neighbor 172.16.43.3 next-hop-self** that was entered on R4 is the immediate cause of this loop. The command was entered to provide a reachable next hop for 157.10.1.208/28, but the command changes the next hop for all routes that are advertised to a neighbor. The next hop for 147.10.1.128/27 was changed as well, causing the routing loop. The root issue is the conflict between the logical exit from AS 65002 and the physical exit. BGP does not anticipate direct connections between nonborder routers in different autonomous systems.

```
R3#show ip bgp
     Network          Next Hop            Metric LocPrf Weight Path
 *>i 147.10.1.128/27  172.16.43.4              0    100      0 (65001) 200 i
 *>i 157.10.1.208/28  172.16.43.4              0    100      0 500 i
R3#show bgp ipv6 unicast
     Network          Next Hop            Metric LocPrf Weight Path
 *>i 2001:DEAD:BEEF:147::/96
                       2005:DEAD:BEEF:43::4
                                                 0    100      0 (65001) 200 i
 *>i 2001:DEAD:BEEF:157::/96
                       2005:DEAD:BEEF:43::4
                                                 0    100      0 500 i
```

There are a number of ways to fix this loop, but the scenario restrictions do not allow advertising the border routes and changing the peering relationships. One approach is to replace the **neighbor 172.16.43.3 next-hop-self** command on R3 with a route map that directly changes the next hop for selected prefixes. The next issue is what to match on. If you match on a route, what will happen if you add more routes in the future (perhaps in Lab 4-3)? You would have to return to this fix and add more routes to your prefix list. If you look at the path that the routes in question go through, you will see that you can match on the AS path. Here is one example:

```
R3#show run | begin bgp
!
router bgp 65002
 bgp router-id 3.3.3.3
 bgp log-neighbor-changes
 bgp confederation identifier 100
 neighbor 2005:DEAD:BEEF:43::4 remote-as 65002
 neighbor 172.16.43.4 remote-as 65002
 !
 address-family ipv4
  no neighbor 2005:DEAD:BEEF:43::4 activate
  neighbor 172.16.43.4 activate
  neighbor 172.16.43.4 route-map ThruR1IPv4 in
 exit-address-family
 !
 address-family ipv6
  neighbor 2005:DEAD:BEEF:43::4 activate
  neighbor 2005:DEAD:BEEF:43::4 route-map ThruR1IPv6 in
 exit-address-family
!
ip as-path access-list 1 permit _\(65001\)_
!
route-map ThruR1IPv6 permit 10
 match as-path 1
 set ipv6 next-hop 2005:DEAD:BEEF:13::1
!
route-map ThruR1IPv6 permit 20
!
route-map ThruR1IPv4 permit 10
 match as-path 1
 set ip next-hop 172.16.13.1
!
route-map ThruR1IPv4 permit 20
!
```

After applying this route map inbound from neighbor R4 and performing a route refresh (**clear ip bgp * in** and **clear bgp ipv6 unicast * in**), R3 has the correct next hops, and the source ping and trace from R5 to R2 are successful. This solution takes advantage of the fact that, unlike an IGP, the BGP next hop is simply an attribute in an update message, and BGP attributes can be manipulated with route maps.

```
R3#show ip bgp
     Network          Next Hop            Metric LocPrf Weight Path
 *>i 147.10.1.128/27  172.16.13.1              0    100      0 (65001) 200 i
 *>i 157.10.1.208/28  172.16.43.4              0    100      0 500 i
R3#show bgp ipv6 unicast
     Network          Next Hop            Metric LocPrf Weight Path
 *>i 2001:DEAD:BEEF:147::/96
                       2005:DEAD:BEEF:13::1
                                                0    100      0 (65001) 200 i
 *>i 2001:DEAD:BEEF:157::/96
                       2005:DEAD:BEEF:43::4
                                                0    100      0 500 i
R5#traceroute 147.10.1.145 source loopback 157
Type escape sequence to abort.
Tracing the route to 147.10.1.145
VRF info: (vrf in name/id, vrf out name/id)
  1 172.16.45.4 9 msec 8 msec 9 msec
  2 172.16.43.3 17 msec 17 msec 14 msec
  3 172.16.13.1 25 msec 25 msec 25 msec
  4 172.16.124.2 24 msec 25 msec 26 msec
R5#traceroute ipv6
Target IPv6 address: 2001:DEAD:BEEF:147::145
Source address: 2001:dead:beef:157::211
```

```
Insert source routing header? [no]:
Numeric display? [no]:
Timeout in seconds [3]:
Probe count [3]:
Minimum Time to Live [1]:
Maximum Time to Live [30]:
Priority [0]:
Port Number [0]:
Type escape sequence to abort.
Tracing the route to 2001:DEAD:BEEF:147::145

  1 2005:DEAD:BEEF:45::4 3 msec 9 msec 8 msec
  2 2005:DEAD:BEEF:43::3 17 msec 17 msec 17 msec
  3 2005:DEAD:BEEF:13::1 25 msec 24 msec 25 msec
  4 2005:DEAD:BEEF:124::2 26 msec 25 msec 24 msec
R5#ping 147.10.1.145 source loopback 157
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 147.10.1.145, timeout is 2 seconds:
Packet sent with a source address of 157.10.1.211
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 25/25/26 ms
R5#ping 2001:dead:beef:147::145 source loopback 157
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DEAD:BEEF:147::145, timeout is 2
seconds:
Packet sent with a source address of 2001:DEAD:BEEF:157::211
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 25/25/26 ms
```

Task 3: Use the Synchronization Method in AS 65001

Step 1 Enable synchronization on R1 and R7.

Step 2 Implement a solution that will ensure that both subnets 147.10.1.128/27 and 157.10.1.208/28 have best indicators in the local RIBs of both routers.

Step 3 Confirm that R8 still has the routes within its RIB.

Solution to Task 3:

The rule of synchronization requires that to use or advertise a route that is learned from an internal peer, there must be a matching route in the forwarding table from a source other than BGP. You can determine if synchronization is enabled with the command **show ip protocols**. If it is not, the command is simply **synchronization** under the BGP process. If you have done no more configuration than what has been discussed so far, you are likely to see this result on R7:

```
R7#sh ip bgp
[lines removed for brevity]

   Network          Next Hop            Metric LocPrf Weight Path
* i147.10.1.128/27  172.16.120.1             0    100      0 200 i
* i157.10.1.208/28  172.16.120.1             0    100      0 (65002) 500 i

Note the lack of a best indicator (>) in front of each path. You can get some
indication as to the reason by examining the details for the prefix:
R7#show ip bgp 147.10.1.128
BGP routing table entry for 147.10.1.128/27, version 0
Paths: (1 available, no best path)
  Not advertised to any peer
  200
    172.16.120.1 from 172.16.120.1 (172.16.101.1)
      Origin IGP, metric 0, localpref 100, valid, internal, not synchronized
```

This output clearly indicates that the synchronization requirement has not been met; there is not a matching route in the forwarding table from a source other than BGP:

```
R7#show ip route 147.10.1.128
```

```
% Network not in table
```

The synchronization requirement is usually met by redistributing BGP into an IGP on the border router. This ensures that all internal routers will have paths to external routes through the IGP and can deliver to external addresses, whether or not they are running BGP. In the synchronization model of internetworking internal routers, use an IGP to move data through the AS, relying on redistribution. An alternative model, the no-synchronization model, is to run BGP on all internal routers, not redistribute BGP into an IGP, and disable the synchronization check. In this lab, the synchronization requirement can be met by redistributing BGP into OSPF on R1. After this is done, both routes have best indicators in the local RIB:

```
R7#show ip bgp
     Network          Next Hop            Metric LocPrf Weight Path
 r>i 147.10.1.128/27  172.16.120.1             0    100      0 200 i
 r>i 157.10.1.208/28  172.16.120.1             0    100      0 (65002) 500 i
R7#show ip bgp 147.10.1.128
BGP routing table entry for 147.10.1.128/27, version 2
Paths: (1 available, best #1, table default, RIB-failure(17))
  Advertised to update-groups:
     2
  Refresh Epoch 1
  200
    172.16.120.1 from 172.16.120.1 (1.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, confed-internal,
synchronized, best
      rx pathid: 0, tx pathid: 0x0
R7#show ip route 147.10.1.128
Routing entry for 147.10.1.128/27
  Known via "ospf 1", distance 110, metric 1
  Tag 200, type extern 2, forward metric 10
  Last update from 172.16.120.1 on Ethernet0/0.17, 02:40:53 ago
  Routing Descriptor Blocks:
  * 172.16.120.1, from 1.1.1.1, 02:40:53 ago, via Ethernet0/0.17
      Route metric is 1, traffic share count is 1
      Route tag 200
```

The “r” in front of each path indicates RIB failure. In this case, it only means that BGP cannot put the route into the forwarding table because there is a route with a lower administrative distance. This is expected for IBGP-learned routes when synchronization is enabled. The numbers 172.16.101.1 are the BGP and OSPF router IDs, respectively, on R1. R7 uses these numbers to compare the BGP and OSPF sources of the routes, and they must be the same in order for the route to be synchronized. You can find a discussion of this issue in RFC 1403.

```
R7#show ip bgp rib-failure
  Network            Next Hop                      RIB-failure   RIB-NH Matches
147.10.1.128/27    172.16.120.1        Higher admin distance              n/a
157.10.1.208/28    172.16.120.1        Higher admin distance              n/a
```

The purpose of R8 to this point is to also show that even though R7 does not need the BGP version of those routes, it does not keep it from advertising those routes to others.

```
R8#show ip bgp
     Network          Next Hop            Metric LocPrf Weight Path
 *>  147.10.1.128/27  172.16.78.7                            0 100 200 i
 *>  157.10.1.208/28  172.16.78.7                            0 100 500 i
```

Lab 4-3 Answer Key: BGP Filters and Path Determination

Task 1: Configure and Connect AS 600

Step 1 Configure AS 600 on R6 and peer it as shown in the Lab 4-1 Visual Objective diagram.

Step 2 Originate only these networks from AS 600 without using network statements or any static routes.

- Loopback200: 200.1.16.0/24 2001:DEAD:BEEF:200:16::/96

- Loopback201: 200.1.17.0/24 2001:DEAD:BEEF:200:17::/96
- Loopback202: 200.1.18.0/24 2001:DEAD:BEEF:200:18::/96
- Loopback203: 200.1.19.0/24 2001:DEAD:BEEF:200:18::/96

Step 3 Make AS 600 nontransit for any potential prefixes, and do not use any prefix filtering.

Solution to Task 1:

There are several ways to meet these requirements on R6. You cannot use prefix filtering, but you could filter on the AS path. You could create an AS path filter that matches on **^$,** which would match on local orginated routes, and use that as an outbound filter to R2 and R4. Another method would be to use the **no-export** community, as shown in the following example:

```
R2#show run | begin router bgp
router bgp 200
 bgp router-id 2.2.2.2
 bgp log-neighbor-changes
 neighbor 2005:DEAD:BEEF:26::6 remote-as 600
 neighbor 2005:DEAD:BEEF:124::1 remote-as 100
 neighbor 172.16.26.6 remote-as 600
 neighbor 172.16.124.1 remote-as 100
 !
 address-family ipv4
  network 147.10.1.128 mask 255.255.255.224
  no neighbor 2005:DEAD:BEEF:26::6 activate
  no neighbor 2005:DEAD:BEEF:124::1 activate
  neighbor 172.16.26.6 activate
  neighbor 172.16.26.6 send-community both
  neighbor 172.16.26.6 route-map NoTrans out
  neighbor 172.16.124.1 activate
 exit-address-family
 !
 address-family ipv6
  network 2001:DEAD:BEEF:147::/96
  neighbor 2005:DEAD:BEEF:26::6 activate
  neighbor 2005:DEAD:BEEF:26::6 send-community both
  neighbor 2005:DEAD:BEEF:26::6 route-map NoTrans out
  neighbor 2005:DEAD:BEEF:124::1 activate
 exit-address-family
!
route-map NoTrans permit 10
 set community no-export
R4#show run | begin router bgp
router bgp 65002
 bgp router-id 4.4.4.4
 bgp log-neighbor-changes
 bgp confederation identifier 100
 bgp confederation peers 65001
 neighbor 2005:DEAD:BEEF:13::1 remote-as 65001
 neighbor 2005:DEAD:BEEF:13::1 ebgp-multihop 2
 neighbor 2005:DEAD:BEEF:43::3 remote-as 65002
 neighbor 2005:DEAD:BEEF:45::5 remote-as 500
 neighbor 2005:DEAD:BEEF:46::6 remote-as 600
 neighbor 172.16.13.1 remote-as 65001
 neighbor 172.16.13.1 ebgp-multihop 2
 neighbor 172.16.43.3 remote-as 65002
 neighbor 172.16.45.5 remote-as 500
 neighbor 172.16.46.6 remote-as 600
 !
 address-family ipv4
  no neighbor 2005:DEAD:BEEF:13::1 activate
  no neighbor 2005:DEAD:BEEF:43::3 activate
  no neighbor 2005:DEAD:BEEF:45::5 activate
  no neighbor 2005:DEAD:BEEF:46::6 activate
  neighbor 172.16.13.1 activate
  neighbor 172.16.13.1 next-hop-self
  neighbor 172.16.43.3 activate
  neighbor 172.16.43.3 next-hop-self
  neighbor 172.16.45.5 activate
  neighbor 172.16.46.6 activate
  neighbor 172.16.46.6 send-community both
  neighbor 172.16.46.6 route-map NoTrans out
 exit-address-family
 !
 address-family ipv6
  neighbor 2005:DEAD:BEEF:13::1 activate
  neighbor 2005:DEAD:BEEF:13::1 next-hop-self
  neighbor 2005:DEAD:BEEF:43::3 activate
  neighbor 2005:DEAD:BEEF:43::3 next-hop-self
  neighbor 2005:DEAD:BEEF:45::5 activate
  neighbor 2005:DEAD:BEEF:46::6 activate
  neighbor 2005:DEAD:BEEF:46::6 send-community both
  neighbor 2005:DEAD:BEEF:46::6 route-map NoTrans out
 exit-address-family
!
route-map NoTrans permit 10
 set community no-export
!
R6#show run | begin router bgp
router bgp 600
 bgp router-id 6.6.6.6
 bgp log-neighbor-changes
 neighbor 2005:DEAD:BEEF:26::2 remote-as 200
 neighbor 2005:DEAD:BEEF:46::4 remote-as 100
 neighbor 172.16.26.2 remote-as 200
 neighbor 172.16.46.4 remote-as 100
 !
 address-family ipv4
  redistribute connected route-map MyLoops
  no neighbor 2005:DEAD:BEEF:26::2 activate
  no neighbor 2005:DEAD:BEEF:46::4 activate
  neighbor 172.16.26.2 activate
  neighbor 172.16.46.4 activate
 exit-address-family
 !
 address-family ipv6
  redistribute connected route-map MyLoops
  neighbor 2005:DEAD:BEEF:26::2 activate
  neighbor 2005:DEAD:BEEF:46::4 activate
 exit-address-family
!
route-map MyLoops permit 10
 match interface Loopback200 Loopback201 Loopback202 Loopback203
!
```

R6 is peered with AS 200 and AS 100. Again note that the subautonomous systems of the confederation are not seen externally. The loopback interfaces are originated into AS 600 using redistribution. A route map is used to ensure that only the specified loopbacks are included. Here is the resulting local RIB:

```
R6#show ip bgp
BGP table version is 7, 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, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *   147.10.1.128/27  172.16.46.4                            0 100 200 i
 *>                   172.16.26.2              0             0 200 i
 *   157.10.1.208/28  172.16.26.2                            0 200 100 500 i
 *>                   172.16.46.4                            0 100 500 i
 *>  200.1.16.0       0.0.0.0                  0         32768 ?
 *>  200.1.17.0       0.0.0.0                  0         32768 ?
 *>  200.1.18.0       0.0.0.0                  0         32768 ?
 *>  200.1.19.0       0.0.0.0                  0         32768 ?
R6#show bgp ipv6 unicast
BGP table version is 7, 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, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *   2001:DEAD:BEEF:147::/96
                       2005:DEAD:BEEF:46::4
                                                              0 100 200 i
 *>                   2005:DEAD:BEEF:26::2
                                                0             0 200 i
 *   2001:DEAD:BEEF:157::/96
                       2005:DEAD:BEEF:26::2
                                                              0 200 100 500 i
 *>                   2005:DEAD:BEEF:46::4
                                                              0 100 500 i
 *>  2001:DEAD:BEEF:200:16::/80
                       ::                       0         32768 ?
 *>  2001:DEAD:BEEF:200:17::/80
                       ::                       0         32768 ?
     Network          Next Hop            Metric LocPrf Weight Path
 *>  2001:DEAD:BEEF:200:18::/80
                       ::                       0         32768 ?
 *>  2001:DEAD:BEEF:200:19::/80
                       ::                       0         32768 ?
R6#show bgp ipv6 unicast 2001:DEAD:BEEF:147::/96
BGP routing table entry for 2001:DEAD:BEEF:147::/96, version 2
Paths: (2 available, best #2, table default, not advertised to EBGP peer)
  Not advertised to any peer
  Refresh Epoch 1
  100 200
    2005:DEAD:BEEF:46::4 (FE80::A8BB:CCFF:FE00:410) from 2005:DEAD:BEEF:46::4
(4.4.4.4)
      Origin IGP, localpref 100, valid, external
      Community: no-export
      rx pathid: 0, tx pathid: 0
  Refresh Epoch 1
  200
    2005:DEAD:BEEF:26::2 (FE80::A8BB:CCFF:FE00:200) from 2005:DEAD:BEEF:26::2
(2.2.2.2)
      Origin IGP, metric 0, localpref 100, valid, external, best
      Community: no-export
      rx pathid: 0, tx pathid: 0x0
```

Note that R6 has two paths to 147.10.1.128 and two paths to 157.10.1.208. In each case, the best indicator is in front of the path with the shortest AS path. The locally originated prefixes have empty AS paths and an origin code of "?," indicating that they were sourced into BGP through redistribution.

The requirement that AS 600 be nontransit for any potential prefixes means that no matter what routes R6 may learn from one external peer, they should not be advertised to another external peer. The wording of this requirement suggests a very general solution. The solution implemented here uses a no-export community. Note that even though community strings are transitive, they are not advertised by default, therefore you need to also add the **send-community** neighbor statement.

Task 2A: Filter Updates

Step 1 Allow R4 to announce only the 200.1.16.0/24 and 2001:DEAD:BEEF:200:16::/96 prefixes and the 200.1.17.0/24 and 2001:DEAD:BEEF:200:17::/96 prefixes to AS 500. Use a minimum number of statements to perform this task.

Step 2 Allow router R4 to announce routes that are originating from AS 200, as well.

Solution to Task 2A:

These requirements can be met in many ways, but it is important to notice that the first requirement is stated in terms of network address and the second in terms of AS path. It is also important to note that these two filters are applied to the same neighbor. One solution would be to apply a route map like this one, outbound on the peering from R4 to R5.

```
R4#show running-config | begin router bgp
router bgp 65002
 bgp router-id 4.4.4.4
 bgp log-neighbor-changes
 bgp confederation identifier 100
 bgp confederation peers 65001
 neighbor 2005:DEAD:BEEF:13::1 remote-as 65001
 neighbor 2005:DEAD:BEEF:13::1 ebgp-multihop 2
 neighbor 2005:DEAD:BEEF:43::3 remote-as 65002
 neighbor 2005:DEAD:BEEF:45::5 remote-as 500
 neighbor 2005:DEAD:BEEF:46::6 remote-as 600
 neighbor 172.16.13.1 remote-as 65001
 neighbor 172.16.13.1 ebgp-multihop 2
 neighbor 172.16.43.3 remote-as 65002
 neighbor 172.16.45.5 remote-as 500
 neighbor 172.16.46.6 remote-as 600
 !
 address-family ipv4
  no neighbor 2005:DEAD:BEEF:13::1 activate
  no neighbor 2005:DEAD:BEEF:43::3 activate
  no neighbor 2005:DEAD:BEEF:45::5 activate
  no neighbor 2005:DEAD:BEEF:46::6 activate
  neighbor 172.16.13.1 activate
  neighbor 172.16.13.1 next-hop-self
  neighbor 172.16.43.3 activate
  neighbor 172.16.43.3 next-hop-self
  neighbor 172.16.45.5 activate
  neighbor 172.16.45.5 route-map ToR5IPv4 out
  neighbor 172.16.46.6 activate
  neighbor 172.16.46.6 send-community both
  neighbor 172.16.46.6 route-map NoTrans out
 exit-address-family
 !
 address-family ipv6
  neighbor 2005:DEAD:BEEF:13::1 activate
  neighbor 2005:DEAD:BEEF:13::1 next-hop-self
  neighbor 2005:DEAD:BEEF:43::3 activate
  neighbor 2005:DEAD:BEEF:43::3 next-hop-self
  neighbor 2005:DEAD:BEEF:45::5 activate
  neighbor 2005:DEAD:BEEF:45::5 route-map ToR5IPv6 out
  neighbor 2005:DEAD:BEEF:46::6 activate
  neighbor 2005:DEAD:BEEF:46::6 send-community both
  neighbor 2005:DEAD:BEEF:46::6 route-map NoTrans out
 exit-address-family
!
ip as-path access-list 1 permit _200$
!
ip prefix-list R6LoopsIPv4 seq 5 permit 200.1.16.0/23 ge 24 le 24
!
ipv6 prefix-list R6LoopsIPv6 seq 5 permit 2001:DEAD:BEEF:200:16::/79 ge 80 le
80
route-map ToR5IPv6 permit 10
 match as-path 1
!
route-map ToR5IPv6 permit 20
 match ipv6 address prefix-list R6LoopsIPv6
!
route-map ToR5IPv4 permit 10
 match as-path 1
!
route-map ToR5IPv4 permit 20
 match ip address prefix-list R6LoopsIPv4
!
route-map NoTrans permit 10
 set community no-export
!
R5#show ip bgp
     Network          Next Hop            Metric LocPrf Weight Path
 *>  147.10.1.128/27  172.16.45.4                              0 100 200 i
 *>  157.10.1.208/28  0.0.0.0                  0         32768 i
 *>  200.1.16.0       172.16.45.4                              0 100 200 600 ?
 *>  200.1.17.0       172.16.45.4                              0 100 200 600 ?
R5#show bgp ipv6 unicast
     Network          Next Hop            Metric LocPrf Weight Path
 *>  2001:DEAD:BEEF:147::/96
                       2005:DEAD:BEEF:45::4
                                                               0 100 200 i
 *>  2001:DEAD:BEEF:157::/96
                       ::                       0         32768 i
 *>  2001:DEAD:BEEF:200:16::/80
                       2005:DEAD:BEEF:45::4
                                                               0 100 200 600 ?
 *>  2001:DEAD:BEEF:200:17::/80
                       2005:DEAD:BEEF:45::4
                                                               0 100 200 600 ?
```

The prefix list takes advantage of the fact that both of the permitted network addresses are identical in the first 23 bits for IPv4 and 79 bits for IPv6, and have the same /24 and /80 masks.

One might be tempted to meet the second filtering requirement by simply adding the R2 loopback addresses to the existing prefix lists, but this would indicate a misinterpretation of the task requirements. Instead, a more general solution is required; one that permits any prefix that is sourced from AS 200. The regular expression "_200$" will do this.

Task 2B: Filter Updates

Step 1 Add the following loopback interfaces to R8:

- Loopback 100: 208.1.16.8/24
- Loopback 101: 208.1.17.8/24
- Loopback 102: 208.1.18.8/24

- Loopback 103: 208.1.19.8/24

Step 2 Inject the new loopback interfaces on R8 into BGP with an origin code of "i".

Step 3 On R7, enable soft reconfiguration inbound relative to R8.

Step 4 Have R8 send only 208.1.16.0/24 and 208.1.17.0/24, but no filtering is allowed on R8. Verify that only those two routes are displayed on R7 for the received routes from R8.

Solution to Task 2B:

The point of this section is to add some additional routes that you can filter using outbound route filtering. The "i" in Step 2 is just asking you to use network statements to inject the routes. The other option could be to redistribute the routes, and then use a route map to change the origin code to IGP. The ORF indication is in Step 4, to filter on R8 without having any filters on R8. ORF is based on RFC 5291. Configure the prefix filter on one router and have BGP send the filter to another router to implement. The point of Step 3 is to prove where the filtering is being performed. Soft reconfiguration inbound stores a copy of the BGP table that is sent by that peer before the policy is applied; therefore, if R7 was actually doing the filtering, you would see the filtered routes within that neighbor's table, but not in the normal BGP table. If R8 is doing the filtering on behalf of R7, and then neither table should have those routes.

```
R7#show run | begin router bgp
router bgp 65001
 bgp router-id 7.7.7.7
 bgp log-neighbor-changes
 bgp confederation identifier 100
 neighbor 2005:DEAD:BEEF:120::1 remote-as 65001
 neighbor 172.16.78.8 remote-as 800
 neighbor 172.16.120.1 remote-as 65001
 !
 address-family ipv4
  synchronization
  no neighbor 2005:DEAD:BEEF:120::1 activate
  neighbor 172.16.78.8 activate
  neighbor 172.16.78.8 capability orf prefix-list send
  neighbor 172.16.78.8 soft-reconfiguration inbound
  neighbor 172.16.78.8 prefix-list FromR8 in
  neighbor 172.16.120.1 activate
  neighbor 172.16.120.1 next-hop-self
 exit-address-family
 !
 address-family ipv6
  neighbor 2005:DEAD:BEEF:120::1 activate
 exit-address-family
!
ip prefix-list FromR8 seq 5 permit 208.1.16.0/23 ge 24 le 24
R8#show run | begin router bgp
router bgp 800
 bgp router-id 8.8.8.8
 bgp log-neighbor-changes
 neighbor 2005:DEAD:BEEF:78::7 remote-as 100
 neighbor 172.16.78.7 remote-as 100
 !
 address-family ipv4
  network 208.1.16.0
  network 208.1.17.0
  network 208.1.18.0
  network 208.1.19.0
  no neighbor 2005:DEAD:BEEF:78::7 activate
  neighbor 172.16.78.7 activate
  neighbor 172.16.78.7 capability orf prefix-list receive
 exit-address-family
 !
 address-family ipv6
```

```
  neighbor 2005:DEAD:BEEF:78::7 activate
 exit-address-family
!
R7#show ip bgp
BGP table version is 17, local router ID is 7.7.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 r>i 147.10.1.128/27  172.16.120.1             0    100      0 200 i
 r>i 157.10.1.208/28  172.16.120.1             0    100      0 (65002) 500 i
 r>i 200.1.16.0       172.16.120.1             0    600      0 200 600 ?
 r>i 200.1.17.0       172.16.120.1             0    600      0 200 600 ?
 r>i 200.1.18.0       172.16.120.1             0    600      0 200 600 ?
 r>i 200.1.19.0       172.16.120.1             0    600      0 200 600 ?
 *>  208.1.16.0       172.16.78.8              0             0 800 i
 *>  208.1.17.0       172.16.78.8              0             0 800 i
R7#show ip bgp neighbors 172.16.78.8 received-routes
BGP table version is 17, local router ID is 7.7.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  208.1.16.0       172.16.78.8              0             0 800 i
 *>  208.1.17.0       172.16.78.8              0             0 800 i

Total number of prefixes 2
R8#show ip prefix-list
R8#
R8#show ip bgp neighbors 172.16.78.7 received prefix-filter
Address family: IPv4 Unicast
ip prefix-list 172.16.78.7: 1 entries
   seq 5 permit 208.1.16.0/23 ge 24 le 24
```

Task 3: Implement Preferred Paths

Step 1 Make sure that router R4 selects R1 for forwarding outbound traffic to prefixes that originate from AS 600.

Solution to Task 3:

Which path does R4 currently prefer?

```
R4#sh ip bgp
[lines removed for brevity]
   Network          Next Hop            Metric LocPrf Weight Path
*> 147.10.1.128/27  172.16.13.1              0    100      0 (65001) 200 i
*> 157.10.1.208/28  172.16.45.5              0             0 500 i
*>i200.1.16.0       172.16.46.6              0    100      0 600 ?
*>i200.1.17.0       172.16.46.6              0    100      0 600 ?
*>i200.1.18.0       172.16.46.6              0    100      0 600 ?
*>i200.1.19.0       172.16.46.6              0    100      0 600 ?
```

By default, R4 has only one route to the prefixes in AS 600: the path directly across the Ethernet. What happened to the path through R1?

Here is the current BGP table for R1:

```
R1#sh ip bgp
[lines removed for brevity]
```

```
*> 147.10.1.128/27  172.16.124.2             0             0 200 i
*> 157.10.1.208/28  172.16.43.4              0    100      0 (65002) 500 i
*> 200.1.16.0       172.16.43.4              0    100      0 (65002) 600 ?
*                   172.16.124.2                           0 200 600 ?
*> 200.1.17.0       172.16.43.4              0    100      0 (65002) 600 ?
*                   172.16.124.2                           0 200 600 ?
*> 200.1.18.0       172.16.43.4              0    100      0 (65002) 600 ?
*                   172.16.124.2                           0 200 600 ?
*> 200.1.19.0       172.16.43.4              0    100      0 (65002) 600 ?
*                   172.16.124.2                           0 200 600 ?
```

R1 has two paths to these prefixes: one through R2 and one through R4. R1 prefers the path through R4 because this path has the shorter AS path length (the sub-AS is not counted). Remember that BGP only advertises the one path that it prefers. Because R1 prefers the path to AS 600 through R4, R1 does not advertise the alternate path through R2 for these prefixes to R4.

Note To have R4 prefer the path through R1 to AS 600, you first need to have R1 prefer that path.

This preference can be accomplished in several ways. You can increase the weight for these paths as they come into R1, and then increase the weight as the prefixes arrive at R4. Alternatively, you can increase the AS path length as prefixes arrive at R4 from R6 by prepending the AS path. The solution that is demonstrated here increases the local preference on these prefixes as they arrive at R1, and it demonstrates that the local preference attribute is advertised between confederation peers. The selected commands increase the local preference to 200 for the routes that are sourced from AS 600. Other prefixes pass through the route map unchanged.

```
R1#show run | begin router bgp
router bgp 65001
 bgp router-id 1.1.1.1
 bgp log-neighbor-changes
 bgp confederation identifier 100
 bgp confederation peers 65002
 neighbor 2005:DEAD:BEEF:43::4 remote-as 65002
 neighbor 2005:DEAD:BEEF:43::4 ebgp-multihop 2
 neighbor 2005:DEAD:BEEF:120::7 remote-as 65001
 neighbor 2005:DEAD:BEEF:124::2 remote-as 200
 neighbor 172.16.43.4 remote-as 65002
 neighbor 172.16.43.4 ebgp-multihop 2
 neighbor 172.16.120.7 remote-as 65001
 neighbor 172.16.124.2 remote-as 200
 !
 address-family ipv4
  synchronization
  no neighbor 2005:DEAD:BEEF:43::4 activate
  no neighbor 2005:DEAD:BEEF:120::7 activate
  no neighbor 2005:DEAD:BEEF:124::2 activate
  neighbor 172.16.43.4 activate
  neighbor 172.16.43.4 next-hop-self
  neighbor 172.16.120.7 activate
  neighbor 172.16.120.7 next-hop-self
  neighbor 172.16.124.2 activate
  neighbor 172.16.124.2 route-map ToR6 in
 exit-address-family
 !
 address-family ipv6
  neighbor 2005:DEAD:BEEF:43::4 activate
  neighbor 2005:DEAD:BEEF:43::4 next-hop-self
  neighbor 2005:DEAD:BEEF:120::7 activate
  neighbor 2005:DEAD:BEEF:120::7 next-hop-self
  neighbor 2005:DEAD:BEEF:124::2 activate
  neighbor 2005:DEAD:BEEF:124::2 route-map ToR6 in
 exit-address-family
!
ip as-path access-list 1 permit _600$
```

```
!
route-map ToR6 permit 10
 match as-path 1
 set local-preference 600
!
route-map ToR6 permit 20
!
R4#show ip bgp
     Network          Next Hop            Metric LocPrf Weight Path
 *>  147.10.1.128/27  172.16.13.1              0    100      0 (65001) 200 i
 *>  157.10.1.208/28  172.16.45.5              0             0 500 i
 *>  200.1.16.0       172.16.13.1              0    600      0 (65001) 200 600
?
 *                    172.16.46.6              0             0 600 ?
 *>  200.1.17.0       172.16.13.1              0    600      0 (65001) 200 600
?
 *                    172.16.46.6              0             0 600 ?
 *>  200.1.18.0       172.16.13.1              0    600      0 (65001) 200 600
?
 *                    172.16.46.6              0             0 600 ?
 *>  200.1.19.0       172.16.13.1              0    600      0 (65001) 200 600
?
 *                    172.16.46.6              0             0 600 ?
 *>  208.1.16.0       172.16.13.1              0    100      0 (65001) 800 i
 *>  208.1.17.0       172.16.13.1              0    100      0 (65001) 800 i
R4#show bgp ipv6 unicast
     Network          Next Hop            Metric LocPrf Weight Path
 *>  2001:DEAD:BEEF:147::/96
                       2005:DEAD:BEEF:13::1
                                                0    100      0 (65001) 200 i
 *>  2001:DEAD:BEEF:157::/96
                       2005:DEAD:BEEF:45::5
                                                0             0 500 i
 *>  2001:DEAD:BEEF:200:16::/80
                       2005:DEAD:BEEF:13::1
                                                0    600      0 (65001) 200 600
?
 *                    2005:DEAD:BEEF:46::6
                                                0             0 600 ?
 *>  2001:DEAD:BEEF:200:17::/80
                       2005:DEAD:BEEF:13::1
     Network          Next Hop            Metric LocPrf Weight Path
                                                0    600      0 (65001) 200 600
?
 *                    2005:DEAD:BEEF:46::6
                                                0             0 600 ?
 *>  2001:DEAD:BEEF:200:18::/80
                       2005:DEAD:BEEF:13::1
                                                0    600      0 (65001) 200 600
?
 *                    2005:DEAD:BEEF:46::6
                                                0             0 600 ?
 *>  2001:DEAD:BEEF:200:19::/80
                       2005:DEAD:BEEF:13::1
                                                0    600      0 (65001) 200 600
?
 *                    2005:DEAD:BEEF:46::6
                                                0             0 600 ?
```

All routers in AS 100 now prefer the path to AS 600 through R1.

Note Whenever you change BGP policies, you must get the BGP routes resent in the direction of the policy change for it to take effect.

Note Depending on how you fixed the looping issue from Lab 4-2, you may have to go back and alter that for the routes coming from R6. If you matched on the AS path to resolve that issue, then there should be no additional changes needed.

Lab 5-1 Answer Key: Establishing Basic Connectivity for MPLS Layer 3 VPNs

Solution to Task 1, Steps 1–3:

This task readies the equipment for the labs to follow. The primary goal is to remove the existing EIGRP, OSPF, and BGP routing processes. This can be done fairly easily by disabling and then enabling the IP routing process on each device.

Simply enter the commands **no ip routing** and **ip routing** in global configuration mode. When this step is complete there should be no output from the command **show ip protocols**.

Also verify that the tunnel interface is shut down on R4. This is the interface with IP address 172.16.124.4.

Finally, make sure that you have same-subnet reachability on each of the links shown in the diagram. You should be able to ping from one device to another using the directly attached IP address.

The routers and switches will have other configurations left over from the earlier BGP lab. These route maps, access lists, prefix lists, and additional loopback interfaces are not used in this lab and should have no effect.

Lab 5-2 Answer Key: Configuring the MPLS Core

Task 1: Enable OSPF in the Core

Solution to Task 1:

Create OSPF process ID 1 on R1, R3, and R4. The process ID number is critical because you will be creating additional customer processes later in the lab.

It is a good practice to hard-code the OSPF router ID, especially when other OSPF processes will be created on the same router.

Add network statements for the required subnets. The OSPF network type should not be an issue, except for the loopback interfaces (they must be advertised with their configured mask, otherwise the labels between the routers will not match up and the LSP will fail to pass traffic); all of the external interfaces are point-to-point.

Verification

The command **show ip ospf interface brief** can be used to quickly verify the configuration. Here is the output for R3:

```
R3#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Lo103        1     0               172.16.103.1/24    1     P2P   0/0
Se1/1        1     0               172.16.43.3/24     64    P2P   1/1
Se1/0        1     0               172.16.13.3/24     64    P2P   1/1
```

This output clearly shows that the interfaces are in the correct process and area. It also verifies adjacent neighbors on each serial subinterface. Here are the OSPF routes on R1 showing the required result. For the purposes of this lab, host routes are acceptable for loopback networks.

```
R1#show ip route ospf
Codes: L - local, 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, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is not set

      172.16.0.0/16 is variably subnetted, 7 subnets, 2 masks
O        172.16.43.0/24 [110/128] via 172.16.13.3, 00:02:55, Serial1/0
O        172.16.103.0/24 [110/65] via 172.16.13.3, 00:02:55, Serial1/0
O        172.16.104.0/24 [110/129] via 172.16.13.3, 00:02:33, Serial1/0
```

You can verify connectivity between Loopback 101 and Loopback 104 using a source ping:

```
R1#ping 172.16.104.1 source loopback 101
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.104.1, timeout is 2 seconds:
Packet sent with a source address of 172.16.101.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/16/17 ms
```

Task 2: Implement MP-BGP Between R1 and R4

Solution to Task 2:

Here is the required configuration on R1 and R4:

```
R1#show run | section router bgp
router bgp 1
 bgp log-neighbor-changes
 neighbor 172.16.104.1 remote-as 1
 neighbor 172.16.104.1 update-source Loopback101
 !
 address-family vpnv4
  neighbor 172.16.104.1 activate
  neighbor 172.16.104.1 send-community extended
 exit-address-family
R4#show run | section router bgp
router bgp 1
 bgp log-neighbor-changes
 neighbor 172.16.101.1 remote-as 1
 neighbor 172.16.101.1 update-source Loopback104
 !
 address-family vpnv4
  neighbor 172.16.101.1 activate
  neighbor 172.16.101.1 send-community extended
 exit-address-family
```

The most critical commands include the **update-source**, **activate**, and **send-community extended** keywords.

Verification

The neighbor capabilities section of the **show ip bgp neighbor** command can be used to verify that the VPNv4 address family has been enabled:

```
R1#show ip bgp neighbor
BGP neighbor is 172.16.104.1,  remote AS 1, internal link
  BGP version 4, remote router ID 172.16.104.1
  BGP state = Established, up for 00:01:53
  Last read 00:00:52, last write 00:00:07, hold time is 180, keepalive interval
is 60 seconds
  Neighbor sessions:
    1 active, is not multisession capable (disabled)
  Neighbor capabilities:
    Route refresh: advertised and received(new)
    Four-octets ASN Capability: advertised and received
    Address family IPv4 Unicast: advertised and received
    Address family VPNv4 Unicast: advertised and received
    Enhanced Refresh Capability: advertised and received
    Multisession Capability:
    Stateful switchover support enabled: NO for session 1
  Message statistics:
```

Task 3: Enable MPLS in the Core

Solution to Task 3:

The LDP router ID must be a reachable IP address. It is a good practice to hard-code the LDP router ID using the command **mpls ldp router-id** *interface*. To enable MPLS in the core, enter the command **mpls ip** on each of the interfaces shown on the task diagram. They are interface S1/0 on R1 and R3 and interface S1/1 on R3 and R4.

Verification and Analysis

When LDP has been properly enabled, R3 should have two LDP neighbors, as you see here:

```
R3#show mpls ldp neighbor
*Sep 23 23:48:10.812: %SYS-5-CONFIG_I: Configured from console by console
R3#show mpls ldp neighbor
    Peer LDP Ident: 172.16.101.1:0; Local LDP Ident 172.16.103.1:0
        TCP connection: 172.16.101.1.646 - 172.16.103.1.16595
        State: Oper; Msgs sent/rcvd: 13/14; Downstream
        Up time: 00:02:11
        LDP discovery sources:
          Serial1/0, Src IP addr: 172.16.13.1
        Addresses bound to peer LDP Ident:
          10.0.0.1        172.16.13.1     1.1.1.1         172.16.101.1
    Peer LDP Ident: 172.16.104.1:0; Local LDP Ident 172.16.103.1:0
        TCP connection: 172.16.104.1.13017 - 172.16.103.1.646
        State: Oper; Msgs sent/rcvd: 11/11; Downstream
        Up time: 00:00:53
        LDP discovery sources:
          Serial1/1, Src IP addr: 172.16.43.4
        Addresses bound to peer LDP Ident:
          10.0.0.4        172.16.43.4     4.4.4.4         172.16.104.1
```

The output of **show mpls forwarding-table** on R1 and R4 should be similar to the following:

```
R1#show mpls forwarding-table
Local      Outgoing   Prefix            Bytes Label   Outgoing   Next Hop
Label      Label      or Tunnel Id      Switched      interface
16         Pop Label  172.16.43.0/24    0             Se1/0      point2point
19         Pop Label  172.16.103.0/24   0             Se1/0      point2point
20         19         172.16.104.0/24   0             Se1/0      point2point
```

R1 will use tag 19 when it sends traffic to IP address 172.16.104.1. You may find different values on your pod.

```
R4#show mpls forwarding-table
Local      Outgoing   Prefix            Bytes Label   Outgoing   Next Hop
Label      Label      or Tunnel Id      Switched      interface
16         Pop Label  172.16.13.0/24    0             Se1/1      point2point
17         18         172.16.101.0/24   0             Se1/1      point2point
18         Pop Label  172.16.103.0/24   0             Se1/1      point2point
```

In this case, R4 will use tag 18 when it sends traffic to 172.16.101.1.

The output of **debug mpls packets** on R3 should indicate that pings from R4 to 172.16.101.1 are switched, not routed. Here is some of the debug output for this pod:

```
R3#debug mpls packet
Packet debugging is on
R3#
*Sep 23 23:57:16.518: MPLS les: Se1/1: rx: Len 108 Stack {18 0 255} - ipv4 data
s:172.16.104.1 d:172.16.101.1 ttl:255 tos:0 prot:1
```

Here you see a packet arriving at R3 on interface S1/1, which would be the echo request from R4 to 172.16.101.1. It arrives with tag 18 (the stack {18 0 255} is interpreted as label CoS TTL, so in this case, label of 18, CoS of 0, TTL of 255), as we expected from the forwarding table displayed on R4. It leaves R3 with no label. This is the "penultimate hop popping" procedure; traffic arrives at the last hop router untagged.

Here you see an echo reply arriving on the R3 interface S1/0:

```
*Sep 23 23:57:16.510: MPLS les: Se1/0: rx: Len 108 Stack {19 0 255} - ipv4 data
s:172.16.101.1 d:172.16.104.1 ttl:255 tos:0 prot:1
```

The forwarding table on R1 indicates that traffic to 172.16.104.1 will be sent with a tag of 19, and that is what we see in the debug.

Every 60 seconds, R3 switches a packet between R1 and R4. Here is an example:

```
*Sep 24 00:00:49.145: MPLS les: Se1/0: rx: Len 67 Stack {19 6 255} - ipv4 data
s:172.16.101.1 d:172.16.104.1 ttl:255 tos:C0 prot:6
*Sep 24 00:00:49.362: MPLS les: Se1/1: rx: Len 48 Stack {18 6 255} - ipv4 data
s:172.16.104.1 d:172.16.101.1 ttl:255 tos:C0 prot:6
```

The time interval is marked with a ToS value of 0xC0 (when this hexadecimal value is converted to a decimal value, it can be viewed to possess the precedence value of 6.). This suggests that this traffic is a BGP hello exchange. Most routing protocol traffic is marked IP Precedence 6 in the IP header. MPLS copies this value into the experimental bits in the label. It is shown in this debug stack marking as the middle value.

To hide the routers that are involved within the MPLS network, it is a common configuration within the service provider world to disable TTL propagation. Use the **no mpls ip propagate-ttl** command in the global configuration on R1, R3, and R4 to hide the core of your MPLS cloud.

Lab 5-3 Answer Key: Creating VPNs and Enabling VPN Routing

Task 1: Create VRFs for VPNA and VPNB on the PE Routers

Solution to Task 1:

Here is the expected configuration for R1.

```
R1#show run | begin vrf
ip vrf VPNA
 rd 1:100
 route-target export 1:11
 route-target import 1:11
!
ip vrf VPNB
 rd 1:200
 route-target export 1:22
 route-target import 1:22
!
```

In addition, interface Tunnel 0 is configured with the command **ip vrf forwarding VPNA**, and interface E0/0.17 is configured with the command **ip vrf forwarding VPNB**. You will need to re-enter the IP addresses and masks on these interfaces. Here is the expected output of the command **show ip vrf** for R1:

```
R1#show ip vrf
  Name                             Default RD          Interfaces
  VPNA                             1:100               Tu0
  VPNB                             1:200               Et0/0.17
```

The configuration on R4 is similar.

Note You can find complete finished configurations for the lab in the Mentor Guide.

Task 2: Enable OSPF for PE-to-CE Routing

Solution to Task 2:

Success in this task requires close attention to detail. VPNA uses OSPF process ID 2 at both sites, but VPNB uses OSPF process ID 3 in Site 1 and ID 30 in Site 2. VPNA uses OSPF Area 10, but VPNB uses OSPF Area 0. Here is the expected OSPF configuration for R1:

```
R1#show run | section router ospf
router ospf 2 vrf VPNA
 router-id 172.16.124.1
 network 172.16.124.0 0.0.0.255 area 10
!
router ospf 3 vrf VPNB
 router-id 172.16.120.1
 network 172.16.120.0 0.0.0.255 area 0
!
router ospf 1
 router-id 172.16.101.1
 network 172.16.13.0 0.0.0.255 area 0
 network 172.16.101.0 0.0.0.255 area 0
```

The output of **show ip ospf interfaces brief** makes it easy to verify that the processes, areas, and interfaces are properly matched and the expected neighbor adjacencies have formed.

```
R1#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Lo101        1     0               172.16.101.1/24    1     P2P   0/0
Se1/0        1     0               172.16.13.1/24     64    P2P   1/1
Et0/0.17     3     0               172.16.120.1/24    10    DR    1/1
Tu0          2     10              172.16.124.1/24    1000  P2P   1/1
```

Here is the expected configuration and **show** output for R4:

```
R4#show run | section router ospf
```

```
router ospf 2 vrf VPNA
 router-id 172.16.43.4
 network 172.16.46.0 0.0.0.255 area 10
!
router ospf 30 vrf VPNB
 router-id 172.16.45.4
 network 172.16.45.0 0.0.0.255 area 0
!
router ospf 1
 router-id 172.16.104.1
 network 172.16.43.0 0.0.0.255 area 0
 network 172.16.104.0 0.0.0.255 area 0
R4#show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Lo104        1     0               172.16.104.1/24    1     P2P   0/0
Se1/1        1     0               172.16.43.4/24     64    P2P   1/1
Se1/0        30    0               172.16.45.4/24     64    P2P   1/1
Et0/1        2     10              172.16.46.4/24     10    BDR   1/1
R1#show ip route vrf VPNA
      172.16.0.0/16 is variably subnetted, 3 subnets, 2 masks
O        172.16.102.1/32 [110/1001] via 172.16.124.2, 00:10:09, Tunnel0
C        172.16.124.0/24 is directly connected, Tunnel0
L        172.16.124.1/32 is directly connected, Tunnel0
R1#show ip route vrf VPNB
      172.16.0.0/16 is variably subnetted, 3 subnets, 2 masks
O        172.16.107.1/32 [110/11] via 172.16.120.7, 00:08:06,
Ethernet0/0.17
C        172.16.120.0/24 is directly connected, Ethernet0/0.17
L        172.16.120.1/32 is directly connected, Ethernet0/0.17
R4#show ip route vrf VPNA
      172.16.0.0/16 is variably subnetted, 3 subnets, 2 masks
C        172.16.46.0/24 is directly connected, Ethernet0/1
L        172.16.46.4/32 is directly connected, Ethernet0/1
O        172.16.106.1/32 [110/11] via 172.16.46.6, 00:00:45, Ethernet0/1
R4#show ip route vrf VPNB
      172.16.0.0/16 is variably subnetted, 4 subnets, 2 masks
C        172.16.45.0/24 is directly connected, Serial1/0
L        172.16.45.4/32 is directly connected, Serial1/0
C        172.16.45.5/32 is directly connected, Serial1/0
O        172.16.105.1/32 [110/65] via 172.16.45.5, 00:08:26, Serial1/0
```

Task 3: Mutually Redistribute OSPF and MP-BGP

Solution to Task 3:

Here is the required configuration on R1.

```
R1#show run | section router
router ospf 2 vrf VPNA
 router-id 172.16.124.1
 redistribute bgp 1 subnets
 network 172.16.124.0 0.0.0.255 area 10
router ospf 3 vrf VPNB
 router-id 172.16.120.1
 redistribute bgp 1 subnets
 network 172.16.120.0 0.0.0.255 area 0
router ospf 1
 router-id 172.16.101.1
 network 172.16.13.0 0.0.0.255 area 0
 network 172.16.101.0 0.0.0.255 area 0
router bgp 1
 bgp log-neighbor-changes
 neighbor 172.16.104.1 remote-as 1
 neighbor 172.16.104.1 update-source Loopback101
 !
 address-family vpnv4
  neighbor 172.16.104.1 activate
  neighbor 172.16.104.1 send-community extended
 exit-address-family
 !
 address-family ipv4 vrf VPNA
  redistribute ospf 2
 exit-address-family
 !
 address-family ipv4 vrf VPNB
  redistribute ospf 3
 exit-address-family
```

BGP AS 1 is redistributed into each VRF using the **subnets** keyword.

The redistribution into BGP has to be done in address family configuration mode for each VRF.

The configuration on R4 is similar. See the Mentor Guide for full configuration details. Here is the expected VRF BGP table on R1 when redistribution is complete on both sites:

```
R1#show ip bgp vpnv4 all
BGP table version is 15, local router ID is 172.16.101.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 1:100 (default for vrf VPNA)
 *>i 172.16.46.0/24   172.16.104.1             0    100      0 ?
 *>  172.16.102.1/32  172.16.124.2          1001         32768 ?
 *>i 172.16.106.1/32  172.16.104.1            11    100      0 ?
 *>  172.16.124.0/24  0.0.0.0                  0         32768 ?
Route Distinguisher: 1:200 (default for vrf VPNB)
 *>i 172.16.45.0/24   172.16.104.1             0    100      0 ?
 *>i 172.16.45.5/32   172.16.104.1             0    100      0 ?
 *>i 172.16.105.1/32  172.16.104.1            65    100      0 ?
 *>  172.16.107.1/32  172.16.120.7            11         32768 ?
 *>  172.16.120.0/24  0.0.0.0                  0         32768 ?
```

Notice that the table is separated on the basis of VPN. The routes from the far site are learned through IBGP and have a next hop of 172.16.104.1, which is the BGP peering address on R4.

Here is the expected routing table on R2:

```
R2#show ip route ospf
      172.16.0.0/16 is variably subnetted, 8 subnets, 2 masks
O IA     172.16.46.0/24 [110/1001] via 172.16.124.1, 00:05:47, Tunnel0
O IA     172.16.106.1/32 [110/1011] via 172.16.124.1, 00:05:47, Tunnel0
```

The routes from Site 2 are interarea routes because the domain IDs are the same in each site. Recall that, by default, the domain ID is equal to the process ID, which is 2 on both sites.

Notice that the OSPF cost from R2 to subnet 172.16.106.0/24 is 66. This includes the cost of 2 reported by MP-BGP to R1, plus the cost of 64 for the R1-R2 link. Compare this value to the cost you will see in Lab 5-4.

Here is the output of a **ping** from R2 to IP address 172.16.106.1 in VPNA Site 2.

```
R2#ping 172.16.106.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.106.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 17/17/18 ms
```

Here is the routing table on R7, which is in VPNB, Site 1.

```
R7#show ip route ospf
      172.16.0.0/16 is variably subnetted, 9 subnets, 2 masks
O E2     172.16.45.0/24 [110/1] via 172.16.120.1, 00:28:16, Ethernet0/0.17
O E2     172.16.45.5/32 [110/1] via 172.16.120.1, 00:28:16, Ethernet0/0.17
O E2     172.16.105.1/32 [110/65] via 172.16.120.1, 00:28:16, Ethernet0/0.17
```

Notice the routes from Site 2 are external. This is because each site uses a different process ID, and will have different domain IDs by default. You will configure domain IDs in a later step, but before doing so, examine the routing table and OSPF database on R1.

Here is the routing table for VPNB on R1:

```
R1#show ip route vrf VPNB | begin Gateway
Gateway of last resort is not set

      172.16.0.0/16 is variably subnetted, 6 subnets, 2 masks
B        172.16.45.0/24 [200/0] via 172.16.104.1, 00:29:20
B        172.16.45.5/32 [200/0] via 172.16.104.1, 00:29:20
B        172.16.105.1/32 [200/65] via 172.16.104.1, 00:29:20
O        172.16.107.1/32 [110/11] via 172.16.120.7, 00:44:32, Ethernet0/0.17
C        172.16.120.0/24 is directly connected, Ethernet0/0.17
L        172.16.120.1/32 is directly connected, Ethernet0/0.17
```

Notice that subnet 172.16.45.0/24 was placed in this table by BGP, with an administrative distance of 200. OSPF has this route in its database with an administrative distance of 110. Here is the LSA:

```
R1#show ip ospf database external 172.16.45.0

            OSPF Router with ID (172.16.101.1) (Process ID 1)

            OSPF Router with ID (172.16.120.1) (Process ID 3)

                Type-5 AS External Link States

  LS age: 1808
  Options: (No TOS-capability, DC, Downward)
  LS Type: AS External Link
  Link State ID: 172.16.45.0 (External Network Number )
  Advertising Router: 172.16.120.1
  LS Seq Number: 80000001
  Checksum: 0x95AB
  Length: 36
  Network Mask: /24
        Metric Type: 2 (Larger than any link state path)
        MTID: 0
        Metric: 1
        Forward Address: 0.0.0.0
        External Route Tag: 3489660929


            OSPF Router with ID (172.16.124.1) (Process ID 2)
```

The route tag is this value in binary: 11010000 00000000 00000000 00000001

The last 16 bits are the source BGP AS number, 1. Since this matches the local BGP AS number, OSPF has cleared the routing bit. This LSA cannot be used to place this route in the forwarding table, thus preventing a potential routing loop. Here is the same LSA in the OSPF database of R7:

```
R7#sh ip ospf database external 172.16.45.0

            OSPF Router with ID (172.16.107.1) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA in topology Base with MTID 0
  LS age: 1882
  Options: (No TOS-capability, DC, Downward)
  LS Type: AS External Link
  Link State ID: 172.16.45.0 (External Network Number )
  Advertising Router: 172.16.120.1
  LS Seq Number: 80000001
  Checksum: 0x95AB
  Length: 36
  Network Mask: /24
        Metric Type: 2 (Larger than any link state path)
        MTID: 0
        Metric: 1
        Forward Address: 0.0.0.0
        External Route Tag: 3489660929
```

R7 has the same route tag, but it is ignored on CE routers. The routing bit is set, indicating that this LSA can be used in the SPF algorithm.

To complete this section, configure matching domain IDs under the VPNB OSPF processes on R1 and R4. We used the command **domain-id 0.0.0.3**. With equal domain IDs, the VPN is treated as one large OSPF domain, and routes are advertised between sites as IA.

```
R4#sh run | s router ospf 30
router ospf 30 vrf VPNB
 router-id 172.16.45.4
 domain-id 0.0.0.3
 redistribute bgp 1 subnets
 network 172.16.45.0 0.0.0.255 area 0
```

Here you see the difference in the R7 routing table:

```
R7#show ip route ospf
      172.16.0.0/16 is variably subnetted, 9 subnets, 2 masks
O IA     172.16.45.0/24 [110/11] via 172.16.120.1, 00:00:10, Ethernet0/0.17
O IA     172.16.45.5/32 [110/11] via 172.16.120.1, 00:00:10, Ethernet0/0.17
O IA     172.16.105.1/32 [110/75] via 172.16.120.1, 00:00:10,
Ethernet0/0.17
```

Lab 5-4: Adding a Backup Link in VPNA

Task 1: Add Subnet 172.16.26.0/24 to OSPF

This is a straightforward task. Do not forget to change the OSPF costs on the interfaces:

```
R2#sh run | s router
router ospf 1
 network 172.16.26.0 0.0.0.255 area 10
 network 172.16.102.0 0.0.0.255 area 10
 network 172.16.124.0 0.0.0.255 area 10
R2#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
200.1.19.1        1   FULL/DR         00:00:36    172.16.26.6     Ethernet0/0.26
172.16.124.1      0   FULL/  -        00:00:39    172.16.124.1    Tunnel0
```

Task 2: Create a Sham Link Between PE Routers R1 and R4

If we add a back door into the mix, between R2 and R6, then they will prefer the backdoor path over the MPLS backbone due to the fact that the backdoor will be an intra-area routeand the MPLS path is an inter-area route.

```
R2#show ip route ospf

      172.16.0.0/16 is variably subnetted, 8 subnets, 2 masks
O        172.16.46.0/24 [110/80] via 172.16.26.6, 00:00:50, Ethernet0/0.26
O        172.16.106.1/32 [110/1501] via 172.16.26.6, 00:00:50, Ethernet0/0.26
```

To make the MPLS cloud look like the same area, you have to configure a sham link between the two PE routers. First you have to tie the process to a loopback interface, and the loopback has to have a /32 address on it. It has to be a part of the VRF that you are trying to extend, and it needs to be advertised within BGP. The **sham-link** command identifies the area that you are trying to extend. Here is the R1 configuration; R4 is similar:

```
R1#show run
!
interface Loopback0
 ip vrf forwarding VPNA
 ip address 1.1.1.1 255.255.255.255
!
router ospf 2 vrf VPNA
 router-id 172.16.124.1
 area 10 sham-link 1.1.1.1 4.4.4.4
 redistribute bgp 1 subnets
 network 172.16.124.0 0.0.0.255 area 10
!
router bgp 1
 bgp log-neighbor-changes
 neighbor 172.16.104.1 remote-as 1
 neighbor 172.16.104.1 update-source Loopback101
 !
 address-family vpnv4
  neighbor 172.16.104.1 activate
  neighbor 172.16.104.1 send-community extended
 exit-address-family
 !
 address-family ipv4 vrf VPNA
  network 1.1.1.1 mask 255.255.255.255
  redistribute ospf 2
 exit-address-family
R1#show ip ospf sham-links
Sham Link OSPF_SL0 to address 4.4.4.4 is up
Area 10 source address 1.1.1.1
  Run as demand circuit
  DoNotAge LSA allowed. Cost of using 1 State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40,
    Hello due in 00:00:01
    Adjacency State FULL (Hello suppressed)
    Index 2/2, retransmission queue length 0, number of retransmission 0
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 0, maximum is 0
    Last retransmission scan time is 0 msec, maximum is 0 msec
```

Notice that the sham link is up, but like with virtual links, you also have to look for the adjacency. If the adjacency is not full, then the link is not really up.

Now look at the routing table on R2:

```
R2#show ip route ospf
      1.0.0.0/32 is subnetted, 1 subnets
O E2     1.1.1.1 [110/1] via 172.16.124.1, 00:01:26, Tunnel0
      4.0.0.0/32 is subnetted, 1 subnets
O E2     4.4.4.4 [110/1] via 172.16.124.1, 00:01:26, Tunnel0
      172.16.0.0/16 is variably subnetted, 8 subnets, 2 masks
O        172.16.46.0/24 [110/12] via 172.16.124.1, 00:01:26, Tunnel0
O        172.16.106.1/32 [110/13] via 172.16.124.1, 00:01:26, Tunnel0
```

Lab 6-1 Answer Key: Establishing Basic Connectivity for Multicast

You need to complete two tasks to establish basic connectivity.

Task 1: Activate the Required Links

Use the **shutdown** and the **no shutdown** commands on the specified interfaces.

Task 2: Remove the MPLS Configuration and Implement the Required IGP

On each router, enter the command **no ip routing**, then the command **ip routing**. This procedure removes all existing routing protocol configuration. On R1 and R4, enter **no ip vrf VPNA** and **no ip vrf VPNB** and reapply the IP addresses. Add the IPv6 address to the loopback interfaces. The interesting part of Task 2 is in Step 3, the bonus challenge. You have a choice to set up two OSPF processes per protocol (IPv4 and IPv6) or to setup a single process for both. OSPFv3 now supports address families and can support both IPv4 and IPv6 under a single process. Here is R1 as an example:

```
R1#show run | begin 101
interface Loopback101
 ip address 172.16.101.1 255.255.255.0
 ipv6 address 2001:DEAD:BEEF:101::1/128
 ospfv3 network point-to-point
 ospfv3 1 ipv6 area 0
 ospfv3 1 ipv4 area 0
!
interface Tunnel0
 ip address 172.16.124.1 255.255.255.0
 no ip redirects
 ip mtu 1416
 ip nhrp map multicast dynamic
 ip nhrp network-id 1
 ipv6 address 2005:DEAD:BEEF:124::1/80
 ipv6 nhrp map multicast dynamic
 ipv6 nhrp network-id 1
 ospfv3 network point-to-multipoint
 ospfv3 cost 1
 ospfv3 1 ipv6 area 0
 ospfv3 1 ipv4 area 0
 tunnel source 10.0.0.1
 tunnel mode gre multipoint
!
interface Ethernet0/0.17
 encapsulation dot1Q 17
 ip address 172.16.120.1 255.255.255.0
 ipv6 address 2005:DEAD:BEEF:120::1/80
 ospfv3 1 ipv6 area 0
 ospfv3 1 ipv4 area 0
!
interface Serial1/0
 ip address 172.16.13.1 255.255.255.0
 ipv6 address 2005:DEAD:BEEF:13::1/80
 ospfv3 1 ipv6 area 0
 ospfv3 1 ipv4 area 0
 serial restart-delay 0
!
router ospfv3 1
 !
 address-family ipv4 unicast
 exit-address-family
 !
 address-family ipv6 unicast
 exit-address-family
!
```

Make sure that you set the network type to point-to-multipoint on the tunnel interface, and on R1, set the cost to 1.

```
interface Tunnel0
 ip address 172.16.124.1 255.255.255.0
 no ip redirects
 ip mtu 1416
 ip nhrp map multicast dynamic
 ip nhrp network-id 1
 ipv6 address 2005:DEAD:BEEF:124::1/80
 ipv6 nhrp map multicast dynamic
 ipv6 nhrp network-id 1
 ospfv3 network point-to-multipoint
 ospfv3 cost 1
 ospfv3 1 ipv6 area 0
 ospfv3 1 ipv4 area 0
 tunnel source 10.0.0.1
 tunnel mode gre multipoint
```

Lab 6-2 Answer Key: Configuring Dense Mode IPv4 Multicast Routing

When you complete this lab activity, you will have established IP multicast connectivity.

Task 1: Configure PIM Sparse-Dense Mode

Step 1 Enable multicast routing on on R1, R2, R3, R4, and R6 with the **ip multicast-routing** global command.

Step 2 Enable the PIM mode on the required interfaces with the **ip pim sparse-dense mode** command.

Note With the sparse-dense mode configuration option, the interface will use sparse mode if an RP is known for the group and dense mode if no RP is known.

Task 2: Configure and Verify Multicast Clients

For Step 1, you configure the loopback interfaces on R1, R2, R3, and R4 to play the part of a multicast client for this group by entering the **ip igmp join-group 239.255.1.1** command on the designated loopbacks.

For Step 2, at this point, we are operating in dense mode since there is no RP defined. The pings should look like this from R5:

```
R5#ping 239.255.1.1 sou s1/0 repeat 999
Type escape sequence to abort.
Sending 999, 100-byte ICMP Echos to 239.255.1.1, timeout is 2 seconds:
Packet sent with a source address of 172.16.45.5

Reply to request 0 from 172.16.104.1, 9 ms
Reply to request 0 from 172.16.106.1, 22 ms
Reply to request 0 from 172.16.102.1, 22 ms
Reply to request 0 from 172.16.101.1, 22 ms
Reply to request 0 from 172.16.103.1, 18 ms
Reply to request 1 from 172.16.104.1, 9 ms
Reply to request 1 from 172.16.106.1, 26 ms
Reply to request 1 from 172.16.102.1, 26 ms
Reply to request 1 from 172.16.101.1, 22 ms
Reply to request 1 from 172.16.103.1, 17 ms
Reply to request 2 from 172.16.104.1, 9 ms
Reply to request 2 from 172.16.106.1, 22 ms
Reply to request 2 from 172.16.102.1, 22 ms
Reply to request 2 from 172.16.101.1, 22 ms
Reply to request 2 from 172.16.103.1, 18 ms
Reply to request 3 from 172.16.104.1, 8 ms
Reply to request 3 from 172.16.106.1, 22 ms
Reply to request 3 from 172.16.102.1, 22 ms
Reply to request 3 from 172.16.101.1, 22 ms
Reply to request 3 from 172.16.103.1, 17 ms
```

A) Is R3 forwarding the traffic to R1? What is your evidence?

The commands **debug ip mpacket** and **show ip pim interface count** should indicate that traffic is being received on the 172.16.43.3 interface and is being forwarded out of the 172.16.13.3 interface.

```
R3#sh ip pim int count

For switching state use "show ip mfib interface"
Address          Interface                Mpackets In/Out
172.16.103.1     Loopback103              0/52
172.16.13.3      Serial1/0                0/52
172.16.43.3      Serial1/1                52/0
```

B) Is the traffic arriving at R1? If so, on what interface? How do you know? What is the RPF interface on R1? Why is R1 not responding to the multicast pings?

On R1, the **show ip pim interface count** command output displays that the multicast stream is arriving on S1/0 but is not being forwarded out of the router. The **debug ip mpacket** command output verifies that traffic is arriving on S1/0 and states “not RPF interface.” This is an appropriate result, because multicast traffic has to be received on the RPF interface. As the incoming interface entry in the output of the command **show ip mroute** shows, the RPF interface is Tunnel 0 for traffic that is sourced from 172.16.45.5.

```
R1#show ip pim int count

For switching state use "show ip mfib interface"
Address          Interface                Mpackets In/Out
172.16.124.1     Tunnel0                  0/0
172.16.101.1     Loopback101              0/134
172.16.120.1     Ethernet0/0.17           0/0
172.16.13.1      Serial1/0                134/0
```

C) What is the RPF interface on R1? Why is R1 not responding to the multicast pings from R5?

R1 does not respond the multicast pings because the traffic is not arriving on the RPF interface and so must be dropped.

```
R1#sh 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, E -
Extranet,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group,
       G - Received BGP C-Mroute, g - Sent BGP C-Mroute,
       Q - Received BGP S-A Route, q - Sent BGP S-A Route,
       V - RD & Vector, v - Vector
Outgoing interface flags: H - Hardware switched, A - Assert winner
```

```
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.255.1.1), 00:28:41/stopped, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial1/0, Forward/Sparse-Dense, 00:28:41/stopped
    Loopback101, Forward/Sparse-Dense, 00:28:41/stopped
    Tunnel0, Forward/Sparse-Dense, 00:28:41/stopped

(172.16.45.5, 239.255.1.1), 00:11:56/00:02:54, flags: LT
  Incoming interface: Tunnel0, RPF nbr 172.16.124.4
  Outgoing interface list:
    Serial1/0, Prune/Sparse-Dense, 00:00:10/00:02:49, A
    Loopback101, Forward/Sparse-Dense, 00:11:56/stopped

(*, 224.0.1.40), 00:40:41/00:02:21, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial1/0, Forward/Sparse-Dense, 00:31:05/stopped
    Tunnel0, Forward/Sparse-Dense, 00:40:41/stopped
R1#show ip route 172.16.45.5
Routing entry for 172.16.45.0/24
  Known via "ospfv3 1", distance 110, metric 192, type intra area
  Last update from 172.16.13.3 on Serial1/0, 00:33:21 ago
  Routing Descriptor Blocks:
  * 172.16.13.3, from 172.16.104.1, 00:33:21 ago, via Serial1/0
      Route metric is 192, traffic share count is 1
```

Step 3 There are two options to fix the multicast operations:

- Send the traffic so that it arrives on the default RPF interface.
- Change the RPF interface.

Since the task does not let you enable PIM on the 172.16.124.4 interface of R4, you must change the RPF interface on R1.

There are two options for changing the RPF interface:

- Change the unicast routing table.
- Override the unicast routing table.

You can change the unicast routing table by shutting down the 172.16.124.4 interface on R4, or by increasing the OSPF cost on the R1 tunnel interface so that the unicast table sees S1/0 as the shortest path to 172.16.45.5. However, the task scenario does not permit changing the unicast routing table.

You need to override the unicast routing table by implementing a multicast specific fix. You should configure a static **mroute** command locally on the router on which you want to change the RPF interface.

Note The static **mroute** command can specify a source host address or a source subnet, or it can be written as a default applying to all sources. Note that the static mroute does not reference the multicast group address.

You should configure the **mroute** command on R1. Your results should look similar to this:

```
R1(config)#ip mroute 172.16.45.0 255.255.255.0 172.16.13.3
```

You should verify multicast operations on R1 after configuring the static **mroute** command with **show** and **debug** commands. Your results should look similar to this:

```
R1#show ip mroute 239.255.1.1
```

```
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C - Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry, E - Extranet,
       X - Proxy Join Timer Running, A - Candidate for MSDP Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group,
       G - Received BGP C-Mroute, g - Sent BGP C-Mroute,
       Q - Received BGP S-A Route, q - Sent BGP S-A Route,
       V - RD & Vector, v - Vector
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.255.1.1), 02:00:08/stopped, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial1/0, Forward/Sparse-Dense, 02:00:08/stopped
    Loopback101, Forward/Sparse-Dense, 02:00:08/stopped
    Tunnel0, Forward/Sparse-Dense, 02:00:08/stopped

(172.16.45.5, 239.255.1.1), 00:00:05/00:02:54, flags: LT
  Incoming interface: Serial1/0, RPF nbr 172.16.13.3, Mroute
  Outgoing interface list:
    Tunnel0, Forward/Sparse-Dense, 00:00:05/stopped
    Loopback101, Forward/Sparse-Dense, 00:00:05/stopped
```

Note The (S,G) entry should now show S1/0 as the RPF interface with an mroute tag.

```
R1#show ip rpf 172.16.45.5
RPF information for ? (172.16.45.5)
  RPF interface: Serial1/0
  RPF neighbor: ? (172.16.13.3)
  RPF route/mask: 172.16.45.0/24
  RPF type: multicast (static)
  Doing distance-preferred lookups across tables
  RPF topology: ipv4 multicast base
R1#show ip mroute static
Mroute: 172.16.45.0/24, RPF neighbor: 172.16.13.3, distance: 1
```

Note The RPF interface should be Serial1/0, and the RPF neighbor should be 172.16.13.3.

```
R1#mtrace 172.16.45.5
Type escape sequence to abort.
Mtrace from 172.16.45.5 to 172.16.13.1 via RPF
From source (?) to destination (?)
Querying full reverse path...
 0  172.16.13.1
-1  172.16.13.1 ==> 172.16.13.1 PIM_MT  [172.16.45.0/24]
-2  172.16.13.3 ==> 172.16.43.3 PIM  [172.16.45.0/24]
-3  172.16.43.4 ==> 172.16.45.4 PIM_MT  [172.16.45.0/24]
-4  172.16.45.5
```

Note The full reverse path should be available.

This is what the pings from R5 should like at this point:

```
R5#ping 239.255.1.1 sou s1/0 repeat 999
Type escape sequence to abort.
Sending 999, 100-byte ICMP Echos to 239.255.1.1, timeout is 2 seconds:
Packet sent with a source address of 172.16.45.5

Reply to request 0 from 172.16.104.1, 9 ms
Reply to request 0 from 172.16.102.1, 26 ms
Reply to request 0 from 172.16.101.1, 26 ms
Reply to request 0 from 172.16.103.1, 18 ms
Reply to request 1 from 172.16.104.1, 9 ms
Reply to request 1 from 172.16.102.1, 26 ms
Reply to request 1 from 172.16.101.1, 26 ms
Reply to request 1 from 172.16.103.1, 17 ms
```

Task 3: Enhance Dense Mode Pruning

Step 1 This task encourages you to closely observe and enhance the pruning operation of PIM-DM. R2 will have the interface connected to R6 on its (S,G) OIL for group 239.255.1.1, because R6 is a PIM neighbor. But since there is no client configured on R6 or downstream of R6, the interface will be in pruned state. The timer values on the pruned entry here indicate that the interface was last flooded 9 seconds ago, and it will be reflooded in 2 minutes and 50 seconds.

```
R2#sh ip mroute 239.255.1.1
[Legend and (*,G) entry removed for brevity)

(172.16.45.5, 239.255.1.1), 00:00:08/00:02:59, flags: LT
  Incoming interface: Tunnel0, RPF nbr 172.16.124.1
  Outgoing interface list:
    Loopback102, Forward/Sparse-Dense, 00:00:09/00:00:00
    Ethernet0/0.26, Prune/Sparse-Dense, 00:00:09/00:02:50
```

Here is the entry for this group on R6. Note the P flag and the Null OIL.

```
R6#show ip mroute 239.255.1.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C - Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry, E - Extranet,
       X - Proxy Join Timer Running, A - Candidate for MSDP Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group,
       G - Received BGP C-Mroute, g - Sent BGP C-Mroute,
       Q - Received BGP S-A Route, q - Sent BGP S-A Route,
       V - RD & Vector, v - Vector
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.255.1.1), 02:12:41/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0/0.26, Forward/Sparse-Dense, 02:12:41/stopped

(172.16.45.5, 239.255.1.1), 00:02:41/00:00:18, flags: PT
  Incoming interface: Ethernet0/0.26, RPF nbr 172.16.26.2
  Outgoing interface list: Null
```

Step 2 Here is the result as the expire timer for this entry approaches and then reaches zero.

Note The lines of the **show** and **debug** outputs have been numbered for clarity.

```
1. R6#sh ip mroute | include 172.16.45.5
2.(172.16.45.5, 239.255.1.1), 00:02:57/00:00:02, flags: PT
3. R6#sh ip mroute | include 172.16.45.5
4. (172.16.45.5, 239.255.1.1), 00:02:58/00:00:01, flags: PT
5. R6#sh ip mroute | include 172.16.45.5
6. (172.16.45.5, 239.255.1.1), 00:02:59/00:00:00, flags: PT
7. R6#sh ip mroute | include 172.16.45.5
8. R6#sh ip mroute | include 172.16.45.5
9. R6#sh ip mroute | include 172.16.45.5
```

```
10. (172.16.45.5, 239.255.1.1), 00:00:01/00:02:58, flags: PT
11. *Feb  1 12:29:33.481: IP(0): s=172.16.45.5 (Ethernet0/0.26) d=239.255.1.1
id=1804, ttl=250, prot=1, len=114(100), mroute olist null
12. *Feb  1 12:29:33.485: PIM(0): Insert (172.16.45.5,239.255.1.1) prune in nbr
172.16.26.2's queue
13. *Feb  1 12:29:33.485: PIM(0): Building Join/Prune packet for nbr
172.16.26.2
14. *Feb  1 12:29:33.485: PIM(0): Adding v2 (172.16.45.5/32, 239.255.1.1) Prune
15. *Feb  1 12:29:33.489: PIM(0): Send v2 join/prune to 172.16.26.2
(Ethernet0/0.26)
```

At line 6, the expire timer reaches zero and the (S,G) entry is removed; note that lines 7 and 8 show no output. Line 10 shows that the (S,G) state has been refreshed with the expire timer at 2 minutes and 58 seconds. Line 11 is output from **debug ip mpacket** showing that the router has received the user mode traffic from R5.

Note Receiving the user mode traffic from R5 refreshed the (S,G) state.

Because there are no clients connected to or downstream from R6, the packet is not forwarded. Lines 12 through 15 show R6 generating and sending a prune message upstream to R2.

Step 3 You can enhance the dense mode pruning operation and avoid flooding unnecessary traffic by enabling the **state-refresh** feature. By default, PIM routers will *forward* state-refresh messages, but you must configure their *origination* on the interface facing the multicast source. To meet this requirement, you need to configure this command on the R4 interface that is associated with the IP address 172.16.45.4:

```
R4(config-if)#ip pim state-refresh origination-interval 30
```

Verification

The output here shows the effects of enabling the state-refresh feature. Both **debug ip pim** and **debug ip mpacket** have been enabled.

Note The lines of the **show** and **debug** outputs have been numbered for clarity.

```
1. R6#sh ip mroute | include 172.16.45.5
2. (172.16.45.5, 239.255.1.1), 00:02:32/00:02:31, flags: PT
3. R6#sh ip mroute | include 172.16.45.5
4. (172.16.45.5, 239.255.1.1), 00:02:32/00:02:30, flags: PT
5. R6#sh ip mroute | include 172.16.45.5
6. (172.16.45.5, 239.255.1.1), 00:02:33/00:02:59, flags: PT
7. R6#sh ip mroute | include 172.16.45.5
8. (172.16.45.5, 239.255.1.1), 00:02:33/00:02:58, flags: PT
9. R6#
10. *Feb  1 12:50:18.253: PIM(0): Received v2 State-Refresh on Ethernet0/0.26
from 172.16.26.2
11. *Feb  1 12:50:18.253: PIM(0): SR on iif from 172.16.26.2 orig 172.16.45.4
for (172.16.45.5,239.255.1.1)
12. *Feb  1 12:50:18.253: PIM(0): flags: prune-indicator
13. *Feb  1 12:50:18.253: PIM(0): Cached metric is [110/192]
14. *Feb  1 12:50:18.253: PIM(0): Keep RPF nbr 172.16.26.2
15. R6#sh ip mroute | include 172.16.45.5
16. (172.16.45.5, 239.255.1.1), 00:02:36/00:02:56, flags: PT
17. R6#sh ip mroute | include 172.16.45.5
18. (172.16.45.5, 239.255.1.1), 00:02:38/00:02:54, flags: PT
```

The expire timer counts down to just 2 minutes and 30 seconds before it is reset. There is no output from **debug ip mpacket**, because no actual multicast ping traffic was forwarded to R6. Instead, R6 receives an explicit message refreshing the state for the 172.16.45.5 source of group 239.255.1.1, as lines 10 through 14 show.

It was not necessary for R6 to send a prune upstream to R2. R2 will keep the interface pruned until it receives a graft message on the interface.

Lab 6-3 Answer Key: Configuring PIM-SM

When you complete this lab activity, you will have established IPv4 and IPv6 multicast connectivity.

Task 1: Enable IPv6 Multicast

Enable IPv6 multicast routing globally on R1, R2, R3, and R4 by using the **ipv6 multicast-routing** command. Also, on R1, R2, R3, and R4, join group FF05::255:1:1 on the loopback interfaces number 10*X*, with *X* as the router number (for example, R1 interface loopback 101) with the **ipv6 mld join-group ff05::255:1:1** command.

Task 2: Configure and Troubleshoot BSR

Step 1 In the previous lab, the command **ip pim state-refresh origination-interval 30** was entered on R4 and a static mroute was configured on R1. These configurations need to be removed to continue with this lab.

Step 2 To make R1 a candidate BSR and RP for multicast groups in the range 239.255.0.0/16 and FF05::255:0:0/96, you need to enter these commands in global configuration mode:

```
ip pim bsr-candidate Loopback101 0
ip pim rp-candidate Loopback101 group-list 1
access-list 1 permit 239.255.0.0 0.0.255.255
ipv6 pim bsr candidate bsr 2001:DEAD:BEEF:101::1
ipv6 pim bsr candidate rp 2001:DEAD:BEEF:101::1 group-list MyMulticast
ipv6 access-list MyMulticast
 permit ipv6 any FF05::255:0:0/96
```

Step 3 You can check to see if RP advertisement is successful with the command **show ip pim rp mapping**. Here is the result on R3:

```
 R3#show ip pim rp mapping
 PIM Group-to-RP Mappings

 Group(s) 239.255.0.0/16
   RP 172.16.101.1 (?), v2
     Info source: 172.16.101.1 (?), via bootstrap, priority 0, holdtime 150
          Uptime: 00:15:34, expires: 00:02:05
```

The entry has been up for 15 minutes and 34 seconds and will expire in 2 minutes and 5 seconds if it is not refreshed. The mapping agent sends an announcement every 60 seconds, by default. The "(?)" indicates a failure of name resolution.

When you run the command **show ip pim rp mapping** on R4, you will likely find that the BSR has failed to advertise the RP to this router. If you observe the output of **debug ip mpacket** on R4 for a few minutes, you will see this output:

```
R4#debug ip mpacket
IP multicast packets debugging is on
R4#
*Mar  4 06:32:55.900: IP(0): s=172.16.101.1 (Serial1/1) d=224.0.1.40 id=38833,
prot=17, len=52(48), RPF lookup failed for source
*Mar  4 06:32:55.904: IP(0): s=172.16.101.1 (Serial1/1) d=224.0.1.40 id=38833,
prot=17, len=52(48), not RPF interface
```

The RP multicast traffic from the BSR agent is arriving on S1/1, but this is not the RPF interface to 172.16.101.1. Therefore, R4 is dropping the traffic and failing to learn the RP mapping. According to the routing table, the shortest path interface to 172.16.101.1 is Tunnel 0. However, the RPF interface does not have PIM configured, so the **mtrace** indicates "No route."

```
 R4#mtrace 172.16.101.1
 Type escape sequence to abort.
 Mtrace from 172.16.101.1 to 172.16.124.4 via RPF
 From source (?) to destination (?)
```

```
Querying full reverse path...
 0  172.16.124.4
-1  172.16.124.4 ==> 0.0.0.0 None No route
```

This is the same as the RPF problem you saw in the dense mode section. You can fix this problem by putting PIM on the current RPF interface. You can also fix the RPF lookup problem by changing the unicast routing table, or by overriding the unicast routing table with a static mroute.

Step 4 Before repairing this issue, investigate the implications of an RP failure by starting a **ping** from R5. You are likely to get responses from R4 and R3, because R4 is distributing the traffic in dense mode. Notice the D flag in the output below:

```
R4#show ip mroute 239.255.1.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C - Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry, E - Extranet,
       X - Proxy Join Timer Running, A - Candidate for MSDP Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group,
       G - Received BGP C-Mroute, g - Sent BGP C-Mroute,
       Q - Received BGP S-A Route, q - Sent BGP S-A Route,
       V - RD & Vector, v - Vector
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.255.1.1), 00:00:02/stopped, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial1/1, Forward/Sparse-Dense, 00:00:02/stopped
    Loopback104, Forward/Sparse-Dense, 00:00:02/stopped

(172.16.45.5, 239.255.1.1), 00:00:01/00:02:58, flags: LT
  Incoming interface: Serial1/0, RPF nbr 0.0.0.0
  Outgoing interface list:
    Loopback104, Forward/Sparse-Dense, 00:00:01/stopped
    Serial1/1, Forward/Sparse-Dense, 00:00:01/stopped
```

If a router does not know the address of an RP for a group, it will distribute the traffic in dense mode. This problem is called dense mode fallback, and it can happen even when you have configured the router interfaces with the **ip pim sparse-mode** command. This is generally considered undesirable, so methods have been developed that prevent dense mode distribution when the router fails to learn an RP dynamically.

One method is to statically configure a local loopback address on each router as an RP of last resort. As of Cisco IOS Software Release 12.3(4)T, the command **no ip pim dm-fallback** can be used and is highly recommended. The "RP of last resort" technique can be implemented by entering this command on R4:

```
R4(config)#ip pim rp-address 172.16.104.1
```

Pings from R5 will be answered by R4 using sparse-mode distribution, but the traffic will go no further.

Step 5 To fix the RP, you can do one of these tasks:

1. Move the RP candidate and BSR, for example, to R4.
2. Change the routing table so that the shortest path to 172.16.101.1 is PIM-capable.
3. Override the routing table with a static mroute.
4. Enable PIM on the current shortest path to 172.16.101.1.

In this lab, you are asked to use option 4: configure PIM on the 172.16.124.4 interface of R4, which is on the current shortest path to 172.16.101.1. This will solve the RPF lookup issue, because R4 will receive the BSR messages on the RPF interface. By default, RP mappings that are learned via BSR override those that are statically configured, so R4 will use 172.16.101.1 as the RP for groups in the range 239.255.0.0/16, instead of the locally configured RP of last resort.

Here are the resulting mappings on R4:

```
R4#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 239.255.0.0/16
  RP 172.16.101.1 (?), v2
    Info source: 172.16.101.1 (?), via bootstrap, priority 0, holdtime 150
         Uptime: 00:01:38, expires: 00:01:52
Group(s): 224.0.0.0/4, Static
    RP: 172.16.104.1 (?)
```

Step 6 Test the IPv6 multicast from R7. If everything is working correctly, you should receive responses from R1, R2, R3, and R4:

```
R7#ping ff05::255:1:1 source e0/0.17 repeat 999
Output Interface: ethernet0/0.17
Type escape sequence to abort.
Sending 999, 100-byte ICMP Echos to FF05::255:1:1, timeout is 2 seconds:
Packet sent with a source address of 2005:DEAD:BEEF:120::7

Reply to request 0 received from 2001:DEAD:BEEF:101::1, 1 ms
Reply to request 0 received from 2001:DEAD:BEEF:102::1, 1 ms
Reply to request 0 received from 2001:DEAD:BEEF:103::1, 9 ms
Reply to request 0 received from 2001:DEAD:BEEF:104::1, 17 ms
Reply to request 1 received from 2001:DEAD:BEEF:101::1, 1 ms
Reply to request 1 received from 2001:DEAD:BEEF:104::1, 1 ms
Reply to request 1 received from 2001:DEAD:BEEF:102::1, 1 ms
Reply to request 1 received from 2001:DEAD:BEEF:103::1, 9 ms
```

Task 3: Enable Spoke-to-Spoke Multicast

Step 1 The most appropriate solution is to configure **ip pim nbma-mode** on the R1 tunnel interface. NBMA mode treats each PIM neighbor as if each PIM neighbor was on a separate interface. This enhancement enables spoke-to-spoke multicast by permitting the multipoint hub interface to be both the RPF interface and to appear on the OIL. Here is the configuration on R1:

```
R1(config)#int tunnel 0
R1(config-subif)#ip pim nbma-mode
PIM nbma-mode is not recommended for sparse-dense-mode
```

Depending on the version of code, you may or may not get the warning message alerting you that the command is not effective for traffic that is distributed in dense mode. Here is the resulting (S,G) state on R1 when you resume the multicast **ping** to group 239.255.1.1 from R5:

```
R1#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, E - Extranet,
       X - Proxy Join Timer Running, A - Candidate for MSDP Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group,
       G - Received BGP C-Mroute, g - Sent BGP C-Mroute,
       Q - Received BGP S-A Route, q - Sent BGP S-A Route,
       V - RD & Vector, v - Vector
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode
```

```
(*, 239.255.1.1), 05:01:53/00:03:27, RP 172.16.101.1, flags: SJCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Tunnel0, 172.16.124.4, Forward/Sparse-Dense, 00:00:57/00:02:32
    Tunnel0, 172.16.124.2, Forward/Sparse-Dense, 00:01:38/00:02:50
    Loopback101, Forward/Sparse-Dense, 05:01:53/00:02:16
    Serial1/0, Forward/Sparse-Dense, 02:05:21/00:03:27

(172.16.45.5, 239.255.1.1), 00:09:50/00:01:13, flags: LT
  Incoming interface: Tunnel0, RPF nbr 172.16.124.4
  Outgoing interface list:
    Tunnel0, 172.16.124.2, Forward/Sparse-Dense, 00:01:38/00:02:50
    Loopback101, Forward/Sparse-Dense, 00:09:50/00:02:16
```

Tunnel 0 is both the RPF interface and is on the OIL. In each case, the interface is paired with the IP address of a particular PIM neighbor. This configuration permits multicast traffic to arrive on a multipoint Frame Relay interface, and also to be forwarded out from the same interface. With this enhancement, R5 should receive replies from R2.

Task 4: Analyze and Control SPT Cutover

Step 1 When you first start a multicast ping from R5, you might notice that initial responses from R3 were sourced from 172.16.13.3 and subsequent responses were sourced from 172.16.43.3. This is the result of the sparse-mode SPT cutover process. Here is the state on R3 after the stream has run for a few seconds:

```
R3#show ip mroute 239.255.1.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C - Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry, E - Extranet,
       X - Proxy Join Timer Running, A - Candidate for MSDP Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group,
       G - Received BGP C-Mroute, g - Sent BGP C-Mroute,
       Q - Received BGP S-A Route, q - Sent BGP S-A Route,
       V - RD & Vector, v - Vector
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.255.1.1), 04:56:38/stopped, RP 172.16.101.1, flags: SJCL
  Incoming interface: Serial1/0, RPF nbr 172.16.13.1
  Outgoing interface list:
    Loopback103, Forward/Sparse-Dense, 04:56:38/00:02:15

(172.16.45.5, 239.255.1.1), 00:00:06/00:02:53, flags: LJT
  Incoming interface: Serial1/1, RPF nbr 172.16.43.4
  Outgoing interface list:
    Loopback103, Forward/Sparse-Dense, 00:00:06/00:02:53
```

You should notice that the RPF interface for the shared tree is S1/0, and the RPF interface for the source tree is S1/1. The T flag on the (S,G) entry indicates that the router has switched to the shortest path tree.

Once the **ping** starts on R5 and the traffic is registered with R1 (the RP), the stream is delivered on the shared tree to R3. R3 then learns the source address, and sends a Join on the shortest path to the source, toward R4. Once traffic is received on the shortest path tree, R3 prunes off the shared tree. You can observe this process on R3 in the output from **debug ip pim** and **debug ip mpacket detail**:

Note Lines have been numbered for clarity.

```
R3#
```

```
1. *Mar  3 22:23:31.766: PIM(0): Insert (172.16.45.5,239.255.1.1) join in nbr
   172.16.43.4's queue

2. *Mar  3 22:23:31.770: IP(0): MAC sa=DLCI 301 (Serial1/1)
3. *Mar  3 22:23:31.770: IP(0): IP tos=0x0, len=100, id=5826, ttl=252, prot=1

4. *Mar  3 22:23:31.770: PIM(0): Building Join/Prune packet for nbr
   172.16.43.4
5. *Mar  3 22:23:31.770: PIM(0): Adding v2 (172.16.45.5/32, 239.255.1.1), S-
   bit Join
6. *Mar  3 22:23:31.770: PIM(0): Send v2 join/prune to 172.16.43.4
   (Serial0/0.43)
7. *Mar  3 22:23:31.778: PIM(0): Insert (172.16.45.5,239.255.1.1) sgr prune in
   nbr 172.16.13.1's queue

8. *Mar  3 22:23:31.782: IP(0): MAC sa=DLCI 304 (Serial1/1)
9. *Mar  3 22:23:31.782: IP(0): IP tos=0x0, len=100, id=5826, ttl=253, prot=1

10.*Mar  3 22:23:31.782: PIM(0): Building Join/Prune packet for nbr
   172.16.13.1
11.*Mar  3 22:23:31.782: PIM(0): Adding v2 (172.16.45.5/32, 239.255.1.1), RPT-
   bit, S-bit Prune
12.*Mar  3 22:23:31.782: PIM(0): Send v2 join/prune to 172.16.13.1 (Serial1/1)

13.*Mar  3 22:23:33.774: IP(0): MAC sa=DLCI 304 (Serial1/1)
14.*Mar  3 22:23:33.774: IP(0): IP tos=0x0, len=100, id=5827, ttl=253, prot=1
```

Lines 1 through 6 show a protocol 1 (ICMP) packet arriving on S1/1, and the router creating and then sending an SPT Join to R4. Lines 7 through 12 show a packet arriving on the SPT (S1/1), and the router sending a prune toward R1 on the shared tree. Finally lines 13 and 14 show ICMP traffic continuing to arrive on the SPT every 2 seconds.

Step 2 You can control SPT cutover by controlling the SPT threshold value. The default value is zero, indicating that SPT cutover should happen as soon as traffic is detected on the SPT. You can have a router stay on the shared tree even when the SPT is available by setting the SPT-threshold value to infinity. This is often done when the shared tree and the source tree are identical and you want to conserve router resources. In this step, you enter the command **ip pim spt-threshold infinity** in global configuration mode on R3. Here is the resulting state on R3:

```
R3#show ip mroute 239.255.1.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C - Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry, E - Extranet,
       X - Proxy Join Timer Running, A - Candidate for MSDP Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group,
       G - Received BGP C-Mroute, g - Sent BGP C-Mroute,
       Q - Received BGP S-A Route, q - Sent BGP S-A Route,
       V - RD & Vector, v - Vector
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.255.1.1), 00:00:02/00:02:56, RP 172.16.101.1, flags: SCL
  Incoming interface: Serial1/0, RPF nbr 172.16.13.1
  Outgoing interface list:
    Loopback103, Forward/Sparse-Dense, 00:00:02/00:02:56
```

Note No (S,G) state is maintained on this router.

When all routers in a network are configured to stay on the sparse mode shared tree, the (S,G) state will be found only on the first-hop and RP routers.

Module 7 Answer Key: Router MQC QoS

The topology includes one Ethernet link and one serial link. Since the Ethernet link is a significantly higher bandwidth link than the serial interface, traffic will originate on the higher speed link and then be forwarded over the lower speed link. This is an ideal topology for applying QoS traffic shaping, congestion management, and congestion avoidance mechanisms.

To match the visual objectives, you had to perform the following steps (this assumes completion of the Multicast labs):

Step 1 Shut down the R1 S1/0 and tunnel interfaces.

Step 2 Apply IPv4 and IPv6 addresses from the tunnel interface to R1 S1/1 (you will get an error about an overlapping IPv6 address with the tunnel interface, but it still should accept the command), and then enable R1 S1/1.

Step 3 Shut down the R2 tunnel interface and apply the IPv4 and IPv6 addresses from the tunnel interface to S1/1. Then enable S1/1.

Step 4 Enable OSPF on S1/1 of R1 and R2.

Step 5 Shut down the R6 E0/0.46 interface.

These steps should be straightforward.

Lab 7-1 Answer Key: Classification and Marking

Task 1: Activity Procedure

In this task, you have to set up some traffic generation from R6 to R1 using IP SLA. This will be the basis of the rest of the QoS section.

Step 1 On R6, configure IP SLA for the following sessions:

- Session 1: UDP echo to port 3000 using the minimum frequency and the R1 IPv4 address as a target.
- Session 2: TCP-connect traffic to port 2999 using the minimum frequency and the R1 IPv6 address as a target.
- Session 3: ICMP echo to the R1 IPv6 address
- Session 4. ICMP jitter to the R1 IPv4 address

```
R6#sh run | b ip sla
ip sla 1
 udp-echo 172.16.124.1 3000
 frequency 5
ip sla 2
 tcp-connect 2005:DEAD:BEEF:124::1 2999
ip sla 3
 icmp-echo 2005:DEAD:BEEF:124::1
 frequency 5
ip sla 4
 icmp-jitter 172.16.124.1
 frequency 5
```

Step 2 Configure R1 as a responder.

```
R1#show run | i sla
ip sla responder
```

Step 3 On R6 E0/0, create a policy to match on the following:

- IPv4 UDP port 3000: AF41
- IPv6 TCP port 2999: AF42
- ICMPv6: AF43
- ICMPv4: EF

```
R6#show run | b class
class-map match-all UDP3000
 match access-group name UDP3000
class-map match-all TCP2999
 match access-group name TCP2999
class-map match-all ICMPv4
 match access-group name ICMPv4
class-map match-all ICMPv6
 match access-group name ICMPv6
!
policy-map MyMarking
 class ICMPv4
  set dscp ef
 class ICMPv6
  set dscp af43
 class TCP2999
  set dscp af42
 class UDP3000
  set dscp af41
!
interface Ethernet0/0.26
 encapsulation dot1Q 26
 ip address 172.16.26.6 255.255.255.0
 ip pim sparse-dense-mode
 ip ospf cost 70
 ipv6 address 2005:DEAD:BEEF:26::6/80
 ospfv3 1 ipv4 area 0
 ospfv3 1 ipv6 area 0
 service-policy output MyMarking
```

Step 4 Enable all the IP SLA sessions to start now and live forever.

```
R6#sh run | i sla
ip sla schedule 1 life forever start-time now
ip sla schedule 2 life forever start-time now
ip sla schedule 3 life forever start-time now
ip sla schedule 4 life forever start-time now
R6#show ip sla statistics
IPSLAs Latest Operation Statistics

IPSLA operation id: 1
        Latest RTT: 8 milliseconds
Latest operation start time: 12:48:19 PST Thu Oct 3 2013
Latest operation return code: OK
Number of successes: 340
Number of failures: 0
Operation time to live: Forever



IPSLA operation id: 2
        Latest RTT: 8 milliseconds
Latest operation start time: 12:48:04 PST Thu Oct 3 2013
Latest operation return code: OK
Number of successes: 29
Number of failures: 0
Operation time to live: Forever
```

```
IPSLA operation id: 3
        Latest RTT: 7 milliseconds
Latest operation start time: 12:48:19 PST Thu Oct 3 2013
Latest operation return code: OK
Number of successes: 340
Number of failures: 0
Operation time to live: Forever



IPSLA operation id: 4
Type of operation: icmp-jitter
        Latest RTT: 9 milliseconds
Latest operation start time: 12:48:19 PST Thu Oct 3 2013
Latest operation return code: OK
RTT Values:
        Number Of RTT: 10              RTT Min/Avg/Max: 8/9/11
milliseconds
Latency one-way time:
        Number of Latency one-way Samples: 9
        Source to Destination Latency one way Min/Avg/Max: 4/4/7
milliseconds
        Destination to Source Latency one way Min/Avg/Max: 4/4/5
milliseconds
Jitter Time:
        Number of SD Jitter Samples: 9
        Number of DS Jitter Samples: 9
        Source to Destination Jitter Min/Avg/Max: 0/1/2 milliseconds
        Destination to Source Jitter Min/Avg/Max: 0/1/1 milliseconds
Over Threshold:
        Number Of RTT Over Threshold: 0 (0%)
Packet Late Arrival: 0
Out Of Sequence: 0
        Source to Destination: 0        Destination to Source 0
        In both Directions: 0
Packet Skipped: 0       Packet Unprocessed: 0
Packet Loss: 0
        Loss Periods Number: 0
        Loss Period Length Min/Max: 0/0
        Inter Loss Period Length Min/Max: 0/0
Number of successes: 340
Number of failures: 0
Operation time to live: Forever
```

Step 5 Verify your policy.

```
R6#sh policy-map interface  e0/0.26
 Ethernet0/0.26

  Service-policy output: MyMarking

    Class-map: ICMPv4 (match-all)
      1720 packets, 110080 bytes
      5 minute offered rate 1000 bps, drop rate 0000 bps
      Match: access-group name ICMPv4
      QoS Set
        dscp ef
          Packets marked 1720

    Class-map: ICMPv6 (match-all)
      205 packets, 16016 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: access-group name ICMPv6
      QoS Set
        dscp af43
          Packets marked 205
```

```
Class-map: TCP2999 (match-all)
  56 packets, 3416 bytes
  5 minute offered rate 0000 bps, drop rate 0000 bps
  Match: access-group name TCP2999
  QoS Set
    dscp af42
      Packets marked 56

Class-map: UDP3000 (match-all)
  172 packets, 11008 bytes
  5 minute offered rate 0000 bps, drop rate 0000 bps
  Match: access-group name UDP3000
  QoS Set
    dscp af41
      Packets marked 172

Class-map: class-default (match-any)
  401 packets, 35760 bytes
  5 minute offered rate 0000 bps, drop rate 0000 bps
  Match: any
```

Lab 7-2 Answer Key: Class-Based Shaper

In this activity, you will examine the operation of the MQC class based shaper. You will configure a nested policy to shape the interface and shape specific types of traffic.

Task 1: Activity Procedure

In this task you have to configure a multistep policy. On R2, you have to configure a policy on ingress from R6 to match on the markings made in the previous lab. You have to set the QoS groups as follows:

- EF: Group 1
- AF41: Group 2
- AF42: Group 3
- AF43: Group 4

```
R2#sho run | b class
class-map match-all AF43
 match dscp af43
class-map match-all AF42
 match dscp af42
class-map match-all AF41
 match dscp af41
class-map match-all EF
 match dscp ef
!
policy-map FromR6
 class EF
  set qos-group 1
 class AF41
  set qos-group 2
 class AF42
  set qos-group 3
 class AF43
  set qos-group 4
!
interface Ethernet0/0.26
 encapsulation dot1Q 26
 ip address 172.16.26.2 255.255.255.0
 ip pim sparse-dense-mode
 ip ospf cost 70
 ipv6 address 2005:DEAD:BEEF:26::2/80
 ospfv3 1 ipv6 area 0
 ospfv3 1 ipv4 area 0
 service-policy input FromR6
```

Then, on the R2 egress interface to R1, you have to create another policy to shape all traffic to 768 kbps, with a Tc of 10 seconds. You also have to create a policy that will shape all traffic from Group 2 to 25 percent of the 768 kbps, Group 3 to 15 percent of the 768 kbps, and Group 4 to 10 percent of the 768 kbps. To accomplish this, you have to do nested policies for the egress interface—a parent to shape all traffic to 768 kbps, and then a child to shape specific traffic to specific percentages of the 768 kbps. To get the Tc to be 10 seconds, set the Bc to be 10 times the value of the CIR.

```
class-map match-all Group4
 match qos-group 4
class-map match-all Group3
 match qos-group 3
class-map match-all Group2
 match qos-group 2
class-map match-all Group1
 match qos-group 1
!
```

```
policy-map ToR1P
 class Group3
  shape average percent 15
 class Group2
  shape average percent 25
 class Group4
  shape average percent 10
policy-map ToR1GP
 class class-default
  shape average 768000 7680000
   service-policy ToR1P
!
```

To verify the operation of the class-based shaper, you can look at the policy that is applied to the egress interface. Note that the configuration of the child policy, in this case named "ToR1P," was configured with percentages, but the **show policy-map interface** command calculates the bps for the shaper.

```
R2#sh policy-map int s1/1
 Serial1/1

  Service-policy output: ToR1GP

    Class-map: class-default (match-any)
      59273 packets, 3944951 bytes
      5 minute offered rate 1000 bps, drop rate 0000 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 59273/3960095
      shape (average) cir 768000, bc 7680000, be 7680000
      target shape rate 768000

      Service-policy : ToR1P

        Class-map: Group3 (match-all)
          984 packets, 63960 bytes
          5 minute offered rate 0000 bps, drop rate 0000 bps
          Match: qos-group 3
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 984/63960
          shape (average) cir 115200, bc 461, be 460
          target shape rate 115200

        Class-map: Group2 (match-all)
          2949 packets, 141552 bytes
          5 minute offered rate 0000 bps, drop rate 0000 bps
          Match: qos-group 2
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 2949/141552
          shape (average) cir 192000, bc 768, be 768
          target shape rate 192000

        Class-map: Group4 (match-all)
          2949 packets, 235920 bytes
          5 minute offered rate 0000 bps, drop rate 0000 bps
          Match: qos-group 4
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 2949/235920
          shape (average) cir 76800, bc 308, be 307
          target shape rate 76800

        Class-map: class-default (match-any)
          52391 packets, 3503519 bytes
          5 minute offered rate 0000 bps, drop rate 0000 bps
          Match: any

          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 52391/3518663
```

Lab 7-3 Answer Key: Class-Based Policer

In this lab, you configure a class-based policer. You enforce the policy that you created on R2 on the ingress of R1, plus some additional limits. This is another example of a nested policy.

Task 1: Activity Procedure

Complete these steps:

On R1, you create a policy that will police the traffic from R2 so that all traffic will be policed at 768 kbps.

```
R1#show run | b policy
policy-map FromR2P
 class class-default
  police 768000
```

Then you create a child policy that policed traffic as follows:

- All traffic marked as AF43 to the minimum rate (8 kbps)
- All traffic marked as AF42 to twice the minimum rate (16 kbps)
- All traffic marked as AF41 to four times the minimum rate (32 kbps)
- All traffic marked as EF to eight times the minimum rate (64 kbps)

```
R1#show run | b class
class-map match-all AF43
 match dscp af43
class-map match-all AF42
 match dscp af42
class-map match-all AF41
 match dscp af41
class-map match-all EF
 match dscp ef
!
policy-map FromR2C
 class AF43
  police 8000
 class AF42
  police 16000
 class AF41
  police 32000
 class EF
  police 64000
policy-map FromR2P
 class class-default
  police 768000
   service-policy FromR2C
```

If all went correctly, when you perform the following ping from R6, the output should be similar. With the policy, you should be over-running the policer for the EF queue and getting drops.

```
ping 172.16.124.1 repeat 1000 size 1000 timeout 1
R6#ping 172.16.124.1 rep 1000 tim 1
Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 172.16.124.1, timeout is 1
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!
Success rate is 98 percent (982/1000), round-trip min/avg/max = 5/8/13
ms
```

Verify that you are seeing drops in your policy on R1.

```
R1#show policy-map interface serial 1/1
 Serial1/1

  Service-policy input: FromR2P

    Class-map: class-default (match-any)
      7466 packets, 1973674 bytes
      5 minute offered rate 1000 bps, drop rate 0000 bps
      Match: any
      police:
          cir 768000 bps, bc 24000 bytes
        conformed 7442 packets, 1963078 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          drop
        conformed 1000 bps, exceeded 0000 bps

      Service-policy : FromR2C

        Class-map: AF43 (match-all)
          235 packets, 18800 bytes
          5 minute offered rate 0000 bps, drop rate 0000 bps
          Match:  dscp af43 (38)
          police:
              cir 8000 bps, bc 1500 bytes
            conformed 235 packets, 18800 bytes; actions:
              transmit
            exceeded 0 packets, 0 bytes; actions:
              drop
            conformed 0000 bps, exceeded 0000 bps

        Class-map: AF42 (match-all)
          76 packets, 4940 bytes
          5 minute offered rate 0000 bps, drop rate 0000 bps
          Match:  dscp af42 (36)
          police:
              cir 16000 bps, bc 1500 bytes
            conformed 76 packets, 4940 bytes; actions:
              transmit
            exceeded 0 packets, 0 bytes; actions:
              drop
            conformed 0000 bps, exceeded 0000 bps

        Class-map: AF41 (match-all)
          235 packets, 11280 bytes
          5 minute offered rate 0000 bps, drop rate 0000 bps
          Match:  dscp af41 (34)
          police:
              cir 32000 bps, bc 1500 bytes
            conformed 235 packets, 11280 bytes; actions:
              transmit
            exceeded 0 packets, 0 bytes; actions:
```

```
          drop
        conformed 0000 bps, exceeded 0000 bps

    Class-map: EF (match-all)
      2803 packets, 201662 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match:  dscp ef (46)
      police:
          cir 64000 bps, bc 2000 bytes
        conformed 2769 packets, 190626 bytes; actions:
          transmit
        exceeded 24 packets, 10596 bytes; actions:
          drop
        conformed 0000 bps, exceeded 0000 bps

    Class-map: class-default (match-any)
      4117 packets, 1736992 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
```

Lab 7-4 Answer Key: Avoiding and Managing Congestion

In this lab, you modify configuration to now include LLQ and WRED.

Task 1: Activity Procedure

On R2, you add a queue that matches all traffic to your existing policy that shapes the different types of traffic. Reserve 50 percent of the bandwidth for that new queue.

```
R2#show run | b class
class-map match-all All
 match any
!
policy-map ToR1P
 class Group3
  shape average percent 15
 class Group2
  shape average percent 25
 class Group4
  shape average percent 10
 class All
  bandwidth percent 50
```

Then you add the following policy within this new queue that matches all traffic:

- EF (Group 1) 50 percent
- Pre-emptive
- Burst of 1000 bytes
- Class of 3 gets 25 percent
- Class of 2 gets 15 percent
- Class of 1 gets 5 percent

For the class of 1, make sure to use congestion avoidance using default thresholds for DSCP values.

```
R2#show run | b class
class-map match-all CS1
 match dscp cs1
class-map match-all CS2
 match dscp cs2
class-map match-all CS3
 match dscp cs3
!
policy-map ToR1C
```

```
 class Group1
  priority percent 50 1000
 class CS3
  bandwidth percent 25
 class CS2
  bandwidth percent 15
 class CS1
  bandwidth percent 5
  random-detect dscp-based
!
policy-map ToR1P
 class Group3
  shape average percent 15
 class Group2
  shape average percent 25
 class Group4
  shape average percent 10
 class All
  bandwidth percent 50
   service-policy ToR1C
```

To be able to ping from R2 to R1 with a class selector of 1, you have to use the extended ping option and then go into the extended options to set the TOS field. The class selector of 1, when represented by the ToS octet, is 32.

```
R2#ping ip
Target IP address: 172.16.124.1
Repeat count [5]: 1000
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface:
Type of service [0]: 32
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 172.16.124.1, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (1000/1000), round-trip min/avg/max = 2/8/13 ms
```

Verify that you get hits on the policy, including within the class of 1.

```
R2#show policy-map interface s 1/1
 Serial1/1

  Service-policy output: ToR1GP

    Class-map: class-default (match-any)
      3941 packets, 314343 bytes
      5 minute offered rate 1000 bps, drop rate 0000 bps
```

```
Match: any
Queueing
queue limit 64 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 3941/314983
shape (average) cir 768000, bc 7680000, be 7680000
target shape rate 768000

Service-policy : ToR1P

  Class-map: Group3 (match-all)
    44 packets, 2860 bytes
    5 minute offered rate 0000 bps, drop rate 0000 bps
    Match: qos-group 3
    Queueing
    queue limit 64 packets
    (queue depth/total drops/no-buffer drops) 0/0/0
    (pkts output/bytes output) 44/2860
    shape (average) cir 115200, bc 461, be 460
    target shape rate 115200

  Class-map: Group2 (match-all)
    127 packets, 6096 bytes
    5 minute offered rate 0000 bps, drop rate 0000 bps
    Match: qos-group 2
    Queueing
    queue limit 64 packets
    (queue depth/total drops/no-buffer drops) 0/0/0
    (pkts output/bytes output) 127/6096
    shape (average) cir 192000, bc 768, be 768
    target shape rate 192000

  Class-map: Group4 (match-all)
    127 packets, 10160 bytes
    5 minute offered rate 0000 bps, drop rate 0000 bps
    Match: qos-group 4
    Queueing
    queue limit 64 packets
    (queue depth/total drops/no-buffer drops) 0/0/0
    (pkts output/bytes output) 127/10160
    shape (average) cir 76800, bc 308, be 307
    target shape rate 76800

  Class-map: All (match-all)
    3643 packets, 295227 bytes
    5 minute offered rate 1000 bps, drop rate 0000 bps
    Match: any
    Queueing
    queue limit 64 packets
    (queue depth/total drops/no-buffer drops) 0/0/0
    (pkts output/bytes output) 3643/295867
    bandwidth 50% (384 kbps)

    Service-policy : ToR1C

      queue stats for all priority classes:
        Queueing
        queue limit 64 packets
        (queue depth/total drops/no-buffer drops) 0/0/0
        (pkts output/bytes output) 1270/55880

      Class-map: Group1 (match-all)
        1270 packets, 55880 bytes
        5 minute offered rate 0000 bps, drop rate 0000 bps
        Match: qos-group 1
        Priority: 50% (192 kbps), burst bytes 1000, b/w exceed drops: 0
      Class-map: CS3 (match-all)
        0 packets, 0 bytes
        5 minute offered rate 0000 bps, drop rate 0000 bps
        Match:  dscp cs3 (24)
        Queueing
        queue limit 64 packets
        (queue depth/total drops/no-buffer drops) 0/0/0
        (pkts output/bytes output) 0/0
        bandwidth 25% (96 kbps)

      Class-map: CS2 (match-all)
        0 packets, 0 bytes
        5 minute offered rate 0000 bps, drop rate 0000 bps
        Match:  dscp cs2 (16)
        Queueing
        queue limit 64 packets
        (queue depth/total drops/no-buffer drops) 0/0/0
        (pkts output/bytes output) 0/0
        bandwidth 15% (57 kbps)

      Class-map: CS1 (match-all)
        2000 packets, 208000 bytes
        5 minute offered rate 0000 bps, drop rate 0000 bps
        Match:  dscp cs1 (8)
        Queueing
        queue limit 64 packets
        (queue depth/total drops/no-buffer drops) 0/0/0
        (pkts output/bytes output) 2000/208000
        bandwidth 5% (19 kbps)
          Exp-weight-constant: 9 (1/512)
          Mean queue depth: 0 packets
        dscp   Transmitted  Random drop  Tail drop  Minimum  Maximum Mark
                pkts/bytes   pkts/byte  pkts/byte   thresh  thresh  prob

         cs1    2000/208000     0/0         0/0        22      40    1/10

      Class-map: class-default (match-any)
        373 packets, 31347 bytes
        5 minute offered rate 0000 bps, drop rate 0000 bps
        Match: any

        queue limit 64 packets
        (queue depth/total drops/no-buffer drops) 0/0/0
        (pkts output/bytes output) 373/31987

  Class-map: class-default (match-any)
    0 packets, 0 bytes
    5 minute offered rate 0000 bps, drop rate 0000 bps
    Match: any

    queue limit 64 packets
    (queue depth/total drops/no-buffer drops) 0/0/0
    (pkts output/bytes output) 0/0
```

Lab 8-1 Answer Key: Implementing NAT

Task 1: Configure the Required Router Interfaces

In this task, you have to set the basic foundation on which the network services will be implemented. On R1, you need to move some addresses and add a default route into OSPF for R7.

```
R1#show run int e0/0.124
Building configuration...

Current configuration : 136 bytes
!
interface Ethernet0/0.124
 encapsulation dot1Q 124
 ip address 172.16.124.1 255.255.255.0
 ipv6 address 2005:DEAD:BEEF:124::1/80
end
R1#show run | section router ospfv3
router ospfv3 1
 !
 address-family ipv4 unicast
  default-information originate always
 exit-address-family
 !
 address-family ipv6 unicast
  default-information originate always
 exit-address-family
```

To see if R7 got the routes it needs, look at the routing table:

```
R7#sh ip route
Codes: L - local, 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, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is 172.16.120.1 to network 0.0.0.0

O*E2  0.0.0.0/0 [110/1] via 172.16.120.1, 00:00:23, Ethernet0/0.17
      7.0.0.0/32 is subnetted, 1 subnets
C        7.7.7.7 is directly connected, Loopback0
      172.16.0.0/16 is variably subnetted, 7 subnets, 2 masks
C        172.16.78.0/24 is directly connected, Ethernet0/0.78
L        172.16.78.7/32 is directly connected, Ethernet0/0.78
O        172.16.101.0/24 [110/11] via 172.16.120.1, 00:03:03, Ethernet0/0.17
C        172.16.107.0/24 is directly connected, Loopback107
L        172.16.107.1/32 is directly connected, Loopback107
C        172.16.120.0/24 is directly connected, Ethernet0/0.17
R7#sh ipv6 route
IPv6 Routing Table - default - 8 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, HA - Home Agent, MR - Mobile Router, R - RIP
       H - NHRP, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       IS - ISIS summary, D - EIGRP, EX - EIGRP external, NM - NEMO
       ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2, ls - LISP site
       ld - LISP dyn-EID, a - Application
OE2 ::/0 [110/1], tag 1
     via FE80::A8BB:CCFF:FE00:100, Ethernet0/0.17
O   2001:DEAD:BEEF:101::1/128 [110/10]
```

```
     via FE80::A8BB:CCFF:FE00:100, Ethernet0/0.17
LC  2001:DEAD:BEEF:105::1/128 [0/0]
     via Loopback107, receive
C   2005:DEAD:BEEF:78::/80 [0/0]
     via Ethernet0/0.78, directly connected
L   2005:DEAD:BEEF:78::7/128 [0/0]
     via Ethernet0/0.78, receive
C   2005:DEAD:BEEF:120::/80 [0/0]
     via Ethernet0/0.17, directly connected
L   2005:DEAD:BEEF:120::7/128 [0/0]
     via Ethernet0/0.17, receive
```

Remember to add OSPF onto the interfaces between R2 and R4.

Task 2: Configure NAT and Verify Connectivity

In this task, you will configure NAT on R1 to hide the real IPv4 address of R7 from R2 and R4. The steps lead you to configure NAT with overloading (also known as PAT).

```
R1#show run | begin 0/0
interface Ethernet0/0
 no ip address
!
interface Ethernet0/0.17
 encapsulation dot1Q 17
 ip address 172.16.120.1 255.255.255.0
 ip pim sparse-dense-mode
 ip nat inside
 ip virtual-reassembly in
 ipv6 address 2005:DEAD:BEEF:120::1/80
 ospfv3 1 ipv6 area 0
 ospfv3 1 ipv4 area 0
!
interface Ethernet0/0.124
 encapsulation dot1Q 124
 ip address 172.16.124.1 255.255.255.0
 ip nat outside
 ip virtual-reassembly in
 ipv6 address 2005:DEAD:BEEF:124::1/80
!
ip nat inside source list FromR7 interface Ethernet0/0.124 overload
!
ip access-list standard FromR7
 permit 172.16.120.7
 permit 172.16.107.1
 permit 172.16.78.7
```

Step 1 Verify that R7 can ping the addresses of R2 and R4 on the shared network between R1, R2, and R4.

```
R7#ping 172.16.124.4 sour lo 107
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.124.4, timeout is 2 seconds:
Packet sent with a source address of 172.16.107.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
R1#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
icmp 172.16.124.1:3    172.16.107.1:3     172.16.124.4:3     172.16.124.4:3
```

Lab 8-2 Answer Key: Implementing FHRP

Task 1: Configure HSRP for IPv4 and IPv6 and Verify Connectivity Router Interfaces

In this task, you will add HSRP for IPv4 and IPv6 on R2 and R4. You will make the virtual router addresses into the default route for R1. Add HSRP on the routers R2 and R4 with a set of virtual addresses of 172.16.124.24 and FE80::5:73FF:FEA0::18, but do set the virtual IPv6 address directly.

First, since you are configuring both IPv4 and IPv6, you have to set HSRP to version 2, because version 1 does not support IPv6. Since you cannot set the IPv6 virtual address directly, you have to use the **autoconfig** option. This will use the virtual MAC address for the virtual router and use EUI-64 to derive the virtual link-local address. The virtual MAC address is 0005.73a0.0*xxx*, where the *xxx* is the group number in hex. In this case, it is 0x018, which turns out to be 24 in decimal. A similar task is needed with IPv4 for the requirement that reads "Make sure the virtual MAC address for the virtual IPv4 address is 0000.0C9F.F00C, but do not use the **standby mac-address** command." The address 0000.0C9F.F*xxx* is the virtual MAC address for HSRPv2 for IPv4. You should set the group number to 12.

For IPv4, make sure that R2 is the active router with a priority of 1 above the default. The default is 100. Make sure that it waits a minimum of 30 seconds before taking over the active role by using **preempt delay minimum 30**.

You are instructed to have R4 take over with no delay if the interface to R6 goes down on R2, and to use the default decrement. This is a fairly straightforward tracking task, with pre-emption enabled on R4.

For IPv6, you reverse the roles, having R4 as the active router with a priority 2 above the default, or 102.

```
R2#sh run int e 0/0.124
Building configuration...

Current configuration : 351 bytes
!
interface Ethernet0/0.124
 encapsulation dot1Q 124
 ip address 172.16.124.2 255.255.255.0
 standby version 2
 standby 12 ip 172.16.124.24
 standby 12 priority 101
 standby 12 preempt delay minimum 30
 standby 12 track 1 decrement 10
 standby 24 ipv6 autoconfig
 ipv6 address 2005:DEAD:BEEF:124::2/80
 ospfv3 1 ipv6 area 0
 ospfv3 1 ipv4 area 0
R4#show run int e 0/0
Building configuration...

Current configuration : 292 bytes
!
interface Ethernet0/0
 ip address 172.16.124.4 255.255.255.0
 standby version 2
 standby 12 ip 172.16.124.24
 standby 12 preempt
 standby 24 ipv6 autoconfig
 standby 24 priority 102
 standby 24 preempt
 ipv6 address 2005:DEAD:BEEF:124::4/80
 ospfv3 1 ipv6 area 0
 ospfv3 1 ipv4 area 0
```

```
R2#show standby brief
                     P indicates configured to preempt.
                     |
Interface   Grp  Pri P State   Active          Standby         Virtual IP
Et0/0.124   12   101 P Active  local           172.16.124.4    172.16.124.24
Et0/0.124   24   100   Standby FE80::A8BB:CCFF:FE00:400
                                               local
FE80::5:73FF:FEA0:18
R4#show standby brief
                     P indicates configured to preempt.
                     |
Interface   Grp  Pri P State   Active          Standby         Virtual IP
Et0/0       12   100 P Standby 172.16.124.2    local           172.16.124.24
Et0/0       24   102 P Active  local           FE80::A8BB:CCFF:FE00:200

FE80::5:73FF:FEA0:18
```

On R1, make the virtual address your default route.

```
R1#show run | include ip route
ip route 0.0.0.0 0.0.0.0 172.16.124.24
R1#show run | include ipv6 route
ipv6 route ::/0 Ethernet0/0.124 FE80::5:73FF:FEA0:18
```

On R2 and R4, make R1 the default for IPv6.

```
R2#show run | include ipv6 route
ipv6 route ::/0 Ethernet0/0.124 FE80::A8BB:CCFF:FE00:100
```

Verify that R7 can ping the virtual addresses.

```
R7#ping 172.16.104.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.104.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
R7#ping 2001:DEAD:BEEF:102::1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DEAD:BEEF:102::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```

Task 2: Configure GLBP for IPv4 and IPv6 and Verify Connectivity

In this task, you are told to configure R2 and R4 for GLBP for IPv4 and IPv6, and to change the default route for R1 to match the new virtual router addresses. On R2 and R4, configure GLBP for IPv4 with a virtual IPv4 address of 172.16.124.124. Make sure that R2 is the AVG as long as it is up. Have R4 stop acting as an AVF if the interface to R5 fails, and use the default decrement.

Configure GLBP for IPv6 with the virtual address of FE80::7:B4FF:FE00:7C00 without setting the address directly. Make sure that R4 is the AVG. The only trick here is setting the IPv6 address without explitly setting it. The **glbp ipv6 autoconfig** command uses the virtual MAC address for the bases of the virtual link-local address. The virtual MAC address for IPv6 GLBP is 0007.B40*x.xxyy*, where *xxx* is the group number in hex and *yy* is the virtual forwarder number in hex. The IPv6 address that you are supposed to use has 07C in the *xxx* field, which converts to 124 in decimal.

```
R2#show run int e 0/0.124
Building configuration...

Current configuration : 440 bytes
!
interface Ethernet0/0.124
 encapsulation dot1Q 124
 ip address 172.16.124.2 255.255.255.0
 standby version 2
 standby 12 ip 172.16.124.24
 standby 12 priority 101
 standby 12 preempt delay minimum 30
 standby 12 track 1 decrement 10
 standby 24 ipv6 autoconfig
 glbp 1 ip 172.16.124.124
 glbp 1 priority 101
 glbp 1 preempt
 glbp 124 ipv6 autoconfig
 ipv6 address 2005:DEAD:BEEF:124::2/80
 ospfv3 1 ipv4 area 0
 ospfv3 1 ipv6 area 0
R4#sh run | b track
track 1 interface Serial1/0 line-protocol
!
interface Ethernet0/0
 ip address 172.16.124.4 255.255.255.0
 standby version 2
 standby 12 ip 172.16.124.24
 standby 12 preempt
 standby 24 ipv6 autoconfig
 standby 24 priority 102
 standby 24 preempt
 glbp 1 ip 172.16.124.124
 glbp 1 weighting 100 lower 95
 glbp 1 weighting track 1
 glbp 124 ipv6 autoconfig
 glbp 124 priority 101
 glbp 124 preempt
 ipv6 address 2005:DEAD:BEEF:124::4/80
 ospfv3 1 ipv4 area 0
 ospfv3 1 ipv6 area 0
R4#sh glbp brief
Interface   Grp  Fwd Pri State    Address         Active router   Standby
router
Et0/0       1    -   100 Standby  172.16.124.124  172.16.124.2    local
Et0/0       1    1   -   Listen   0007.b400.0101  172.16.124.2    -
Et0/0       1    2   -   Active   0007.b400.0102  local           -
Et0/0       124  -   101 Active   FE80::7:B4FF:FE00:7C00
                                                  local
FE80::A8BB:CCFF:FE00:200
Et0/0       124  1   -   Listen   0007.b400.7c01  FE80::A8BB:CCFF:FE00:200
                                                                  -
Et0/0       124  2   -   Active   0007.b400.7c02  local           -
```

To verify that GLBP is functional, you have to change the default gateway on R1 to match the virtual address from GLBP, and from R7 ping the Loopback 10x addresses of R2 and R4.

```
R1#show ip route s
R1#show ip route static
Codes: L - local, 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, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is 172.16.124.124 to network 0.0.0.0

S*    0.0.0.0/0 [1/0] via 172.16.124.124
R1#show ipv6 route static
IPv6 Routing Table - default - 9 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, HA - Home Agent, MR - Mobile Router, R - RIP
       H - NHRP, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       IS - ISIS summary, D - EIGRP, EX - EIGRP external, NM - NEMO
       ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2, ls - LISP site
       ld - LISP dyn-EID, a - Application
S   ::/0 [1/0]
     via FE80::7:B4FF:FE00:7C00, Ethernet0/0.124
R7#ping 172.16.102.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.102.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
R7#ping 172.16.104.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.104.1, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 1/1/1 ms
R7#ping 2001:dead:beef:102::1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DEAD:BEEF:102::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/14 ms
R7#ping 2001:dead:beef:104::1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DEAD:BEEF:104::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```

Task 3: Configure VRRP for IPv4 and Verify Connectivity

In this task, you will configure R2 and R4 for VRRP for IPv4. You will also change the default route for R1 to match the new virtual router addresses. On R2 and R4, configure VRRP for IPv4 with a virtual IPv4 address of 172.16.124.224. Make sure that R2 is the master for this group with a priority of 1 more than default. The default is 100. You are told tomake sure that the virtual MAC address is 0000.5e00.01e0. Like HSRPv1, the group number is the last 2 hex characters of the MAC address. 0xE0 in hex is 224 is decimal.

Make sure that the master sends hellos every 250 ms using the **vrrp timers advertise msec** command. Do not configure timers on R4; instead have it learn from R2 using the **vrrp timers learn** command.

```
R2#show run int e0/0.124
Building configuration...

Current configuration : 491 bytes
!
interface Ethernet0/0.124
 encapsulation dot1Q 124
 ip address 172.16.124.2 255.255.255.0
 standby version 2
 standby 12 ip 172.16.124.24
 standby 12 priority 101
 standby 12 preempt delay minimum 30
 standby 12 track 1 decrement 10
 standby 24 ipv6 autoconfig
 glbp 1 ip 172.16.124.124
 glbp 1 priority 101
 glbp 1 preempt
 glbp 124 ipv6 autoconfig
 ipv6 address 2005:DEAD:BEEF:124::2/80
 ospfv3 1 ipv4 area 0
 ospfv3 1 ipv6 area 0
 vrrp 224 ip 172.16.124.224
 vrrp 224 timers advertise msec 250
 vrrp 224 priority 101
R4#show run int e 0/0
Building configuration...

Current configuration : 493 bytes
!
```

```
interface Ethernet0/0
 ip address 172.16.124.4 255.255.255.0
 standby version 2
 standby 12 ip 172.16.124.24
 standby 12 preempt
 standby 24 ipv6 autoconfig
 standby 24 priority 102
 standby 24 preempt
 glbp 1 ip 172.16.124.124
 glbp 1 weighting 100 lower 95
 glbp 1 weighting track 1
 glbp 124 ipv6 autoconfig
 glbp 124 priority 101
 glbp 124 preempt
 ipv6 address 2005:DEAD:BEEF:124::4/80
 ospfv3 1 ipv4 area 0
 ospfv3 1 ipv6 area 0
 vrrp 224 ip 172.16.124.224
 vrrp 224 timers learn
R2#show vrrp
Ethernet0/0.124 - Group 224
  State is Master
  Virtual IP address is 172.16.124.224
  Virtual MAC address is 0000.5e00.01e0
  Advertisement interval is 0.250 sec
  Preemption enabled
  Priority is 101
  Master Router is 172.16.124.2 (local), priority is 101
  Master Advertisement interval is 0.250 sec
  Master Down interval is 1.355 sec
```

Change the default gateway on R1 to match the virtual address from VRRP and verify that R7 can still ping the Loopback 10x addresses of R2 and R4.

```
R1#sh ip route static
Codes: L - local, 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, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is 172.16.124.224 to network 0.0.0.0

S*    0.0.0.0/0 [1/0] via 172.16.124.224
R7#ping 172.16.102.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.102.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
R7#ping 172.16.104.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.104.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```

Lab 8-3 Answer Key: Implementing NTP

Activity Objective

In this section, you will configure NTP to run over IPv4 and IPv6.

Task 1: Configure NTP on R7

Activity Procedure

In this task, you will configure R7 to be the NTP server that will feed time to R1. R1 will feed the other routers with authentication.

Configure R7 as the NTP master. Have R1 pull time from R7 using the IPv6 address 2001:DEAD:BEEF:107::1. Have R2 and R4 get their time from their peer R1 without configuring the peer address. Make sure that NTP is secure between R1, R2, and R4 with a key of 42 and string of C1sc0. By default, NTP broadcast mode is enabled, so R1 will send time out its different interfaces. R2 and R4 will pickup on the time as long as there is authentication tied to it.

```
R7#show run | i ntp
ntp authentication-key 42 md5 143443180F54 7
ntp trusted-key 42
ntp master
R1#show run | i ntp
ntp authentication-key 42 md5 04785A150C71 7
ntp trusted-key 42
ntp server 2001:DEAD:BEEF:107::1
R2#show run | i ntp
ntp authentication-key 42 md5 096F1F1A1A55 7
ntp authenticate
ntp trusted-key 42
R4#show run | i ntp
ntp authentication-key 42 md5 143443180F54 7
ntp authenticate
ntp trusted-key 42
R1#show ntp associations detail
2001:DEAD:BEEF:107::1 configured, ipv6, our_master, sane, valid, stratum 8
ref ID 127.127.1.1    , time D5ED9D85.8AC08490 (09:36:05.542 PST Wed Sep 25
2013)
our mode client, peer mode server, our poll intvl 256, peer poll intvl 256
root delay 0.00 msec, root disp 2.16, reach 377, sync dist 7.25
delay 0.00 msec, offset 0.0000 msec, dispersion 3.10, jitter 0.97 msec
precision 2**10, version 4
assoc id 62272, assoc name 2001:DEAD:BEEF:107::1
assoc in packets 28, assoc out packets 28, assoc error packets 0
org time 00000000.00000000 (16:00:00.000 PST Wed Dec 31 1899)
rec time D5ED9D86.0DD2F1D0 (09:36:06.054 PST Wed Sep 25 2013)
xmt time D5ED9D86.0DD2F1D0 (09:36:06.054 PST Wed Sep 25 2013)
filtdelay =     1.00    1.00    0.00    1.00    0.00    1.00    1.00    1.00
filtoffset =    0.50    0.50    0.00   -0.50    0.00    0.50    0.50    0.50
filterror =     1.95    1.98    2.01    2.04    6.00    6.03    6.06    6.09
minpoll = 6, maxpoll = 10 0.50
filterror =     1.95    1.98    3.96    3.99    5.95    5.98    7.00    8.04
minpoll = 6, maxpoll = 10
R1#show ntp status
Clock is synchronized, stratum 9, reference is 223.121.117.38
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**10
ntp uptime is 75500 (1/100 of seconds), resolution is 4000
reference time is D5ED9AE4.0DD2F1D0 (09:24:52.054 PST Wed Sep 25 2013)
clock offset is 0.0000 msec, root delay is 0.00 msec
root dispersion is 7.94 msec, peer dispersion is 3.43 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000000000 s/s
system poll interval is 128, last update was 77 sec ago.
R2#show ntp associations detail
172.16.124.1 dynamic, ipv4, authenticated, our_master, sane, valid, stratum 9
```

```
ref ID 223.121.117.38 , time D5ED9C78.0D916898 (09:31:36.053 PST Wed Sep 25
2013)
our mode passive, peer mode active, our poll intvl 64, peer poll intvl 256
root delay 0.00 msec, root disp 8.45, reach 60, sync dist 3952.07
delay 0.00 msec, offset 0.0000 msec, dispersion 3942.07, jitter 0.97 msec
precision 2**10, version 4
assoc id 16148, assoc name 172.16.124.1
assoc in packets 20, assoc out packets 13, assoc error packets 11
org time D5ED9E06.E353FA40 (09:38:14.888 PST Wed Sep 25 2013)
rec time D5ED9D09.0D0E5628 (09:34:01.051 PST Wed Sep 25 2013)
xmt time D5ED9D09.0D0E5628 (09:34:01.051 PST Wed Sep 25 2013)
filtdelay =     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filtoffset =    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filterror =  16000.0    5.76    6.76 16000.0 16000.0 16000.0 16000.0 16000.0
minpoll = 6, maxpoll = 10
```

Lab 8-4 Answer Key: Implementing DHCP for IPv4 and IPv6

Task 1: Configure DHCP for IPv4 and IPv6

In this task, you will configure R1 as the DHCP server for IPv4 and IPv6 and serve addressing to R7.

Make R1 the DHCP server for R7 to issue its addresses for both IPv4 and IPv6. Make sure that R7 gets the correct IPv4 address as listed on the visual objective from previous tasks, which was 172.16.120.7. This requirement means you can either create a host-specific set of DHCP pools or exclude all other addresses so R7 can only get its addresses. Here is an example of the latter option.

```
R1#show run | b dhcp
ip dhcp excluded-address 172.16.120.1 172.16.120.6
ip dhcp excluded-address 172.16.120.8 172.16.120.255
!
ip dhcp pool ToR7
 network 172.16.120.0 255.255.255.0
!
ipv6 dhcp pool ToR7
 address prefix 2005:DEAD:BEEF:120::/80
!
interface Ethernet0/0.17
 encapsulation dot1Q 17
 ip address 172.16.120.1 255.255.255.0
 ip pim sparse-dense-mode
 ip nat inside
 ip virtual-reassembly in
 ipv6 address 2005:DEAD:BEEF:120::1/80
 ipv6 dhcp server ToR7
 ospfv3 1 ipv4 area 0
 ospfv3 1 ipv6 area 0
R7#sh run int e0/0.17
Building configuration...

Current configuration : 149 bytes
!
interface Ethernet0/0.17
 encapsulation dot1Q 17
 ip address dhcp
 ipv6 address dhcp
 ipv6 enable
 ospfv3 1 ipv6 area 0
 ospfv3 1 ipv4 area 0
```

Note You have to use the **ipv6 enable** command on the interface to get DHCPv6 to work. This command is not typically needed, since you typically have a static IPv6 address on the interface, which also creates a link-local address for that interface.

```
R7#show ip int brief | include DHCP
Ethernet0/0.17             172.16.120.7    YES DHCP   up                    up
R7#show ipv6 dhcp interface
Ethernet0/0.17 is in client mode
  Prefix State is IDLE
  Address State is OPEN
  Renew for address will be sent in 11:57:48
  List of known servers:
    Reachable via address: FE80::A8BB:CCFF:FE00:100
    DUID: 00030001AABBCC000100
    Preference: 0
    Configuration parameters:
      IA NA: IA ID 0x00100001, T1 43200, T2 69120
        Address: 2005:DEAD:BEEF:120:0:D3AE:C4B5:23A4/128
                preferred lifetime 86400, valid lifetime 172800
                expires at Sep 27 2013 10:26 AM (172668 seconds)
      Information refresh time: 0
  Prefix Rapid-Commit: disabled
  Address Rapid-Commit: disabled
```

Here is what it looks like from the server point of view.

```
R1#show ip dhcp pool

Pool ToR7 :
 Utilization mark (high/low)    : 100 / 0
 Subnet size (first/next)       : 0 / 0
 Total addresses                : 254
 Leased addresses               : 1
 Pending event                  : none
 1 subnet is currently in the pool :
 Current index        IP address range                    Leased addresses
 172.16.120.8         172.16.120.1     - 172.16.120.254    1
R1#show ip dhcp binding
Bindings from all pools not associated with VRF:
IP address          Client-ID/              Lease expiration        Type
                    Hardware address/
                    User name
172.16.120.7        0063.6973.636f.2d61.    Sep 26 2013 10:05 AM    Automatic
                    6162.622e.6363.3030.
                    2e30.3730.302d.4574.
                    302f.302e.3137
R1#show ipv6 dhcp pool
DHCPv6 pool: ToR7
  Address allocation prefix: 2005:DEAD:BEEF:120::/80 valid 172800 preferred
86400 (1 in use, 0 conflicts)
  Active clients: 1
R1#show ipv6 dhcp binding
Client: FE80::A8BB:CCFF:FE00:700
  DUID: 00030001AABBCC000700
  Username : unassigned
  IA NA: IA ID 0x00100001, T1 43200, T2 69120
    Address: 2005:DEAD:BEEF:120:0:D3AE:C4B5:23A4
            preferred lifetime 86400, valid lifetime 172800
            expires at Sep 27 2013 10:26 AM (172538 seconds)
```