

Copyright Information

Copyright © 2003 - 2010 Internetwork Expert, Inc. All rights reserved.

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

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

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

Disclaimer

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

This workbook was developed by Internetwork Expert, Inc. Any similarities between material presented in this workbook and actual CCIE™ lab material is completely coincidental.

Table of Contents

| | |
|---|----|
| About VOL2..... | 2 |
| VOL2 Troubleshooting Lab 1..... | 4 |
| VOL2 Troubleshooting Lab 1 Solutions..... | 14 |
| VOL2 Configuration Lab 1 | 18 |
| VOL2 Configuration Lab 1 Solutions | 35 |

About VOL2

Internetwork Expert's CCIE Routing & Switching Lab Workbook Volume II (IEWB-RS-VOL2) is designed to be used as a supplement to other self-paced and instructor-led training materials in preparation for Cisco Systems' CCIE Routing & Switching Lab Exam. Please, refer to the company website <http://www.INE.com> for more information on additional products and product bundles.

IEWB-RS-VOL2 consists of various lab scenarios, designed from the ground up, based on Cisco Systems' newest specification for the CCIE Routing & Switching Lab Exam. The labs contained in IEWB-RS-VOL2 are designed to simulate the actual CCIE Routing & Switching Lab Exam and at the same time illustrate the principles behind the technologies that it covers.

VOL2 Troubleshooting Lab 1

Difficulty Rating (10 highest): 5

Lab Overview:

The following scenario is a practice lab exam designed to test your skills at configuring Cisco networking devices. Specifically, this scenario is designed to assist you in your preparation for Cisco Systems' CCIE Routing & Switching Lab exam. However, remember that in addition to being designed as a simulation of the actual CCIE lab exam, this practice lab should be used as a learning tool. Instead of rushing through the lab in order to complete all the configuration steps, take the time to research the networking technology in question and gain a deeper understanding of the principles behind its operation.

Lab Instructions:

Prior to starting, ensure that the initial configuration scripts for this lab have been applied. For a current copy of these scripts, see the Internetwork Expert members' site at <http://members.INE.com>. If you have any questions related to the scenario solutions, visit our CCIE support forum at <http://IEOC.com>.

Refer to the attached diagrams for interface and protocol assignments. Any reference to X in an IP address refers to your rack number, while any reference to Y in an IP address refers to your router number.

Upon completion, all devices should have full IP reachability to all networks in the routing domain, including any networks generated by the backbone routers unless explicitly specified.

Lab Do's and Don'ts:

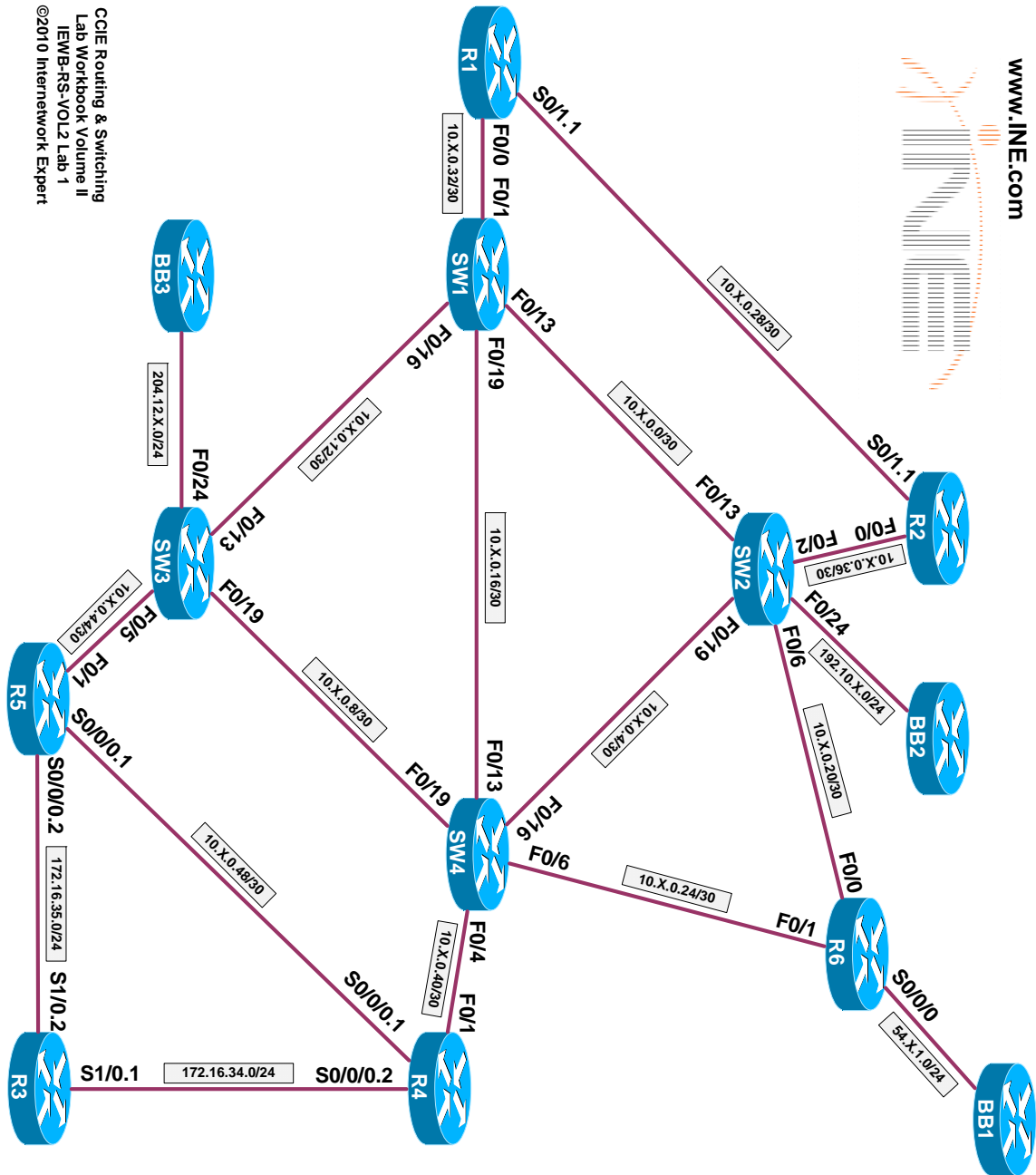
- Do not change or add any IP addresses from the initial configuration unless otherwise specified or required for troubleshooting
- If additional IP addresses are needed but not specifically permitted by the task use IP unnumbered
- Do not change any interface encapsulations unless otherwise specified
- Do not change the console, AUX, and VTY passwords or access methods unless otherwise specified
- Do not use any static routes, default routes, default networks, or policy routing unless otherwise specified
- Save your configurations often

Grading:

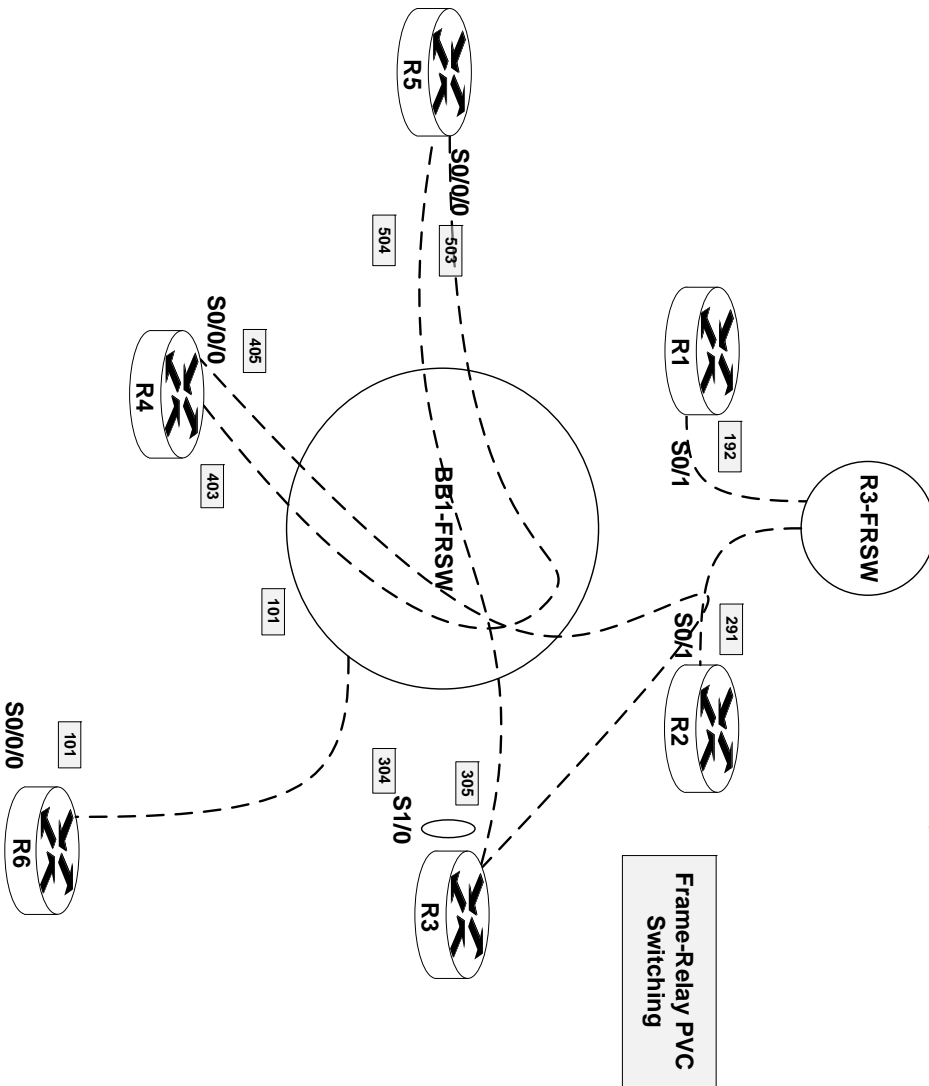
This practice lab consists of trouble tickets totaling 21 points. A score of 16 points is required to achieve a passing grade. A trouble ticket must be fixed 100% with the requirements given in order to be awarded the points for that ticket. No partial credit is awarded. If a ticket has multiple possible solutions, choose the solution that best meets the requirements and requires minimal changes. Per the CCIE R&S lab exam requirements, you are required to finish this lab in two hours.

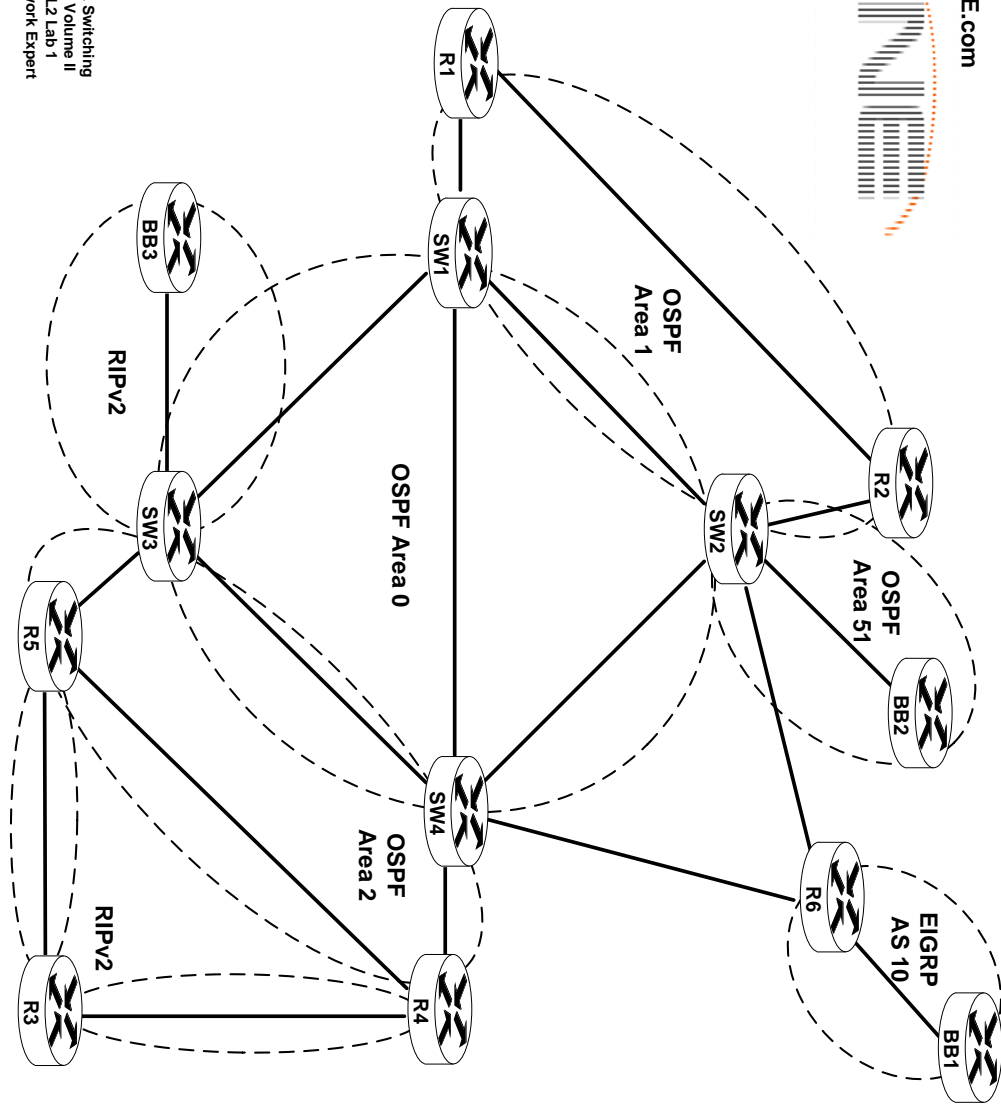
The tickets generally have no dependencies, unless explicitly stated in the ticket outline, so you may work through them in any order you like. It's up to you to select the tickets that you feel most easy to deal with and manage your time accordingly.

GOOD LUCK!

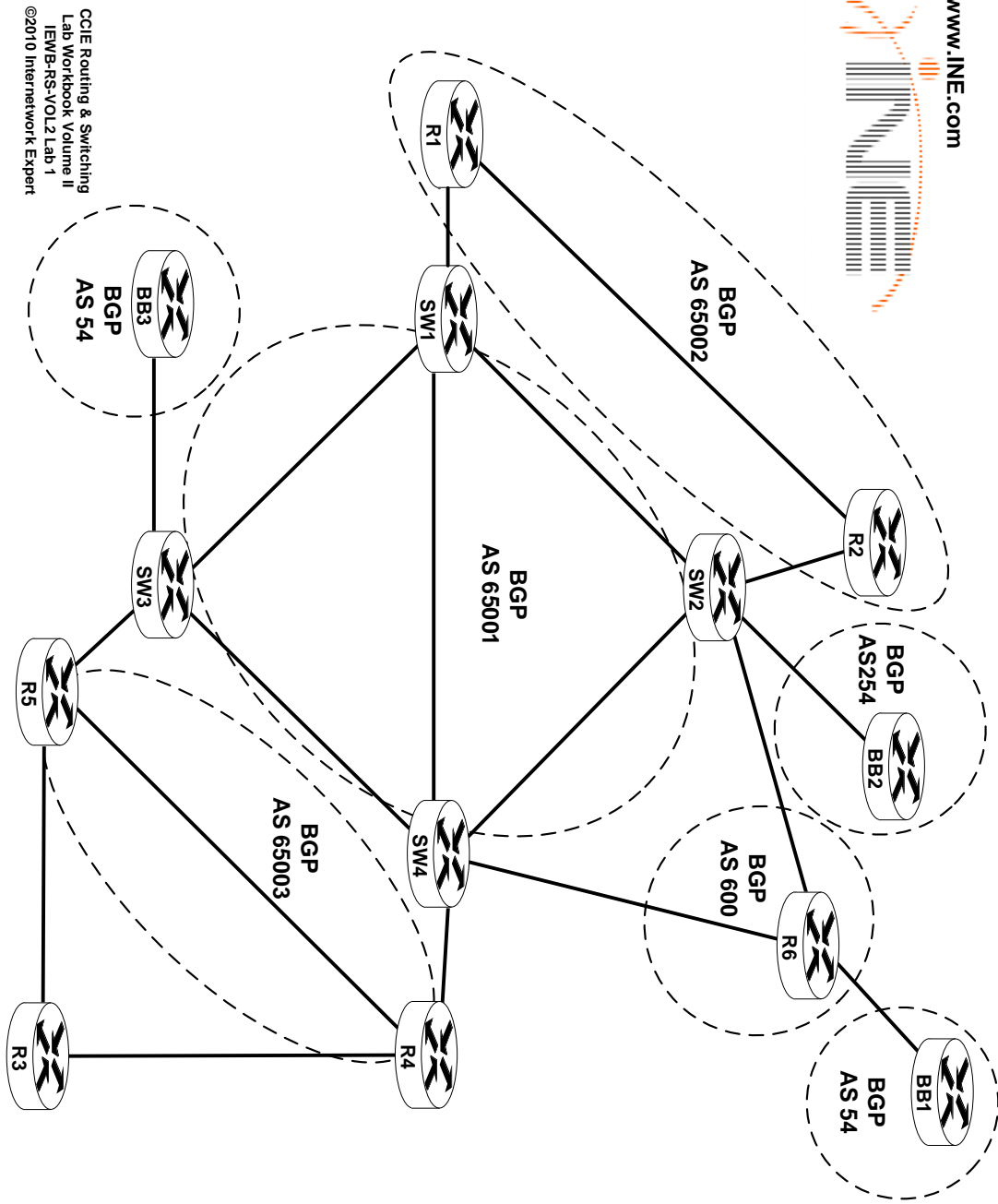
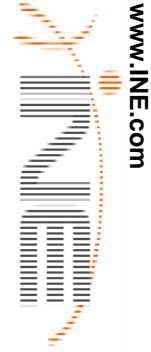


CCIE Routing & Switching
 Lab Workbook Volume II
 IEWB-RS-VOL2 Lab 1
 ©2010 Internetwork Expert





CCIE Routing & Switching
Lab Workbook Volume II
IEWR-RS-VOL2 Lab 1
©2010 Internetwork Expert



CCIE Routing & Switching
Lab Workbook Volume II
IEWB-RS-VOL2 Lab 1
©2010 Internetwork Expert

Ticket 1

- R1 and R2 cannot establish OSPF Adjacency.

2 Points**Ticket 2**

- AS 65001 cannot use the backup path via AS 600 to reach AS 54 when the direct connection between SW3 and BB3 fails.

2 Points**Ticket 3**

- R3 is a CE router connected to MPLS VPN PE routers R4 and R5. Recently, the Service Provider has been configured to provide Internet access to R3.
- However, a quick test shows that R3 is not able to reach the rest of the network (10.0.0.0/8). Fix this problem and restore connectivity.

3 Points**Ticket 4**

- The network administrator of BB3 reports that he does not see any routes from SW3.

2 Points

Ticket 5

- Resolve the problem preventing R4 from reaching SW2 over the shortest-path (based on hop-count).

2 Points**Ticket 6**

- Users from subnets 10.0.0.0/24 accessing Web servers behind BB1 should be redirected by R6 to the web-cache engines using Cisco's proprietary protocol.
- However the web-caches show no hits. Fix this problem by providing a working configuration.

2 Points**Ticket 7**

- R3 is not learning any VPN routing information from the PE router R5. RIPv2 is the routing protocol.
- Ensure R3 can reach R5's Loopback interface residing in the same VRF using the optimum path.

2 Points**Ticket 8**

- R1 is not learning any routes via SW1.

2 Points

Ticket 9

- Resolve the issues preventing management of R6 via HTTPS. Access credentials should be cisco/cisco and should be stored locally in R6.

2 Points**Ticket 10**

- Restore management access via telnet from R4 to R5.

2 Points

VOL2 Troubleshooting Lab 1 Solutions

Ticket 1 Solution

The problem is that R3 (configured to switch the PVC between R1 and R2) has an incorrect interface type on its connection to R2.

```
R3:
interface Serial1/2
  frame-relay intf-type dce
```

Ticket 2 Solution

The AS 65001 border router is not changing the BGP next-hop when advertising prefixes learned from AS 600. This results in the routes being treated as invalid in SW1's BGP table.

```
SW2:
router bgp 65001
  neighbor 10.1.7.7 next-hop-self
```

```
SW4:
router bgp 65001
  neighbor 10.1.7.7 next-hop-self
```

Ticket 3 Solution

R4 is set up for Internet access for the VPN, but the static default route in VRF345 is pointing to the wrong next hop.

```
R4:
no ip route vrf VPN345 0.0.0.0 0.0.0.0 10.1.0.40 global
ip route vrf VPN345 0.0.0.0 0.0.0.0 10.1.0.42 global
```

Ticket 4 Solution

SW3 is set to send unicast updates to BB3, but the neighbor statement points to the wrong IP address.

```
SW3:
router rip
  passive-interface default
  neighbor 204.12.1.254
```

Ticket 5 Solution

R4's OSPF adjacency with SW4 is broken because the dead-interval is set to the wrong value on SW4.

```
SW4:
interface Fa0/1
no ip ospf dead-interval 41
```

Ticket 6 Solution

The problem is that WCCP is configured for dynamic service number 80, not the WWW service.

```
R6:
ip wccp web-cache
!
interface FastEthernet 0/0
  ip wccp web-cache redirect in
  no ip wccp 80 redirect in
!
interface FastEthernet 0/1
  ip wccp web-cache redirect in
  no ip wccp 80 redirect in
!
access-list 100 permit ip 10.0.0.0 0.0.0.255 any
!
ip wccp web-cache redirect-list 100
```

Ticket 7 Solution

R5 is configured to send RIPv1 only routes to R3.

```
R5:
interface Serial 0/0/0.2
  no ip rip send version 1
```

Ticket 8 Solution

R1 is not learning any routes via SW1 because of an OSPF database filter set up in SW1:

```
SW1:
interface FastEthernet 0/1
 no ip ospf database-filter all out
```

Ticket 9 Solution

The HTTP port is set to the wrong value; and the access-list controlling HTTP access has a deny entry at the beginning:

```
R6:
ip access-list standard 98
 no deny any
 !
ip http secure-port 443
 !
ip http authentication local
```

Ticket 10 Solution

Control plane policing is set up in R5 that is designed to drop fragmented traffic. However, Telnet traffic is also being dropped.

```
R5:
ip access-list extended FRAGMENTS
 no permit tcp any any eq 23
```


Volume 2 Configuration Lab 1

Difficulty Rating (10 highest): 6

Lab Overview:

The following scenario is a practice lab exam designed to test your skills at configuring Cisco networking devices. Specifically, this scenario is designed to assist you in your preparation for Cisco Systems' CCIE Routing & Switching Lab exam. However, remember that in addition to being designed as a simulation of the actual CCIE lab exam, this practice lab should be used as a learning tool. Instead of rushing through the lab in order to complete all the configuration steps, take the time to research the networking technology in question and gain a deeper understanding of the principles behind its operation.

Lab Instructions:

Prior to starting, ensure that the initial configuration scripts for this lab have been applied. For a current copy of these scripts, see the Internetwork Expert members' site at <http://members.INE.com>. If you have any questions related to the scenario solutions, visit our CCIE support forum at <http://IEOC.com>.

Refer to the attached diagrams for interface and protocol assignments. Any reference to X in an IP address refers to your rack number, while any reference to Y in an IP address refers to your router number.

Upon completion, all devices should have full IP reachability to all networks in the routing domain, including any networks generated by the backbone routers unless explicitly specified.

Lab Do's and Don'ts:

- Do not change or add any IP addresses from the initial configuration unless otherwise specified or required for troubleshooting
- If additional IP addresses are needed but not specifically permitted by the task use IP unnumbered
- Do not change any interface encapsulations unless otherwise specified
- Do not change the console, AUX, and VTY passwords or access methods unless otherwise specified
- Do not use any static routes, default routes, default networks, or policy routing unless otherwise specified
- Save your configurations often

Grading:

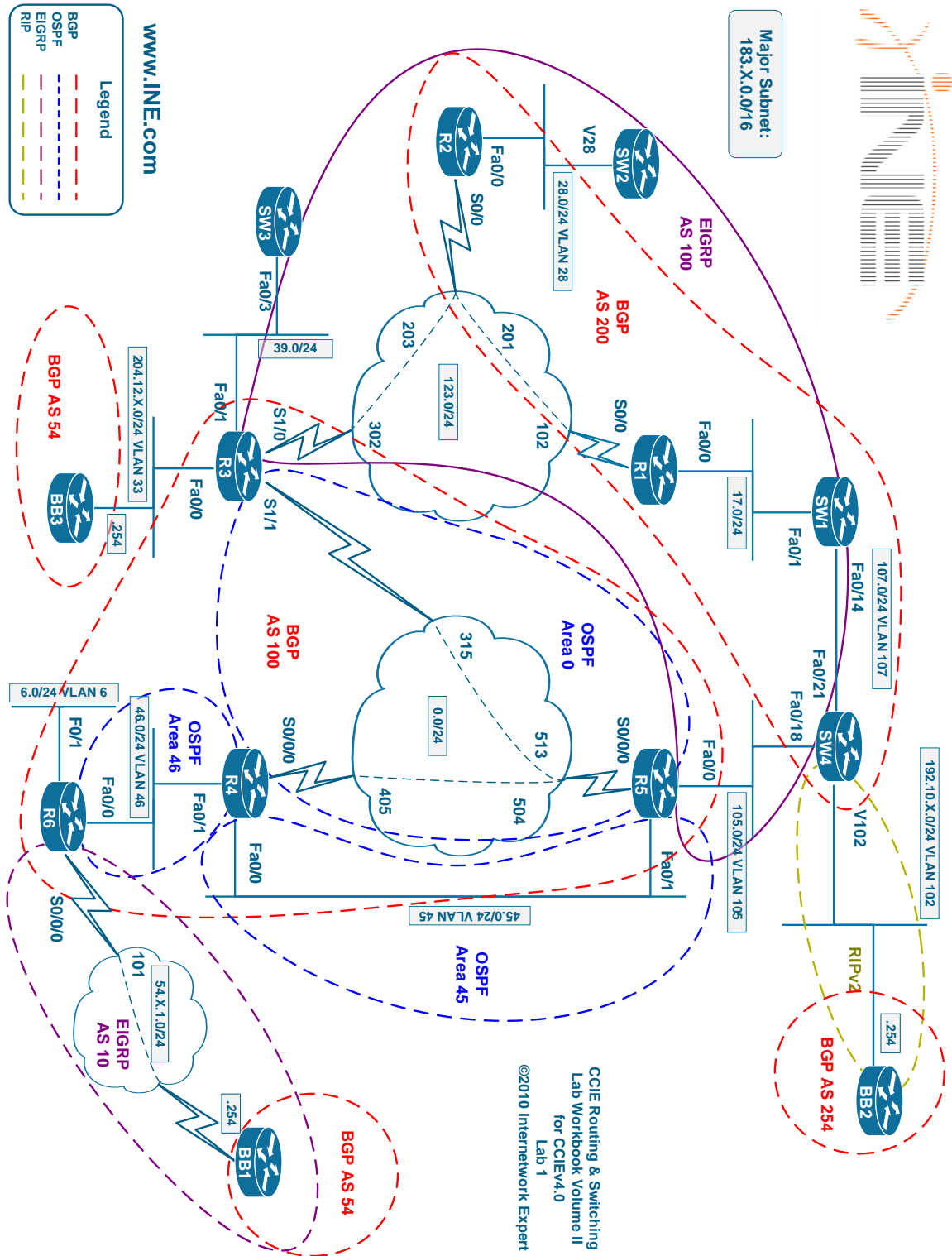
This practice lab consists of various sections totaling 79 points. A score of 64 points is required to pass the exam. A section must work 100% with the requirements given in order to be awarded the points for that section. No partial credit is awarded. If a section has multiple possible solutions, choose the solution that best meets the requirements.

Point Values:

The point values for each section are as follows:

| Section | Point Value |
|----------------------|-------------|
| Layer 2 Technologies | 4 |
| IPv4 | 12 |
| IPv6 | 7 |
| MPLS VPN | 6 |
| Multicast | 7 |
| Security | 10 |
| Network Services | 21 |
| QoS | 12 |

GOOD LUCK!



1. Layer 2 Technologies

Some Layer 2 settings have been pre-configured for you; VLANs have been created and assigned to the switchports according to the diagram supplied with the task.

1.1 Layer 2 Features

- Ensure SW2 is the primary STP root bridge for VLAN105
- Configure the network in such a way to ensure that VLAN 102's traffic never traverses SW3.
- Ports Fa0/7 on SW1 and Fa0/7 on SW2 belong to VLAN 28.
- Configure SW1 and SW2 so that users in VLAN 28 do not have to wait for spanning-tree's forwarding delay when they connect to the network.
- Ensure that any ports in VLAN 28 will be shut down if a device running spanning-tree protocol is detected.
- Additionally, the same ports should not be able to communicate directly with each other within VLAN 28.
- These ports should still be allowed to communicate with R2's VLAN28 interface but not SW2's V28 interface.
- You are not allowed to use VLAN ACLs to accomplish this, but you are allowed to create and use additional VLAN 281.

4 Points

2. IPv4

Some IGP protocol settings and IP addressing have been preconfigured for you. Notice that there might be some issues deliberately introduced into the initial configurations. Use the diagram as you reference to fix those.

2.1 OSPF

- Ensure that R5 is always elected the Designated Router for the segment between R3, R4 and R5.
- Ensure that other devices running OSPF on the segment between R4 and R5 cannot intercept the OSPF communication between R4 and R5.
- Advertise VLAN 6 into OSPF on R6; but do not use the **network** or **ip ospf** statements to accomplish this.
- Configure the network so that traffic is only sent over the VLAN45 link if the Frame Relay circuit between R4 and R5 is down. Do not use the **backup interface** command to accomplish this.
- To minimize downtime in the event of a failure configure the network so that R4 can detect a loss of the Frame Relay circuit to R5 within 1 second

4 Points

2.2 IGP Features

- Advertise VLAN 33 and R3's interface connected to SW3 into the EIGRP domain.
- These prefixes should appear as follows throughout the EIGRP domain:

```
D EX    204.12.X.0 [170/...  
D EX    183.X.39.0 [170/...
```

- In order to ensure that all routes learned over the Frame Relay cloud via EIGRP are legitimate, configure R6 to use the most secure authentication for any neighbor relationships formed on this interface.
- Use key number 1 with a password of CISCO for this authentication.
- In order to protect against false route injection from RIP as well, configure SW4 to use the strongest authentication on any RIP updates received on this Ethernet segment using key 1 and the password CISCO.
- Redistribute between RIP and EIGRP on SW4.
- Redistribute between OSPF and EIGRP on R3, R5, and R6.

3 Points

2.3 BGP Bestpath Selection

- For the purposes of load-sharing and redundancy, AS 100 has multiple connections to AS 54. In order to maximize throughput your corporate policy dictates that all traffic destined for prefixes originated in AS 54 should traverse the Frame Relay link between R6 and BB1.
- In the case that the Frame Relay link between R6 and BB1 goes down AS 100 should still have reachability to AS 54 via the Ethernet segment between R3 and BB3.
- Do not modify weight to accomplish the above requirements.
- Configure a new Loopback interface on R1 with the IP address 150.X.11.1/24 and advertise it into BGP.
- Configure AS 200 so that all traffic from AS 100 destined to this prefix traverses the Ethernet segment between SW4 and R5.
- In the case that the link between SW4 and R5 is down traffic destined for the 150.X.11.0/24 prefix should transit the Frame Relay link between R2 and R3.
- Do not use AS-PATH prepending to accomplish this.

4 Points

3. IPv6

3.1 IPv6 Addressing

- The network administrator has requested that VLAN 46 and VLAN 105 be configured to support a test deployment of IPv6.
 - Address R4's interface attached to VLAN 46 with the IPv6 network 2001:CC1E:X:404::/64.
 - Address R5's interface attached to VLAN 105 with the IPv6 network 2001:CC1E:X:505::/64.
- The host addresses on these interfaces should be derived from the interface's MAC address.
- Configure a tunnel between R4 and R5 using their Loopback0 interfaces as the source.
- The tunnel should use the addresses 2001:CC1E:X:4545::Y/64.
- This tunnel should use a mode that specifies IPv6 as the passenger protocol and IPv4 as the encapsulation and transport protocol.
- Enable EIGRPv6 on VLAN 46, VLAN 105 and the tunnel interfaces.
- Use 45 as the AS# for the processes on both R4 and R5.

4 Points

3.2 IPv6 Multicast Basics

- Configure R6 to join the multicast group FF06::6 on its connection to R4.
- Configure R4 to accept MLD messages only for the group range FF06::/16.
- Use R5 as the PIM RP and ensure multicast packets from R5 can reach R6.
- Do not configure the RP address statically and do not configure the same router as a BSR and RP.

3 Points

4. MPLS VPN

4.1 LDP

- Configure MPLS label redistribution between R4, R5 and R6 using the industry-standard protocol.
- Make sure the labels are redistributed even if the primary Frame-Relay link between R4 and R5 fails.
- Configure so that the labels are only generated for the respective router's Loopback0 interfaces.

3 Points

4.2 VPN

- Using the RD 100:5 and 100:6 configure two VRFs named VPN_A and VPN_B in R5 and R6 respectively.
- Use the same route-target values to tag the respective routes.
- Create two new Loopback interfaces in R5 and R6 with the IP addresses 172.16.5.5/24 and 192.168.6.6/24 and assign them to VRFs VPN_A and VPN_B respectively.
- Configure R5 and R6 to provide reachability between the two subnets.

3 Points

5. IP Multicast

5.1 RP Assignment

- Discover the active multicast topology using the respective show commands.
- Configure R3 to announce its most reliable interface as the RP for all multicast groups using Auto-RP protocol.
- R2 should be responsible for group to RP mappings.

2 Points

5.2 Multicast Testing

- There is a Windows® Media Server located on VLAN 28 that is streaming a video feed into your network, however your administrators have been getting complaints from users on VLAN 105 that they are unable to receive this feed.
- In order to help track down the source of this problem configure R5's Ethernet interface attached to VLAN 105 to join the multicast group 226.26.26.26.
- Ensure that R5 responds to ICMP echo-requests sourced from VLAN 28 which are sent to 226.26.26.26.
- You are allowed to use one static multicast route to accomplish this.

3 Points

5.3 Multicast Filtering

- Development engineers are testing a new multicast application located on VLAN 28 prior to its deployment in your network. This application is generating random multicast streams destined for addresses in the administratively scoped multicast range.
- In order to prevent this test traffic from being unnecessarily forwarded throughout the network, configure R3 so that hosts in VLAN 33 are not allowed to join any groups in this range.

2 Points

6. Security

6.1 Denial of Service Tracking

- Your network administrators have been getting complaints from users that the web server with the IP address 183.X.28.100 is inaccessible. After further investigation you have determined that this server is undergoing a TCP SYN attack.
- In order to assist in tracking down the source of this attack configure R3 and SW4 to generate a log message when HTTP SYN packets are received on VLANs 33 or 102 respectively that are destined for 183.X.28.100.
- These log messages should include the MAC address of the device which forwarded the packet onto the segment.

3 Points

6.2 Spoof Prevention

- After reviewing your log files you have determined that the DoS attack on your web server came from hosts with spoofed source addresses.
- To help prevent this type of attack in the future configure your network so that traffic will not be accepted from BB1, BB2, or BB3 if it sourced from your address space 183.X.0.0/16.

2 Points

6.3 Information Leaking

- Your security manager is concerned with potential network reconnaissance attacks originating from behind BB2.
- In order to minimize information exposure, configure SW4 not to notify hosts behind BB2 of any networks that it does not know about.
- Additionally, SW4 should not disclose its network mask to any host on the VLAN 102 segment.

2 Points

6.4 Control Plane Protection

- Configure R4 to drop any outgoing IP packets with the TTL lower than 3.
- Log the drop events to the system console and the router's memory buffer.

3 Points

7. Network Services

7.1 RMON

- In order to help detect possible flood attacks in the future configure R2 to generate an SNMP trap when the MIB value ifEntry.11.1 rises more than 15000 per minute, and when the value falls back below 5000 per minute.
- The sampling interval should be every sixty seconds.
- When the 15000 threshold is breached an event should be generated that reads "Above 15000 for ifInUcastPkts".
- When the value falls back to 5000 an event should be generated that reads "Below 5000 for ifInUcastPkts".
- The server to send these SNMP traps to is 183.X.17.100.
- This server will be expecting the community string to be IETRAP.

3 Points

7.2 NTP

- After implementing syslog logging your NOC engineers have noticed inconsistent timestamps on your device logs. In order to resolve this problem you have decided to maintain consistent time by implementing Network Time Protocol.
 - Configure R3 and R6 to get network time from BB3 and BB1 respectively.
 - Configure R1, R2, and SW1 to get network time from R3.
 - Configure R4, R5, and SW4 to get network time from R6.
- R1 and R2 should cooperatively coordinate their clocks and adjust each other's time.

2 Points

7.3 NTP Authentication

- In order to ensure that your internal time servers are not being spoofed, configure R3 and R6 to be authenticated using the MD5 password CISCO.
- Make sure the NTP peering session between R1 and R2 is authenticated as well.

3 Points

7.4 Traffic Accounting

- Your design team would like to implement a new QoS policy using IP precedence on the Frame Relay circuit between R2 and R3. However, prior to implementing this new policy they need to know if packets transiting this link already have an IP precedence value set.
- To accomplish this configure R2 and R3 to collect usage statistics on packets with an IP precedence value and store them locally.
- R2 and R3 should store up to 50000 of these entries in their memory.

3 Points

7.5 Gateway Redundancy

- Your administrators are concerned about default gateway redundancy for the hosts located on VLAN 105. In order to allow them to survive a network failure you have assigned the virtual IP address 183.X.105.254 as the default gateway for these hosts.
- As long as R5's Frame Relay physical connection is up it should respond to ARP requests sent to this IP address.
- In the event that R5's Frame Relay port goes down hosts should use SW4 as their default gateway.
- Do not use VRRP or GLBP to accomplish this.
- Configure your network to reflect this policy.

3 Points

7.6 Network Address Translation

- Your operations team does not want BB3 and its customers to have specific reachability information about your network. Instead, BB3 should only have reachability to your hosts if a connection is initiated from inside your network.
- Configure R3 to reflect this policy.
- Ensure that all devices in the 183.X.0.0/16 network can successfully ping BB3.

3 Points

7.7 Embedded Event Management

- Configure R6 to generate a syslog message once its Frame-Relay output utilization exceeds 80% of the total interface bandwidth.
- The sylog message should contain the interface name and the current interface load.
- Ensure your configuration responds to the transmit load changes as fast as possible with Cisco IOS routers.
- Do not use the RMON feature to accomplish this.

4 Points

8. QoS

8.1 Frame Relay Traffic Shaping

- You have been noticing drops on R5's connection to the Frame Relay cloud. After further investigation, you have discovered that R5 has been overwhelming R3 and R4's connections to the Frame Relay cloud. Configure Frame Relay Traffic Shaping on R5 in order to resolve this issue.
- R5's connection to the Frame Relay cloud supports a transmission rate of 1536Kbps.
- R5 should send at an average rate of 128Kbps on DLCI 513 to R3.
- R5 should send at an average rate of 512Kbps on DLCI 504 to R4.
- In the case that the Frame Relay cloud notifies R5 of congestion it should reduce its sending rate to no lower than 96Kbps for the DLCI to R3 and 384Kbps for the DLCI to R4.
- In the case that R5 has accumulated credit it should be allowed to burst up to the maximum transmission rate supported on the circuit to R4.
- Bursting on the circuit to R3 should not be allowed.
- Assume an interval (Tc) of 50ms.

3 Points

8.2 Rate Limiting

- One of your NOC engineers has noticed suspiciously high utilization on the Ethernet segment of R1. After further investigation you have found that a large number of ICMP packets have been traversing this link.
- In order to alleviate congestion configure R1 so that it does not send more than 128Kbps of ICMP traffic out this interface.
- Allow for a burst of 1/4th of this rate.

3 Points

8.3 CBWFQ

- Your company plans to reduce expenses by sending PSTN calls to the remote office connected to R5 across the WAN. Currently the WAN link is used primarily for data transfers and remote desktop application.
- Configure R5 to allocate 64Kbps of PVC bandwidth to VoIP bearer traffic, which is marked as DSCP EF.
- At the same time, guarantee 30% of remaining bandwidth to Citrix application traffic.
- Set the queue depth for the Citrix traffic class to 16 packets.
- All other remaining traffic should receive flow-based fair scheduling.

3 Points

8.4 Catalyst QoS

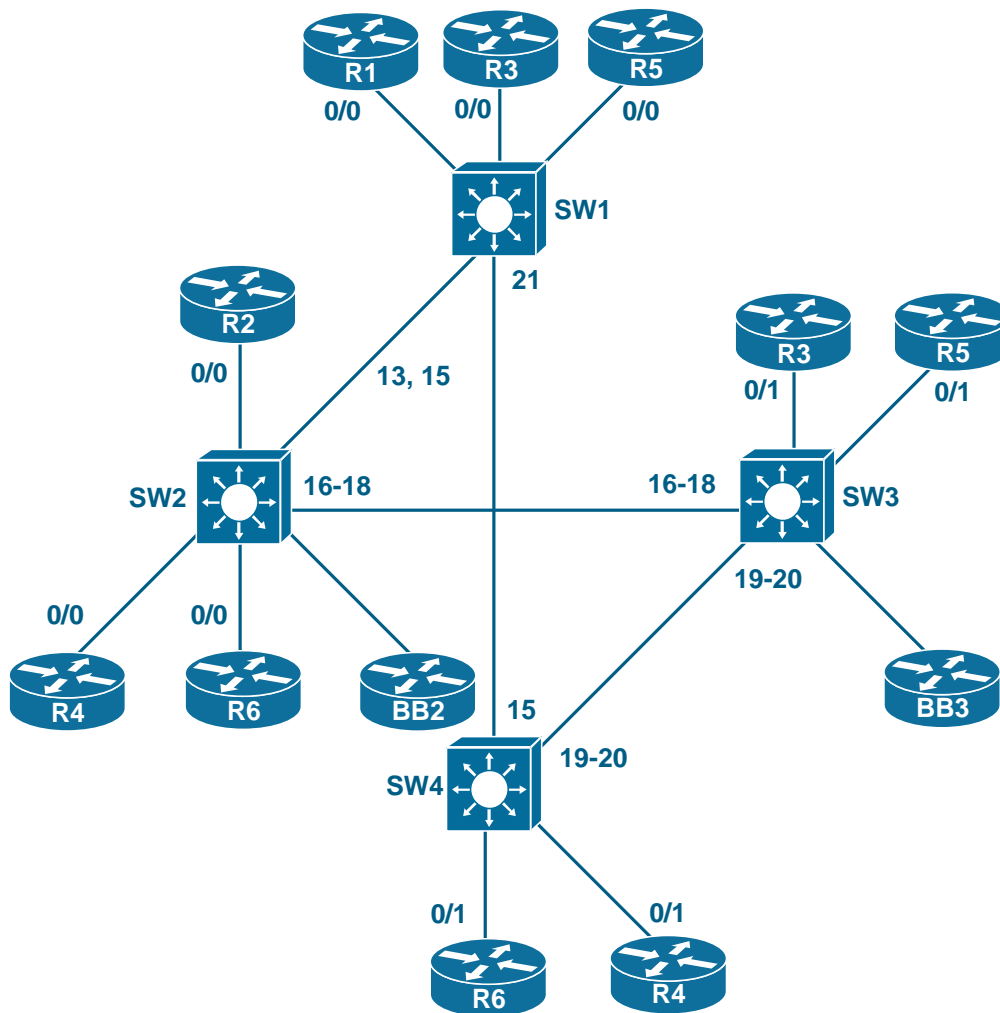
- R6 should see OSPF packets sent from R4 marked with IP Precedence value of 3.
- Limit the aggregate traffic rate for packets sent to R6 VLAN 46 interface to 4 Mbps but ensure the exceeding packets are queued.
- At the same time, traffic flows generated by HTTP server responses should be limited to 1Mbps on this connection.
- Use DSCP value of CS2 to mark the HTTP packets for this task.
- Do not configure any of the routers to accomplish this.

3 Points

VOL2 Configuration Lab 1 Solutions

Preliminary Tasks

Before you start with the Layer 2 technologies section, you need to discover the existing Layer 2 topology. The easiest way to accomplish this is to use the `show cdp neighbor`, `show interfaces` and `show interfaces trunk` commands and put this information on a physical connections diagram. Below is a sample active connections diagram which lists the trunk interface numbers. The ports connecting the routers have numbers matching the router numbers.



Further Learning

The only real way to improve your skills for diagramming is practicing as many diagrams as possible. With every VOL2 lab, make your own Layer 3 topology diagrams, in addition to Layer 2 topology and BGP diagrams (as discussed later in the workbook). It takes some time to be able to quickly translate the show command output into a corresponding diagram.

Task 1.1 Solution

Before You Begin

The solution utilizes Private VLAN technology, which you need to be familiar with prior to starting. For an in-depth explanation of Private VLAN technology you should study VOL1 scenario **Bridging and Switching > Private VLANs** or read the blog post titled “Private VLANs Revisited”

<http://blog.ine.com/2008/07/14/private-vlans-revisited/>

on the INE blog.

A trick used in the solution is spanning-tree manipulation by means of VLAN filtering. This does not involve using any new technology, but it does require proper interpretation of the Layer 2 diagram made previously. Lastly, the benefits and use of the STP BPDU Guard feature is explained in the VOL1 scenario entitled **STP BPDU Guard**. You may want to take this opportunity to study other STP features as well, as they are co-related.

SW2:

```
!  
! Make SW2 the root for VLAN 105  
!  
spanning-tree vlan 105 root primary
```

SW1 and SW2:

```
!  
! BPDU Guard will shutdown the port upon BPDU detection  
!  
interface FastEthernet0/7  
  spanning-tree portfast  
  spanning-tree bpduguard enable
```

SW2:

```
!  
interface FastEthernet0/2  
  spanning-tree bpduguard enable
```

 **Note**

In this case, traffic engineering is achieved by pruning a VLAN on select trunks, to make sure VLAN 102 does not go across SW3. All other VLANs are not affected by this policy.

SW2:

```
!  
! These links go to SW3, so we exclude VLAN 102  
!  
interface range Fa0/16 - 18  
  switchport trunk allowed vlan except 102
```

SW3:

```
!  
! All links on SW3 are stripped off VLAN 102  
!  
interface range Fa0/16 - 20  
  switchport trunk allowed vlan except 102
```

SW4:

```
!  
! The links to SW3 may carry all VLANs but not 102  
!  
interface range Fa0/19 - 20  
  switchport trunk allowed vlan except 102
```

 **Note**

Here we see the Private VLAN configuration section – ports may be segregated using a single secondary isolated VLAN. Notice that VTP is already set to Transparent mode, which allows for Private VLAN creation.

SW1:

```
!  
! Create the secondary VLAN  
!  
vlan 281  
    private-vlan isolated  
!  
! Make VLAN 28 primary and associate with secondary  
!  
vlan 28  
    name VLAN_28  
    private-vlan primary  
    private-vlan association 281  
!  
! Configure the host port using the primary/secondary pair 28/281  
!  
interface FastEthernet0/7  
    switchport private-vlan host-association 28 281  
    switchport mode private-vlan host
```

SW2:

```
vlan 281  
    private-vlan isolated  
!  
vlan 28  
    name VLAN_28  
    private-vlan primary  
    private-vlan association 281  
  
!  
! R2 is the router and thus belongs to the promiscuous port  
!  
interface FastEthernet0/2  
    switchport private-vlan mapping 28 281  
    switchport mode private-vlan promiscuous  
!  
interface FastEthernet0/7  
    switchport private-vlan host-association 28 281  
    switchport mode private-vlan host  
  
!  
! We do not apply the private VLAN settings for the SVI in SW2  
! and thus the hosts ports will not be able to reach it  
!
```

Task 1.1 Breakdown

The requirement to make SW2 the root for VLAN 105 translates into setting it as a primary bridge for this VLAN. The requirement to minimize forwarding delays when the port comes up translates into using the STP PortFast feature. The other requirement to shutdown the port upon receiving a BPDU translates into using the BPDU Guard feature. A normal PortFast port without BPDU Filtering or guarding enabled will revert to sending BPDUs and lose its PortFast status upon receiving a single BPDU. While BPDU Guard can be enabled either at the global level or the interface level, the global command will apply the feature only to PortFast enabled ports. Since the interface Fa0/2 on SW2 is not PortFast enabled, it makes more sense to use the port level command.

To correctly interpret the part about VLAN 102's traffic engineering, you need to find out the route that VLAN 102 should be taking across the physical topology. This is where you need the diagram you made before starting this section. Per the logical diagram, VLAN 102 connects SW4 to BB2. Per the physical connectivity diagram, BB2 connects to SW2. Thus, there are two paths from BB2 to SW2:

- 1) SW2 -> SW1 -> SW4
- 2) SW2 -> SW3 -> SW4

And the task requires that the second path should never be used. The fact that it should *never* be used eliminates any STP manipulations and leaves us with the VLAN filtering option. Specifically, the solution ensures that VLAN 102 is deleted from all trunks connecting to SW3.

The last part is concerned with private VLANs, and the configuration is straightforward for this scenario.



Deep Dive

- Why would you avoid the "VLAN stripping" from trunks in real-world scenarios?
- Re-work this scenario to use STP cost manipulation for preferred path engineering.
- If required to maximize link bandwidth usage, what alternative to STP might you use to provide redundancy and bandwidth optimization?
- Why would you prefer BPDU Guard over BPDU Filter at the edge of your network?

Task 1.1 Verification

Verify the Private VLAN configuration:

```
Rack1SW1#show interfaces fa0/7 switchport | include private|28|281
Administrative Mode: private-vlan host
Administrative private-vlan host-association: 28 (VLAN_28) 281
(VLAN0281)
Administrative private-vlan mapping: none
Administrative private-vlan trunk native VLAN: none
Administrative private-vlan trunk Native VLAN tagging: enabled
Administrative private-vlan trunk encapsulation: dot1q
Administrative private-vlan trunk normal VLANs: none
Administrative private-vlan trunk private VLANs: none
Operational private-vlan: none
```

```
Rack1SW2#show interfaces fa0/2 switchport | include private|28|281
Administrative Mode: private-vlan promiscuous
Operational Mode: private-vlan promiscuous
Administrative private-vlan host-association: none
Administrative private-vlan mapping: 28 (VLAN_28) 281 (VLAN0281)
Administrative private-vlan trunk native VLAN: none
Administrative private-vlan trunk Native VLAN tagging: enabled
Administrative private-vlan trunk encapsulation: dot1q
Administrative private-vlan trunk normal VLANs: none
Administrative private-vlan trunk private VLANs: none
Operational private-vlan:
  28 (VLAN_28) 281 (VLAN0281)
```

```
Rack1SW2#show interfaces fa0/7 switchport | include private|28|281
Administrative Mode: private-vlan host
Administrative private-vlan host-association: 28 (VLAN_28) 281
(VLAN0281)
Administrative private-vlan mapping: none
Administrative private-vlan trunk native VLAN: none
Administrative private-vlan trunk Native VLAN tagging: enabled
Administrative private-vlan trunk encapsulation: dot1q
Administrative private-vlan trunk normal VLANs: none
Administrative private-vlan trunk private VLANs: none
Operational private-vlan: none
```

For testing purposes, we will temporarily change R6's Fa0/0 IP address and VLAN to facilitate the test.

```
Rack1SW2#show running-config interface fa0/6
Building configuration...

Current configuration : 117 bytes
!
interface FastEthernet0/6
  switchport private-vlan host-association 28 281
  switchport mode private-vlan host
end
```

```
Rack1R6#show running-config interface Fa0/0
```

```
Building configuration...
```

```
Current configuration : 98 bytes
```

```
!
```

```
interface FastEthernet0/0
```

```
 ip address 183.1.28.6 255.255.255.0
```

```
end
```

```
Rack1R6#ping 183.1.28.2
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 183.1.28.2, timeout is 2 seconds:
```

```
!!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

```
Rack1R6#ping 183.1.28.8
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 183.1.28.8, timeout is 2 seconds:
```

```
.....
```

```
Success rate is 0 percent (0/5)
```

```
Rack1SW2#ping 183.1.28.2
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 183.1.28.2, timeout is 2 seconds:
```

```
!!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```

```
Rack1SW2#ping 183.1.28.6
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 183.1.28.6, timeout is 2 seconds:
```

```
.....
```

```
Success rate is 0 percent (0/5)
```

Verify that BPDU Guard is enabled on SW2 Fa0/2:

```
Rack1SW2#show spanning-tree interface fastEthernet 0/2 detail
```

```
Port 4 (FastEthernet0/2) of VLAN0028 is designated forwarding
```

```
Port path cost 19, Port priority 128, Port Identifier 128.4.
```

```
Designated root has priority 32796, address 0018.738d.9e00
```

```
Designated bridge has priority 32796, address 0018.738d.9e00
```

```
Designated port id is 128.4, designated path cost 0
```

```
Timers: message age 0, forward delay 0, hold 0
```

```
Number of transitions to forwarding state: 1
```

```
Link type is point-to-point by default
```

```
Bpdu guard is enabled
```

```
BPDU: sent 1052, received 0
```

Verify STP settings; for example, confirm that SW2 is the root for VLAN 105. Notice that you cannot verify the PortFast setting using the show commands as both Fa0/7 ports are inactive.

```
Rack1SW2#show spanning-tree vlan 105
```

```
VLAN0105
Spanning tree enabled protocol ieee
Root ID    Priority    24681
           Address    0018.738d.9e00
           This bridge is the root
           Hello Time 2 sec  Max Age 20 sec  Forward Delay 15 sec

Bridge ID  Priority    24681 (priority 24576 sys-id-ext 105)
           Address    0018.738d.9e00
           Hello Time 2 sec  Max Age 20 sec  Forward Delay 15 sec
           Aging Time 300
```

| Interface | Role | Sts | Cost | Prio.Nbr | Type |
|-----------|------|-----|------|----------|------|
| Fa0/13 | Desg | FWD | 19 | 128.15 | P2p |
| Fa0/15 | Desg | FWD | 19 | 128.17 | P2p |
| Fa0/16 | Desg | FWD | 19 | 128.18 | P2p |
| Fa0/17 | Desg | FWD | 19 | 128.19 | P2p |
| Fa0/18 | Desg | FWD | 19 | 128.20 | P2p |
| Fa0/21 | Desg | FWD | 19 | 128.23 | P2p |

```
Rack1SW2#show spanning-tree vlan 105 root cost
```

```
VLAN0105 0
```

Note

Verify the traffic-engineering status. VLAN 102 should follow the path across SW1 and should not appear on any trunks connected to SW3. Check the spanning tree for VLAN 102 to ensure this.

```
Rack1SW1#show spanning-tree vlan 102
```

```
VLAN0102
Spanning tree enabled protocol ieee
Root ID    Priority    32870
           Address    000f.f769.3a80
           Cost      19
           Port      23 (FastEthernet0/21)
           Hello Time 2 sec  Max Age 20 sec  Forward Delay 15 sec

Bridge ID  Priority    32870 (priority 32768 sys-id-ext 102)
           Address    001e.bdaa.ba80
           Hello Time 2 sec  Max Age 20 sec  Forward Delay 15 sec
           Aging Time 300
```

| Interface | Role | Sts | Cost | Prio.Nbr | Type |
|-----------|------|-----|------|----------|------|
| Fa0/13 | Desg | FWD | 19 | 128.15 | P2p |
| Fa0/15 | Desg | FWD | 19 | 128.17 | P2p |
| Fa0/21 | Root | FWD | 19 | 128.23 | P2p |

Rack1SW2#show spanning-tree vlan 102

VLAN0102

Spanning tree enabled protocol ieee

Root ID Priority 32870
 Address 000f.f769.3a80
 Cost 38
 Port 15 (FastEthernet0/13)
 Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec

Bridge ID Priority 32870 (priority 32768 sys-id-ext 102)
 Address 0019.5684.3700
 Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec
 Aging Time 300

| Interface | Role | Sts | Cost | Prio.Nbr | Type |
|-----------|------|-----|------|----------|------|
| Fa0/13 | Root | FWD | 19 | 128.15 | P2p |
| Fa0/15 | Altn | BLK | 19 | 128.17 | P2p |
| Fa0/24 | Desg | FWD | 100 | 128.26 | Shr |

```
Rack1SW3#show spanning-tree vlan 102
```

```
Spanning tree instance(s) for vlan 102 does not exist.
```

```
Rack1SW4#show spanning-tree vlan 102
```

```
VLAN0102
```

```
Spanning tree enabled protocol ieee
```

```
Root ID      Priority      32870
Address      000f.f769.3a80
This bridge is the root
Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
```

```
Bridge ID   Priority      32870 (priority 32768 sys-id-ext 102)
Address     000f.f769.3a80
Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec
Aging Time 300
```

| Interface | Role | Sts | Cost | Prio.Nbr | Type |
|-----------|------|-----|------|----------|------|
| Fa0/15 | Desg | FWD | 19 | 128.15 | P2p |

Lastly, verify that you can ping BB2 from SW4. This is to ensure that VLAN filtering did not break connectivity.

```
Rack1SW4#ping 192.10.1.254
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 192.10.1.254, timeout is 2 seconds:
!!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/4 ms
```

Task 2.1 Solution

Before You Begin

The solution uses the OSPF non-broadcast network type, OSPF priority manipulation, and virtual links to repair a discontinuous area. Also, the OSPF configuration requires sub-second timers. If you feel you are missing some information on these topics, you may need to study fundamental OSPF scenarios from VOL1 such as **OSPF > OSPF over Non-Broadcast Media**. These lessons provide more information about running OSPF on top of NBMA networks. The sub-second hello-timers are covered under the same VOL1 section as the scenario named **OSPF Interface Timers** and if you need more information about the use of virtual links in backup scenarios check the OSPF scenario named **OSPF Path Selection with Non-Backbone Transit Areas and Repairing Discontinuous OSPF Areas with Virtual links**. If you find that you need to work with any of the above mentioned scenarios, this probably means you need to study the core OSPF topics a bit more. It could be a good idea to circle back and redo the OSPF section of VOL1 to refresh your memory and hands-on skills, since OSPF is a core topic in the CCIE R&S exam.

R3:

```
!  
! Ensure R3 is NEVER elected as DR and tune Hello timers to minimum  
!  
interface Serial1/1  
  ip ospf priority 0  
  ip ospf dead-interval minimal hello-multiplier 3
```

R4:

```
!  
! Ensure R4 is never the DR on the Frame-Relay cloud  
!  
interface Serial0/0/0  
  ip ospf priority 0  
!  
! Set the OSPF network to non-broadcast and configure a neighbor  
! This will prevent other nodes from intercepting OSPF broadcasts  
!  
interface FastEthernet0/0  
  ip ospf network non-broadcast  
!  
router ospf 1  
  neighbor 183.1.45.5
```

```
!  
! Configure OSPF costs so that the Ethernet link b/w R4 and R5  
! is used for backup. Virtual link is needed as the area used  
! over the backup link is non-backbone.  
!  
interface FastEthernet0/0  
  ip ospf cost 10000  
router ospf 1  
  area 45 virtual-link 150.1.5.5  
!  
! Tune the dead/hello timers for 1-second neighbor loss detection  
!  
interface Serial0/0/0  
  ip ospf dead-interval minimal hello-multiplier 3  
  
R5:  
!  
! R5 will use the default priority on the FR cloud and become a DR  
!  
!  
! Set the OSPF network to non-broadcast and configure a neighbor  
! This will prevent other nodes from intercepting OSPF broadcasts  
!  
interface FastEthernet0/1  
  ip ospf network non-broadcast  
!  
router ospf 1  
  neighbor 183.1.45.4  
  
!  
! Configure OSPF costs so that the Ethernet link b/w R4 and R5  
! is used for backup. Virtual link is needed as the area used  
! over the backup link is non-backbone.  
!  
interface FastEthernet0/1  
  ip ospf cost 10000  
!  
router ospf 1  
  area 45 virtual-link 150.1.4.4  
!  
! Tune the dead/hello timers for 1-second neighbor loss detection  
!  
interface Serial0/0/0  
  ip ospf dead-interval minimal hello-multiplier 3  
  
R6:  
!  
! Originate VLAN 6 into OSPF via redistribution  
!  
router ospf 1  
  redistribute connected route-map CONNECTED->OSPF subnets  
!  
ip prefix-list VLAN_6 permit 183.1.6.0/24  
!  
route-map CONNECTED->OSPF permit 10  
  match ip address prefix-list VLAN_6
```

Task 2.1 Breakdown

The main points to be taken from the scenario requirements are as follows:

- 1) R5 should always be elected as a DR on the shared segment. This implies other OSPF routers should have the priority value of zero, in order to never participate in the DR election.
- 2) Devices on VLAN 45 should not be able to intercept OSPF communications between R4 and R5. In this context this means that broadcast packets should not be used as those are flooded to all Ethernet ports. This results in non-broadcast OSPF network type and static neighbors configured on R4 and R5.
- 3) A prefix on R6 should be advertised into OSPF without the use of the two commonly used commands. This leaves us with redistribution as the only solution.
- 4) The Ethernet link between R4 and R5 should be used as a backup only. Since this link is not in Area 0 it will never be used as transit anyways. However, the other bullet requires this link to be usable in case of the Frame-Relay link on R4 or R5 going down. This means we need to run a virtual link across area 45 to allow for transit capability. And in effect, the OSPF costs for this link on R4 and R5 should be incremented to make sure it is not preferred over the other Frame Relay path.
- 5) Like mentioned previously, the fact that area 45 is non-backbone and may lose connection to Area 0 if the Frame Relay cloud fails, we need to run a virtual link across this area.
- 6) The sub-second failure detection requirement implies that we need to enable OSPF fast hellos on the cloud between R3, R4, and R5.

Deep Dive

- Why is it not sufficient to simply raise the OSPF router's priority to ensure it is elected as a DR?
- What are the drawbacks of OSPF non-deterministic DR election?
- What is the difference between an OSPF virtual link and a tunnel?
- Could you peer OSPF routers that are using broadcast and non-broadcast network types respectively?
- What are the large-scale limitations of using OSPF fast-hello timers? What could be an alternative?

Task 2.1 Verification

Verify the OSPF configuration – notice that R4 is adjacent with R3 and R5 over the Frame Relay interface.

Rack1R5#show ip ospf interface

```
Serial0/0/0 is up, line protocol is up
  Internet Address 183.1.0.5/24, Area 0
  Process ID 1, Router ID 150.1.5.5, Network Type BROADCAST, Cost: 64
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 150.1.5.5, Interface address 183.1.0.5
  No backup designated router on this network
<output omitted>
  Neighbor Count is 2, Adjacent neighbor count is 2
    Adjacent with neighbor 150.1.3.3
    Adjacent with neighbor 150.1.4.4

<output omitted>
```

```
Loopback0 is up, line protocol is up
  Internet Address 150.1.5.5/24, Area 0
  Process ID 1, Router ID 150.1.5.5, Network Type LOOPBACK, Cost: 1
  Loopback interface is treated as a stub Host
```

Rack1R5#show ip ospf neighbor

| Neighbor ID | Pri | State | Dead Time | Address | Interface |
|-------------|-----|--------------|-----------|------------|-----------------|
| 150.1.4.4 | 0 | FULL/ - | - | 183.1.45.4 | OSPF_VL0 |
| 150.1.3.3 | 0 | FULL/DROTHER | 736 msec | 183.1.0.3 | Serial0/0/0 |
| 150.1.4.4 | 0 | FULL/DROTHER | 952 msec | 183.1.0.4 | Serial0/0/0 |
| 150.1.4.4 | 1 | FULL/DR | 00:01:32 | 183.1.45.4 | FastEthernet0/1 |

Verify the OSPF network types on the segment between R4 and R5.

Rack1R4#show ip ospf interface FastEthernet 0/0

```
FastEthernet0/0 is up, line protocol is up
  Internet Address 183.1.45.4/24, Area 45
  Process ID 1, Router ID 150.1.4.4, Network Type NON_BROADCAST, Cost: 10
<output omitted>
```

Rack1R5#sh ip ospf interface FastEthernet 0/1

```
FastEthernet0/1 is up, line protocol is up
  Internet Address 183.1.45.5/24, Area 45
  Process ID 1, Router ID 150.1.5.5, Network Type NON_BROADCAST, Cost: 10
<output omitted>
```

Check that the VLAN 6 prefix is being listed as external:

```
Rack1R4#show ip route ospf
    183.1.0.0/24 is subnetted, 4 subnets
O E2   183.1.6.0 [110/20] via 183.1.46.6, 00:00:10, FastEthernet0/1
<output omitted>
```

Verify that OSPF uses unicast on the Ethernet links between R4 and R5:

```
Rack1R5#debug condition interface fastEthernet 0/1
Rack1R5#debug ip ospf hello
```

```
OSPF: Send hello to 183.1.45.4 area 45 on FastEthernet0/1 from
183.1.45.5
```

```
Rack1R5#undebug all
```

Verify the OSPF virtual link:

```
Rack1R4#show ip ospf virtual-links
Virtual Link OSPF_VL0 to router 150.1.5.5 is up
<output omitted>
    Transit area 45, via interface FastEthernet0/0, Cost of using 10000
<output omitted>
```

Check the OSPF routes:

```
Rack1R4#show ip route ospf
<output omitted>
O       150.1.5.5/32 [110/65] via 183.1.0.5, 00:00:21, Serial0/0/0
O       150.1.3.3/32 [110/65] via 183.1.0.3, 00:00:21, Serial0/0/0
```

Verify the backup configuration:

```
Rack1R4(config)#interface serial 0/0/0
Rack1R4(config-if)#shutdown
Rack1R4(config-if)#do show ip route ospf
<output omitted>
O       183.1.0.0 [110/10064] via 183.1.45.5, 00:00:23, FastEthernet0/0
<output omitted>
O       150.1.5.5/32 [110/10001] via 183.1.45.5, 00:00:23,
FastEthernet0/0
O       150.1.3.3/32 [110/10065] via 183.1.45.5, 00:00:23,
FastEthernet0/0
Rack1R4(config-if)#no shutdown
```

Verify the OSPF Fast Timers on R3, R4 and R5:

```
Rack1R5#show ip ospf interface S0/0/0 | include Timer
```

```
Timer intervals configured, Hello 333 msec, Dead 1, Wait 1,  
Retransmit 5
```

```
Rack1R4#show ip ospf interface S0/0/0 | include Timer
```

```
Timer intervals configured, Hello 333 msec, Dead 1, Wait 1,  
Retransmit 5
```

```
Rack1R3#show ip ospf interface S1/1 | include Timer
```

```
Timer intervals configured, Hello 333 msec, Dead 1, Wait 1,  
Retransmit 5
```

Task 2.2 Solution

Before You Begin

This section involves routing redistribution. Routing redistribution can be confusing at first. We recommend you reading the INE blog post named "Understanding Redistribution". It is located here:

<http://blog.ine.com/2008/02/09/understanding-redistribution-part-i/>

This blog post is intended to solidify your fundamental understanding of the redistribution process. When working with the scenario, take route redistribution step-by-step and verify each step as you go. For example, redistribute between EIGRP and RIP on SW4. Verify the redistribution by having SW2 ping BB2. If redistribution is not working properly, SW2 would not be able to ping BB2.

R3:

```
!  
! Redistribute the prefixes into EIGRP. Metric choice is arbitrary  
!  
router eigrp 100  
  redistribute connected metric 10000 100 255 1 1500 route-map  
CONNECTED->EIGRP  
!  
route-map CONNECTED->EIGRP permit 10  
  match interface FastEthernet0/0 FastEthernet0/1  
  
!  
! Redistribute between OSPF and EIGRP  
!  
router eigrp 100  
  redistribute ospf 1 metric 10000 100 255 1 1500  
!  
router ospf 1  
  redistribute eigrp 100 subnets  
!  
! See breakdown for more explanations  
!  
route-map CONNECTED->EIGRP permit 20  
  match interface Serial1/1 Loopback0
```

R5:

```
router eigrp 100
 redistribute ospf 1 metric 10000 100 255 1 1500
!
router ospf 1
 redistribute eigrp 100 subnets
```

R6:

```
!
! Configure EIGRP authentication
!
key chain EIGRP
 key 1
  key-string CISCO
!
interface Serial0/0/0
 ip authentication mode eigrp 10 md5
 ip authentication key-chain eigrp 10 EIGRP

!
! Set up redistribution
!
router eigrp 10
 redistribute ospf 1 metric 10000 100 255 1 1500
!
router ospf 1
 redistribute eigrp 10 subnets

! See breakdown for more explanations
!
route-map CONNECTED->OSPF permit 20
 match interface Serial0/0/0
```

```
SW4:
!
! Configure EIGRP authentication
!
key chain RIP
  key 1
    key-string CISCO
!
interface Vlan102
  ip rip authentication mode md5
  ip rip authentication key-chain RIP

!
! Configure mutual redistribution
!
router eigrp 100
  redistribute rip metric 10000 100 255 1 1500
!
router rip
  redistribute eigrp 100 metric 1
```

Task 2.2 Breakdown

The prefix advertisement and authentication part of this task are basic. The only complex task here is route redistribution. First, the redistribution between EIGRP and RIP on SW4 is straightforward and does not create any potential loops, as there is only one connection between the two domains. The same goes for the redistribution between EIGRP and OSPF on R6 – there is no potential for routing loops. There are some other problems here, however.

One of these problems is located on R6, and involves the redistribution of EIGRP into OSPF. In a previous OSPF section on R6, VLAN 6 was advertised into OSPF through redistribution. When this redistribution was configured a route-map was used to limit redistribution to only the VLAN 6 interface. However, when EIGRP is then redistributed into OSPF on R6, connected interfaces running EIGRP will **not** be redistributed into OSPF. This is due to the fact that the route-map *CONNECTED->OSPF* ends in an implicit “deny”. Therefore, either the route-map should be removed from the configuration, or it should be modified to allow the connected Serial interface to be redistributed into OSPF.

The same problem occurs on R3 when redistributing into EIGRP. Since connected redistribution is already occurring with a route-map filter, the Serial1/1 Frame Relay link in the OSPF domain will not be redistributed into EIGRP. This is because the link is treated as a connected interface first before being treated as an OSPF interface. To solve this, like on R6, either the connected to EIGRP

route-map should be removed on R3, or it should be modified to include the Serial1/1 link.

Let us see if there is potential for the routing loops here. Mutual redistribution is performed on R3 and R5, which creates two points of redistribution between the two domains. However, the external EIGRP prefixes distance is automatically set to 170, which should prevent loops from appearing in the topology. Additionally, even if RIP prefixes learned from BB2 would loop back to SW4 across the EIGRP and OSPF domains, EIGRP's external distance of 170 will not allow them to preempt the native prefixes learned from BB2. In reality however, the same prefixes are learned by SW4 via BGP and preempt RIP prefixes. This is not considered an issue in this scenario, as the same prefixes become reachable via BGP. However you should notice that you will not see any RIP prefixes redistributed into EIGRP.

The same idea applies to EIGRP prefixes learned from BB1, as EIGRP's native distance is better than OSPF's external distance of 110. If you are not sure about your solution, use the command `debug ip routing` on all routers and check if there are any instabilities.

Deep Dive

- What is the main difference with the redistribute function between IPv6 and IPv4 routing protocols?
- What is the best practice for setting AD in IGP routing protocols?
- How could you fix the lack of external AD in RIP?
- Why are routing loops possible in scenarios with redistribution?

Task 2.2 Verification

Check that the networks appear as EIGRP external routes:

```
Rack1R1#show ip route eigrp | include D EX
D EX 204.12.1.0/24 [170/2707456] via 183.1.123.2, 00:00:51, Serial0/0/0
D EX 183.1.39.0 [170/2707456] via 183.1.123.2, 00:02:20, Serial0/0/0
```

Check that we have BB1 as an EIGRP neighbor with authentication enabled:

```
Rack1R6#show ip eigrp neighbors
IP-EIGRP neighbors for process 100
H Address Interface Hold Uptime SRTT RTO Q Seq Type
0 54.1.1.254 Se0/0/0 13 00:01:38 70 420 0 91
```

See if we actually receive authenticated packets:

```
Rack1R6#debug eigrp packets hello
<output omitted>
EIGRP: received packet with MD5 authentication, key id = 1
EIGRP: Received HELLO on Serial0/0/0 nbr 54.1.1.254
AS 10, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
```

Check if we have RIP enabled and the key-chain attached:

```
Rack1SW4#show ip protocols | begin rip
Routing Protocol is "rip"
  Sending updates every 30 seconds, next due in 14 seconds
  Invalid after 180 seconds, hold down 180, flushed after 240
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Redistributing: rip
  Default version control: send version 2, receive version 2
    Interface Send Recv Triggered RIP Key-chain
    Vlan102 2 2 RIP
  Automatic network summarization is not in effect
  Maximum path: 4
  Routing for Networks:
    192.10.1.0
  Routing Information Sources:
    Gateway Distance Last Update
    192.10.1.254 120 00:00:03
  Distance: (default is 120)
```

Verify full connectivity with the following TCL script:

```
tclsh
proc ping-internal {} {
  foreach i {
    150.1.1.1
    150.1.2.2
    150.1.3.3
    150.1.4.4
    150.1.5.5
    150.1.6.6
    150.1.7.7
    150.1.8.8
    150.1.10.10
    183.1.0.3
    183.1.0.4
    183.1.0.5
    183.1.123.1
    183.1.123.2
    183.1.123.3
    183.1.17.1
    183.1.17.7
    183.1.28.2
    183.1.28.8
    183.1.45.4
    183.1.45.5
    183.1.46.4
    183.1.46.6
    183.1.105.5
    183.1.105.10
    183.1.6.6
    183.1.107.7
    183.1.107.10
    192.10.1.10
    204.12.1.3
    54.1.1.6
  } { puts [ exec "ping $i" ] }
}
```

© Strategy Tip

By using procedures within TCL, it allows you to re-run your ping test without having to paste the foreach loop back into the router. The procedure can be called at any time by just typing the procedure's name on the command line. Additionally, if you need to build the list of IP addresses quickly, use the command **show ip alias** on all routers to quickly list the available addresses. After this, use the block selection feature (holding Alt while selecting a region with your mouse) to copy just the IP addresses in your notepad.

Use the following script, to check backbone IGP connectivity:

```
proc ping-external {} {  
    foreach i {  
        200.0.0.1  
        200.0.1.1  
        200.0.2.1  
        200.0.3.1  
        222.22.2.1  
        220.20.3.1  
        205.90.31.1  
    } { puts [ exec "ping $i" ] }  
}
```

Rack1R1#**tclsh**

```
Rack1R1(tcl)#proc ping-internal {} {  
+> foreach i {  
+> 150.1.1.1  
+> 150.1.2.2  
+> 150.1.3.3  
+> 150.1.4.4  
+> 150.1.5.5  
+> 150.1.6.6  
+> 150.1.7.7  
+> 150.1.8.8  
+> 150.1.10.10  
+> 183.1.0.3  
+> 183.1.0.4  
+> 183.1.0.5  
+> 183.1.123.1  
+> 183.1.123.2  
+> 183.1.123.3  
+> 183.1.17.1  
+> 183.1.17.7  
+> 183.1.28.2  
+> 183.1.28.8  
+> 183.1.39.3  
+> 183.1.39.9  
+> 183.1.45.4  
+> 183.1.45.5  
+> 183.1.46.4  
+> 183.1.46.6  
+> 183.1.105.5  
+> 183.1.105.10  
+> 183.1.6.6  
+> 183.1.107.7  
+> 183.1.107.10  
+> 192.10.1.10  
+> 204.12.1.3  
+> 54.1.1.6  
+> } { puts [ exec "ping $i" ] }  
+>}
```

```
Rack1R1(tcl)#ping-internal
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 150.1.1.1, timeout is 2 seconds:
```

```
!!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```

```
<output omitted>
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 54.1.1.6, timeout is 2 seconds:
```

```
!!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 140/143/145 ms
```

```
Rack1R1(tcl)#ping-external
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 200.0.0.1, timeout is 2 seconds:
```

```
!!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 204/205/208 ms
```

```
<output omitted>
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 205.90.31.1, timeout is 2 seconds:
```

```
!!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 204/205/212 ms
```

```
Rack1R1(tcl)#tclquit
```

```
Rack1R1#
```

☠ Pitfall

Remember to exit the TCL shell using the `tclquit` command when finished with the reachability verification. If the TCL shell is enabled, commands that overlap between TCL and the IOS will be interpreted by TCL and not the IOS. An example is the `set` command used in a route-map. Both TCL and the IOS use the `set` command. If you try to use the `set` command in a route-map when the TCL shell is still enabled the TCL shell will display an error message:

```
Rack1R1(tcl)#conf t
Rack1R1(config)#route-map TEST
Rack1R1(config-route-map)#set ip next-hop 1.1.1.1
wrong # args: should be "set varName ?newValue?"
Rack1R1(config-route-map)#do tclquit
Rack1R1(config-route-map)#set ip next-hop 1.1.1.1
Rack1R1(config-route-map)#
```

 **Note**

Older Catalyst IOS versions do not support TCL scripting. A smartport macro can be used in place of the TCL shell for ping tests on the switches as follows.

```
Rack1SW3(config)#macro name PINGS
```

```
Enter macro commands one per line. End with the character '@'.
```

```
do ping 150.1.1.1
```

```
do ping 150.1.2.2
```

```
<output omitted>
```

```
@
```

```
Rack1SW3(config)#macro global apply PINGS
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 150.1.1.1, timeout is 2 seconds:
```

```
!!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/113/116 ms
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 150.1.2.2, timeout is 2 seconds:
```

```
!!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/60 ms
```

```
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 150.1.3.3, timeout is 2 seconds:
```

```
!!!!!
```

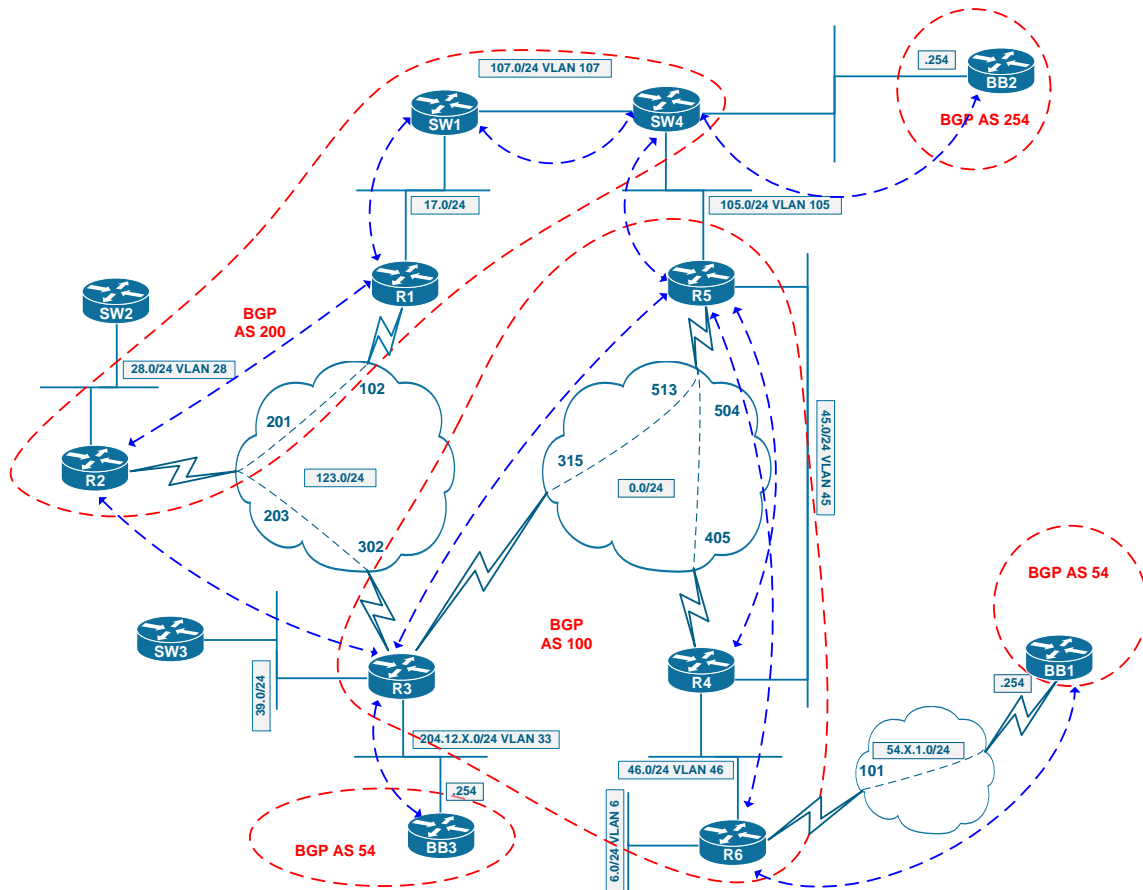
```
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/60 ms
```

```
<output omitted>
```

Task 2.3 Solution

 **Note**

The document does not say anything about BGP peering sessions, so you have to figure out those on your own and draw a separate BGP diagram. Use the commands `show ip bgp summary` to discover the BGP peering links and find out about any peering issues. The diagram below displays the BGP peering sessions for this scenario. You may quickly figure out that R5 is the RR for AS 100, and R1/SW1 are the RRs for AS 200.



 **Before You Begin**

The solution uses BGP ingress and egress path manipulations by means of the MED and LOCAL_PREFERENCE attributes. For scenarios illustrating the use of these features, you could look into VOL1's scenarios **BGP > BGP Bestpath Selection using Local Preference** and **BGP > BGP Bestpath Selection using MED**. We suggest doing all the BGP best-path selection scenarios to become familiar with all possible BGP path manipulations however, not just these two.

R1:

```
!  
! Advertise the Loopback prefix  
!  
interface Loopback1  
  ip address 150.1.11.1 255.255.255.0  
!  
router bgp 200  
  network 150.1.11.0 mask 255.255.255.0
```

R2:

```
ip prefix-list R1_BGP_LOOPBACK seq 5 permit 150.1.11.0/24  
!  
! Configure R2 to set higher metric to R1's new loopback,  
! when advertising it to R3. This will make R3 prefer the exit via R5  
!  
route-map MED permit 10  
  match ip address prefix-list R1_BGP_LOOPBACK  
  set metric 200  
!  
route-map MED permit 1000  
!  
router bgp 200  
  neighbor 183.1.123.3 route-map MED out
```

R5:

```
!  
! Changing next-hop is required as R3 prefers to reach the  
! VLAN 105 link across the EIGRP domain, i.e. another AS  
!  
router bgp 100  
  neighbor 183.1.0.3 next-hop-self
```

 **Note**

We change the BGP next-hop on R5 to “self” when sending prefixes to R3. Otherwise, R3 would recursively resolve the next hop to 183.1.105.10 which is reachable via EIGRP and not OSPF (lower administrative distance). Based on that, the packets would actually be forwarded out R3’s serial connection with R2.

R6:

```
!  
! AS_PATH access-list to match prefixes originated in AS 54  
!  
ip as-path access-list 1 permit _54$  
!  
! Increase local preference for the prefixes originated in AS 54  
!  
route-map LOCAL_PREFERENCE permit 10  
  match as-path 1  
  set local-preference 200  
!  
route-map LOCAL_PREFERENCE permit 1000  
!  
router bgp 100  
  neighbor 54.1.1.254 route-map LOCAL_PREFERENCE in
```

SW4:

```
ip prefix-list R1_BGP_LOOPBACK seq 5 permit 150.1.11.0/24  
!  
! SW4 sets R1's new loopback metric to 100, which is better  
! than R2's setting of 200. This will make AS 100 prefer to go  
! via R5 to reach R1's new loopback  
!  
route-map MED permit 10  
  match ip address prefix-list R1_BGP_LOOPBACK  
  set metric 100  
!  
route-map MED permit 1000  
!  
router bgp 200  
  neighbor 183.1.105.5 route-map MED out
```

Task 2.3 Breakdown

There are two main goals in this task:

- 1) Make AS 100 prefer BB1 for reaching AS 54 originated prefixes while preserving the second link for backup – this is egress path manipulation.
- 2) Configure AS 200 to signal AS 100 to use R5 to reach the new Loopback address of R1 – this is ingress path manipulation.

The first goal is achieved through ingress prefix manipulation and adjusting Local Preference. One special feature used here is the AS_PATH access-list matching just the prefixes ORIGINATED in AS 54 utilizing the regexp “_54\$”. The second goal is achieved by assigning different MED values to the newly advertised R1’s Loopback prefix. This is because the task explicitly prohibits the use of AS_PATH manipulation.

Further Learning

Let us provide a quick refresh for the BGP regular expressions, as those are used in this lab. First, recall the basic regular expression meta-characters or modifiers: “.” – any character, “?” – repeat the previous character one or zero times, “*” – repeat the previous character zero or any times, “+” – repeat the previous character one or more times, “^” – match the beginning of a string, “\$” match the end of a string, “[]” means range or elements and “_” is the character to match the “space” separating AS numbers OR the end of the AS_PATH list.

Other important regexp features include **grouping** and **back-referencing**. You can use parentheses to group AS numbers, e.g. "(123 124 1+)" and every group is assigned a number starting from left to right. For example in the string "1 2 (3 4) 5 6 (7 8)" the first group is assigned number "1" and the second group number "2". You can later "recall" the grouping using the commands \1, \2 and so on for the group numbers. For example the string "(1 2) 3 \1" would match "1 2 3 1 2". You may use the pipe character "|" in addition to the grouping characters for the concept of alternation. For example (1 2)|(5 6) would match "1 2" or "5 6".

Now some practical examples:

"^\$" – means an empty AS_PATH attribute, which identifies the prefixes advertised in the local AS.

"^254_" - means prefixes received from the directly adjacent AS 254. Notice that using "_" is important, as there could be another adjacent AS with the number starting with 254.

"_254_" - prefixes transiting AS 254. The "_" characters are needed to clearly separate the AS number.

"_254\$" - means prefixes originated in AS 254. This expression matches the rightmost position in the string, meaning that the expression could be of arbitrary length.

"^[0-9+]_254" - routes from AS 254 when it's just "one-hop" away.

"^254_[0-9+]" - prefixes from the clients of the directly connected AS 254.

"^(254_)+([0-9]+)" - prefixes from the clients of the adjacent AS 254, accounting for the fact that AS 254 may do AS_PATH prepending.

"^254_([0-9+]_)+)" - prefixes from the clients of the adjacent AS 254, accounting for the fact that the clients may do AS_PATH prepending.

"^(65100)" – prefixes learned from the confederation peer 65100.

You configure BGP regular-expressions using the AS-PATH access-lists syntax: `ip as-path access-list <N> {permit|deny} <Regex>`. This access-list might be applied as a filter-list to a peer using the syntax: `neighbor <IP> filter-list <N> {in|out}`. However, the best approach is to match AS_PATH access-lists under a route-map applied to the peer (`match as-path`), as this allows for flexible policy editing. If you are wondering about the order features are applied, it is as follows:

For inbound updates:

1. route-map
2. filter-list
3. prefix-list OR distribute-list

For outbound updates:

1. prefix-list OR distribute-list
2. filter-list
3. route-map

Keep in mind that you may test regular expression on the BGP table using the command `show ip bgp regexp` or `show ip bgp quote-regexp`. The latter command allows using the “|” character to additionally filter the output.

Task 2.3 Verification

Verify that local preference is correctly set to 200 for routes originating from AS 54. Do not forget to issue the command `clear ip bgp 54.1.1.254 soft in` to ensure the new policy enforcement.:

```
Rack1R6#clear ip bgp 54.1.1.254 soft in
Rack1R6#show ip bgp regexp _54$
<output omitted>
*> 28.119.16.0/24    54.1.1.254          200      0 54 i
*> 28.119.17.0/24    54.1.1.254          200      0 54 i
*> 114.0.0.0        54.1.1.254          0        200     0 54 i
*> 115.0.0.0        54.1.1.254          0        200     0 54 i
*> 116.0.0.0        54.1.1.254          0        200     0 54 i
*> 117.0.0.0        54.1.1.254          0        200     0 54 i
*> 118.0.0.0        54.1.1.254          0        200     0 54 i
*> 119.0.0.0        54.1.1.254          0        200     0 54 i
```

And that all other paths have the local preference of 100:

```
Rack1R6#show ip bgp regexp ^200
```

```
<output omitted>
```

```
*>i150.1.11.0/24    183.1.105.10      100    100    0 200 i
*>i205.90.31.0     183.1.105.10      0       100    0 200 254 ?
*>i220.20.3.0      183.1.105.10      0       100    0 200 254 ?
*>i222.22.2.0      183.1.105.10      0       100    0 200 254 ?
```

```
Rack1R6#show ip bgp regexp ^54.
```

```
<output omitted>
```

```
*> 112.0.0.0        54.1.1.254        0                0 54 50 60 i
*> 113.0.0.0        54.1.1.254        0                0 54 50 60 i
```

```
Rack1R6#show ip bgp 112.0.0.0
```

```
BGP routing table entry for 112.0.0.0/8, version 22
```

```
Paths: (1 available, best #1, table Default-IP-Routing-Table)
```

```
  Advertised to update-groups:
```

```
    1
    54 50 60
    54.1.1.254 from 54.1.1.254 (212.18.3.1)
      Origin IGP, metric 0, localpref 100, valid, external, best
```

```
Rack1R6#show ip bgp 113.0.0.0
```

```
BGP routing table entry for 113.0.0.0/8, version 21
```

```
Paths: (1 available, best #1, table Default-IP-Routing-Table)
```

```
  Advertised to update-groups:
```

```
    1
    54 50 60
    54.1.1.254 from 54.1.1.254 (212.18.3.1)
      Origin IGP, metric 0, localpref 100, valid, external, best
```

Confirm that R3, R4, and R5 take the exit towards R6 for AS 54 originated networks:

```
Rack1R5#show ip bgp regexp _54$
```

```
<output omitted>
```

| Network | Next Hop | Metric | LocPrf | Weight | Path |
|-------------------|------------|--------|--------|--------|------|
| *>i28.119.16.0/24 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| *>i28.119.17.0/24 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| *>i114.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| *>i115.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| *>i116.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| *>i117.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| *>i118.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| *>i119.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |

```
Rack1R4#show ip bgp regexp _54$
```

```
<output omitted>
```

| Network | Next Hop | Metric | LocPrf | Weight | Path |
|-------------------|------------|--------|--------|--------|------|
| *>i28.119.16.0/24 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| *>i28.119.17.0/24 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| *>i114.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| *>i115.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| *>i116.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| *>i117.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| *>i118.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |

```
Rack1R3#show ip bgp rege _54$
```

```
<output omitted>
```

| Network | Next Hop | Metric | LocPrf | Weight | Path |
|-------------------|--------------|--------|--------|--------|------|
| *>i28.119.16.0/24 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| * | 204.12.1.254 | 0 | | 0 | 54 i |
| *>i28.119.17.0/24 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| * | 204.12.1.254 | 0 | | 0 | 54 i |
| *>i114.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| * | 204.12.1.254 | | | 0 | 54 i |
| *>i115.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| * | 204.12.1.254 | | | 0 | 54 i |
| *>i116.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| * | 204.12.1.254 | | | 0 | 54 i |
| *>i117.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| * | 204.12.1.254 | | | 0 | 54 i |
| *>i118.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| * | 204.12.1.254 | | | 0 | 54 i |
| *>i119.0.0.0 | 54.1.1.254 | 0 | 200 | 0 | 54 i |
| * | 204.12.1.254 | | | 0 | 54 i |

Ensure that R3 has two paths to R1's Loopback, but selects only one as the best path based on the MED value.

```
Rack1R3# show ip bgp 150.1.11.0
```

```
BGP routing table entry for 150.1.11.0/24, version 20
```

```
Paths: (2 available, best #1, table Default-IP-Routing-Table)
```

```
Advertised to update-groups:
```

```
1
```

```
200
```

```
183.1.0.5 from 183.1.0.5 (150.1.5.5)
```

```
Origin IGP, metric 100, localpref 100, valid, internal, best
```

```
200
```

```
183.1.123.1 from 183.1.123.2 (150.1.2.2)
```

```
Origin IGP, metric 200, localpref 100, valid, external
```

```
Rack1R5#show ip bgp 150.1.11.0
```

```
BGP routing table entry for 150.1.11.0/24, version 32
```

```
Paths: (1 available, best #1, table Default-IP-Routing-Table)
```

```
Advertised to update-groups:
```

```
2 3
```

```
200
```

```
183.1.105.10 from 183.1.105.10 (150.1.10.10)
```

```
Origin IGP, metric 100, localpref 100, valid, external, best
```

Verify that we can shut down R5's link to SW4 and R5 will use the path via R1:

```
Rack1R3#traceroute 150.1.11.1
```

```
Type escape sequence to abort.
```

```
Tracing the route to 150.1.11.1
```

```
1 183.1.0.5 28 msec 28 msec 56 msec
```

```
2 183.1.105.10 28 msec 32 msec 48 msec
```

```
3 183.1.107.7 28 msec 28 msec 28 msec
```

```
4 183.1.17.1 32 msec * 28 msec
```

```
Rack1R5#conf t
```

```
Enter configuration commands, one per line. End with CNTL/Z.
```

```
Rack1R5(config)#interface FastEthernet 0/0
```

```
Rack1R5(config-if)#shutdown
```

```
Rack1R5#show ip bgp 150.1.11.0
```

```
BGP routing table entry for 150.1.11.0/24, version 29
```

```
Paths: (1 available, best #1, table Default-IP-Routing-Table)
```

```
Advertised to update-groups:
```

```
3
```

```
200, (Received from a RR-client)
```

```
183.1.123.1 (metric 20) from 183.1.0.3 (150.1.3.3)
```

```
Origin IGP, metric 200, localpref 100, valid, internal, best
```

```
Rack1R5#traceroute 150.1.11.1
```

```
Type escape sequence to abort.
```

```
Tracing the route to 150.1.11.1
```

```
1 183.1.0.3 28 msec 32 msec 32 msec
```

```
2 183.1.123.2 44 msec 48 msec 44 msec
```

```
3 183.1.123.1 52 msec * 48 msec
```

```
Rack1R5(config-if)#no shutdown
```

 **Deep Dive**

- What is the purpose of the MED attribute in BGP?
- How is “cold potato” routing different from “hot potato” routing in BGP?
- Why would comparing the MED between multiple Autonomous Systems not work?

Task 3.1 Solution

Before You Begin

For this particular scenario, you may want to check **VOL1's IPv6** scenario named **EIGRPv6** as well as **IPv6 Tunneling**. These sections directly correspond to the features used in the solution.

R4:

```
!  
! Enable IPv6 routing and configure IPv6 addressing  
!  
ipv6 unicast-routing  
!  
interface FastEthernet0/1  
  ipv6 address 2001:CC1E:1:404::/64 eui-64  
!  
! Create the IPv6 in IPv4 tunnel between R4 and R5  
!  
interface Tunnel45  
  ipv6 address 2001:CC1E:1:4545::4/64  
  tunnel source 150.1.4.4  
  tunnel destination 150.1.5.5  
  tunnel mode ipv6ip  
  
!  
! Configure EIGRPv6 for IPv6 routing  
!  
ipv6 router eigrp 45  
  no shutdown  
!  
interface Tunnel45  
  ipv6 eigrp 45  
!  
interface FastEthernet0/1  
  ipv6 eigrp 45
```

R5:

```
!  
! Enable IPv6 routing and configure IPv6 addressing  
!  
ipv6 unicast-routing  
!  
interface FastEthernet0/0  
  ipv6 address 2001:CC1E:1:505::/64 eui-64
```

```
!  
! Create the IPv6 in IPv4 tunnel between R4 and R5  
!  
interface Tunnel45  
  ipv6 address 2001:CC1E:1:4545::5/64  
  tunnel source 150.1.5.5  
  tunnel destination 150.1.4.4  
  tunnel mode ipv6ip  
!  
! Configure EIGRPv6 for IPv6 routing  
!  
ipv6 router eigrp 45  
  no shutdown  
!  
interface Tunnel45  
  ipv6 eigrp 45  
!  
interface FastEthernet0/0  
  ipv6 eigrp 45
```

Task 3.1 Breakdown

The task requirements are straightforward, directing you to use the EUI-64 host-porting for IPv6 addressing and configure a point-to-point IPv6 tunnel. EIGRPv6 is used as the routing protocol for this task.

Further Learning

If you do not know much about IPv6, then working through the relatively compact IPv6 section of **VOL1** could be a good idea. Simply omit complex topics such as IPv6 multicast and look into IPv6 tunnels, IGP routing (RIPng, EIGRPv6 and OSPFv3) and addressing. This constitutes the major part of the IPv6 configuration, and you can always circle back to the advanced IPv6 topics such as multicasting when you are solid with the fundamentals.

Task 3.1 Verification

Verify IPv6 addressing:

```
Rack1R5#show ipv6 interface brief
FastEthernet0/0 [up/up]
FE80::207:EBFF:FEDE:5621
2001:CC1E:1:505:207:EBFF:FEDE:5621
```

```
Rack1R4#show ipv6 interface brief
FastEthernet0/1 [up/up]
FE80::230:94FF:FE7E:E582
2001:CC1E:1:404:250:80FF:FE04:8E01
```

```
Rack1R5#show ipv6 interface brief fastEthernet 0/0
FastEthernet0/0 [up/up]
FE80::C004:CFF:FE7A:0
2001:CC1E:1:505:C004:CFF:FE7A:0
```

```
Rack1R4#show ipv6 interface brief fastEthernet 0/1
FastEthernet0/1 [up/up]
FE80::C003:CFF:FE7A:1
2001:CC1E:1:404:C003:CFF:FE7A:1
```

Verify the tunnel:

```
Rack1R5#show interfaces tunnel 45
Tunnel45 is up, line protocol is up
<output omitted>
Tunnel source 150.1.5.5, destination 150.1.4.4
Tunnel protocol/transport IPv6/IP
```

```
Rack1R5#ping 2001:CC1E:1:404:C003:CFF:FE7A:1
```

```
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:404:C003:CFF:FE7A:1,
timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/71/76 ms
```

Check to see that EIGRPv6 is configured correctly:

Rack1R5#show ipv6 eigrp neighbors

```
IPv6-EIGRP neighbors for process 45
H Address Interface Hold Uptime SRTT RTO Q Seq
(sec) (ms) Cnt Num
0 Link-local address: Tu45 11 00:02:55 122 5000 0 3
FE80::9601:404
```

Rack1R4#show ipv6 eigrp neighbors

```
IPv6-EIGRP neighbors for process 45
H Address Interface Hold Uptime SRTT RTO Q Seq
(sec) (ms) Cnt Num
0 Link-local address: Tu45 12 00:05:48 45 5000 0 4
FE80::9601:505
```

Verify EIGRPv6 learned prefixes.

Rack1R4#show ipv6 route eigrp

IPv6 Routing Table - 6 entries

Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP

U - Per-user Static route, M - MIPv6

I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary

O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2

ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2

D - EIGRP, EX - EIGRP external

D 2001:CC1E:1:505::/64 [90/297246976]

via FE80::9601:505, Tunnel45

Rack1R5#show ipv6 route eigrp

IPv6 Routing Table - 6 entries

Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP

U - Per-user Static route, M - MIPv6

I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary

O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2

ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2

D - EIGRP, EX - EIGRP external

D 2001:CC1E:1:404::/64 [90/297246976]

via FE80::9601:404, Tunnel45

Task 3.2 Solutions

Before You Begin

This task requires configuring IPv6 multicasting. This is an advanced topic and you may even skip it the first time you do this lab. It is recommended to master IPv4 multicasting first so dealing with IPv6 mutlciast next would be easy due to many similarities between the technologies. Both protocols use the same concepts, just IPv6 multicasting may slightly different commands for verification. If you are not familiar with the topics at all, it is recommended to complete the following scenarions from VOL1: IPv6: IPv6 PIM and MLD and IPv6 PIM BSR.

R5:

```
!
! The below is the IPv6 EUI-64 address of R5's Fa 0/0 interface
! It will be different depending on the interface's MAC address
! So you need to adjust it to your environment.
!
ipv6 multicast-routing
!
ipv6 pim bsr candidate rp 2001:CC1E:1:505:21A:6CFF:FE6B:E2BE
```

R4:

```
!
! The below is the IPv6 EUI-64 address of R4's Fa0/1 interface
! It will be different depending on the interface MAC address
!
ipv6 multicast-routing
ipv6 pim bsr candidate bsr 2001:CC1E:1:404:219:56FF:FE22:8DD7 priority
100
!
! IPv6 access-list to be used with MLD filtering
!
ipv6 access-list MLD_FILTER
 permit ipv6 any FF06::6/16
!
! Apply the MLD filter
!
interface FastEthernet 0/1
 ipv6 mld access-group MLD_FILTER
```

R6:

```
!
! Configure R6 to join the group statically
!
interface FastEthernet 0/0
 ipv6 enable
 ipv6 mld join-group ff06::6
```

Task 3.2 Breakdown

The task instructs you to configure IPv6 multicast on the two IPv6 routers connected via a tunnel. There are two requirements – R5 should be the RP, and R4 should be the BSR. As the task states, a single router cannot be both. We configure the two devices using their Ethernet interfaces for PIM BSR and RP addresses. R6 is configured to statically join the given group and thus should have IPv6 enabled on its connection to R4. The final requirement is with respect to MLD access filtering – R4 has to have an IPv6 access-list created to permit only the mentioned IPv6 multicast groups in MLD reports.



Deep Dive

- Why does IPv6 PIM use a special tunnel interface built for every RP?
- What version of IGMP could be thought of as equivalent to MLD?

Task 3.2 Verification

Check the RPs learned by the BSR, and the groups mapped to the RP on R4 and R5:

```
Rack1R4#show ipv6 pim bsr rp-cache
```

```
PIMv2 BSR C-RP Cache
```

```
BSR Candidate RP Cache
```

```
Group(s) FF00::/8, RP count 1
```

```
RP 2001:CC1E:1:505:21A:6CFF:FE6B:E2BE SM
```

```
Priority 192, Holdtime 150
```

```
Uptime: 00:01:29, expires: 00:02:00
```

```
Rack1R4#show ipv6 pim range-list
```

```
Static SSM Exp: never Learnt from : ::
```

```
FF33::/32 Up: 00:02:32
```

```
FF34::/32 Up: 00:02:32
```

```
FF35::/32 Up: 00:02:32
```

```
FF36::/32 Up: 00:02:32
```

```
FF37::/32 Up: 00:02:32
```

```
FF38::/32 Up: 00:02:32
```

```
FF39::/32 Up: 00:02:32
```

```
FF3A::/32 Up: 00:02:32
```

```
FF3B::/32 Up: 00:02:32
```

```
FF3C::/32 Up: 00:02:32
```

```
FF3D::/32 Up: 00:02:32
```

```
FF3E::/32 Up: 00:02:32
```

```
FF3F::/32 Up: 00:02:32
```

```
BSR SM RP: 2001:CC1E:1:505:21A:6CFF:FE6B:E2BE Exp: 00:01:58 Learnt from
```

```
: 2001:CC1E:1:404:219:56FF:FE22:8DD7
```

```
FF00::/8 Up: 00:01:31
```

Rack1R5#show ipv6 pim range-list

```
Static SSM Exp: never Learnt from : ::
  FF33::/32 Up: 00:05:41
  FF34::/32 Up: 00:05:41
  FF35::/32 Up: 00:05:41
  FF36::/32 Up: 00:05:41
  FF37::/32 Up: 00:05:41
  FF38::/32 Up: 00:05:41
  FF39::/32 Up: 00:05:41
  FF3A::/32 Up: 00:05:41
  FF3B::/32 Up: 00:05:41
  FF3C::/32 Up: 00:05:41
  FF3D::/32 Up: 00:05:41
  FF3E::/32 Up: 00:05:41
  FF3F::/32 Up: 00:05:41
BSR SM RP: 2001:CC1E:1:505:21A:6CFF:FE6B:E2BE Exp: 00:01:38 Learnt from
: 2001:CC1E:1:404:219:56FF:FE22:8DD7
  FF00::/8 Up: 00:02:51
```

Check the dynamic multicast routes in R4 and R5 after this:

Rack1R4#show ipv6 mroute

```
Rack1R4#show ipv6 mroute
Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group,
       C - Connected, L - Local, I - Received Source Specific Host
Report,
       P - Pruned, R - RP-bit set, F - Register flag, T - SPT-bit set,
       J - Join SPT
Timers: Uptime/Expires
Interface state: Interface, State

(*, FF06::6), 00:04:51/never, RP 2001:CC1E:1:505:21A:6CFF:FE6B:E2BE,
flags: SCJ
  Incoming interface: Tunnel45
  RPF nbr: FE80::961E:505
  Immediate Outgoing interface list:
    FastEthernet0/1, Forward, 00:04:51/never
```

R5 shows the incoming interface as "Tunnel2" as this is the dynamic virtual interface used to process incoming registration messages.

Rack1R5#show ipv6 mroute

```
Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group,
       C - Connected, L - Local, I - Received Source Specific Host
Report,
       P - Pruned, R - RP-bit set, F - Register flag, T - SPT-bit set,
       J - Join SPT
Timers: Uptime/Expires
Interface state: Interface, State
```

```
(* , FF06::6), 00:04:13/00:03:15, RP 2001:CC1E:1:505:21A:6CFF:FE6B:E2BE,
flags: S
  Incoming interface: Tunnel2
  RPF nbr: 2001:CC1E:1:505:21A:6CFF:FE6B:E2BE
  Immediate Outgoing interface list:
    Tunnel45, Forward, 00:04:13/00:03:15
```

Simulate multicast traffic and make sure it is being received:

```
Rack1R6#debug ipv6 icmp
```

```
ICMP Packet debugging is on
```

```
Rack1R5#ping ff06::6 repeat 100
```

```
Output Interface: Tunnel45
```

```
Type escape sequence to abort.
```

```
Sending 100, 100-byte ICMP Echos to FF06::6, timeout is 2 seconds:
```

```
Packet sent with a source address of 2001:CC1E:1:4545::5
```

```
Rack1R6#
```

```
ICMPv6: Sent N-Solicit, Src=FE80::21B:D4FF:FE70:3D8, Dst=FF02::1:FF00:5
```

```
ICMPv6: Received echo request, Src=2001:CC1E:1:4545::5, Dst=FF06::6
```

```
ICMPv6: Sent echo reply, Src=FE80::21B:D4FF:FE70:3D8,
```

```
Dst=2001:CC1E:1:4545::5
```

```
ICMPv6: Sent N-Solicit, Src=FE80::21B:D4FF:FE70:3D8, Dst=FF02::1:FF00:5
```

```
Rack1R6#
```

```
ICMPv6: Received echo request, Src=2001:CC1E:1:4545::5, Dst=FF06::6
```

```
ICMPv6: Sent echo reply, Src=FE80::21B:D4FF:FE70:3D8,
```

```
Dst=2001:CC1E:1:4545::5
```

```
ICMPv6: Sent N-Solicit, Src=FE80::21B:D4FF:FE70:3D8, Dst=FF02::1:FF00:5
```

Task 4.1 Solutions

Before You Begin

If you are not familiar with MPLS fundamentals, you may skip this section for the first time in order to revisit it later. However, if you feel like spending some time now on Teir 1 MPLS, check out the **MPLS LDP** and **MPLS Label Filtering** mini-scenarios from **VOL1**. These give you enough introduction within the scope of the R&S exam.

Next, when dealing with MPLS VPN scenarios, you may want to draw a small diagram in case if you have a complex VPN (more than two sites). In our case, these routers emulating the SP network are lined up in a simple manner and there is no PE-CE routing requirements, so you may skip this step.

R4:

```
!  
! Enable MPLS and configure label advertisement for the loopbacks only  
!  
mpls ip  
access-list 1 permit 150.1.0.0 0.0.255.255  
!  
no mpls ldp advertise-labels  
mpls ldp advertise-labels for 1  
!  
! Enable MPLS and LDP on the interfaces  
!  
interface FastEthernet 0/0  
  mpls ip  
!  
interface FastEthernet 0/1  
  mpls ip  
!  
interface Serial 0/0/0  
  mpls ip
```

R5:

```
!  
! Enable MPLS and configure label advertisement for the loopbacks only  
!  
mpls ip  
access-list 1 permit 150.1.0.0 0.0.255.255  
!  
no mpls ldp advertise-labels  
mpls ldp advertise-labels for 1  
!  
! Enable MPLS and LDP on the interfaces  
!  
interface FastEthernet 0/1  
    mpls ip  
!  
interface Serial 0/0/0  
    mpls ip
```

R6:

```
!  
! Enable MPLS and configure label advertisement for loopback only  
!  
mpls ip  
access-list 1 permit 150.1.0.0 0.0.255.255  
!  
no mpls ldp advertise-labels  
mpls ldp advertise-labels for 1  
  
!  
! Enable MPLS and LDP on the interface  
!  
interface FastEthernet 0/0  
    mpls ip
```

Task 4.1 Breakdown

The task hints to use the IETF standard LDP protocol and enable LDP on all interfaces connecting R4 and R5. Thus, if any of the links fail, the LDP session will remain active on the backup interface. The tasks point toward using the label announce filters by associating the access-list permitting only the Loopback subnets.

Deep Dive

- What label distribution modes do you know of?
- Why is a /32 prefix required for PE Loopback interfaces?
- How does LDP validate label advertisement against the local RIB?

Task 4.1 Verification

Check the MPLS LDP neighbors on R4, since it peers with both R5 and R6:

Rack1R4#show mpls ldp neighbor

```
Peer LDP Ident: 150.1.6.6:0; Local LDP Ident 150.1.4.4:0
  TCP connection: 150.1.6.6.39534 - 150.1.4.4.646
  State: Oper; Msgs sent/rcvd: 137/139; Downstream
  Up time: 01:49:44
  LDP discovery sources:
    FastEthernet0/1, Src IP addr: 183.1.46.6
  Addresses bound to peer LDP Ident:
    183.1.6.6      54.1.1.6      150.1.6.6      183.1.46.6
Peer LDP Ident: 150.1.5.5:0; Local LDP Ident 150.1.4.4:0
  TCP connection: 150.1.5.5.53030 - 150.1.4.4.646
  State: Oper; Msgs sent/rcvd: 138/138; Downstream
  Up time: 01:48:15
  LDP discovery sources:
    Serial0/0/0, Src IP addr: 183.1.0.5
    FastEthernet0/0, Src IP addr: 183.1.45.5
  Addresses bound to peer LDP Ident:
    150.1.5.5      183.1.105.5    183.1.45.5    183.1.0.5
    183.1.105.254
```

Now check that labels were generated for Loopback0 interfaces. Notice that all prefixes except the 150.1.0.0/16 range do not have labels assigned.

Rack1R4#show mpls forwarding-table | e 150.1

| Local Label | Outgoing Label or VC | Prefix or Tunnel Id | Bytes Label Switched | Outgoing interface | Next Hop |
|-------------|----------------------|---------------------|----------------------|--------------------|------------|
| 16 | No Label | 54.1.1.0/24 | 0 | Fa0/1 | 183.1.46.6 |
| 26 | No Label | 183.1.6.0/24 | 0 | Fa0/1 | 183.1.46.6 |
| 27 | No Label | 183.1.17.0/24 | 0 | Se0/0/0 | 183.1.0.3 |
| | No Label | 183.1.17.0/24 | 0 | Se0/0/0 | 183.1.0.5 |
| 28 | No Label | 183.1.28.0/24 | 0 | Se0/0/0 | 183.1.0.3 |
| | No Label | 183.1.28.0/24 | 0 | Se0/0/0 | 183.1.0.5 |
| 29 | No Label | 183.1.39.0/24 | 0 | Se0/0/0 | 183.1.0.5 |
| 30 | No Label | 183.1.105.0/24 | 0 | Se0/0/0 | 183.1.0.3 |
| | No Label | 183.1.105.0/24 | 0 | Se0/0/0 | 183.1.0.5 |
| 31 | No Label | 183.1.107.0/24 | 0 | Se0/0/0 | 183.1.0.3 |
| | No Label | 183.1.107.0/24 | 0 | Se0/0/0 | 183.1.0.5 |
| 32 | No Label | 183.1.123.0/24 | 0 | Se0/0/0 | 183.1.0.3 |
| | No Label | 183.1.123.0/24 | 0 | Se0/0/0 | 183.1.0.5 |
| 33 | No Label | 192.10.1.0/24 | 0 | Se0/0/0 | 183.1.0.3 |
| 34 | No Label | 200.0.0.0/24 | 0 | Fa0/1 | 183.1.46.6 |
| 35 | No Label | 200.0.1.0/24 | 0 | Fa0/1 | 183.1.46.6 |
| 36 | No Label | 200.0.2.0/24 | 0 | Fa0/1 | 183.1.46.6 |
| 37 | No Label | 200.0.3.0/24 | 0 | Fa0/1 | 183.1.46.6 |
| 38 | No Label | 204.12.1.0/24 | 0 | Se0/0/0 | 183.1.0.5 |

Rack1R4#show mpls forwarding-table | i 150.1

| | | | | | |
|----|-----------|---------------|------|---------|------------|
| 17 | 17 | 150.1.1.0/24 | 0 | Se0/0/0 | 183.1.0.5 |
| | No Label | 150.1.1.0/24 | 0 | Se0/0/0 | 183.1.0.3 |
| 18 | 18 | 150.1.2.0/24 | 0 | Se0/0/0 | 183.1.0.5 |
| | No Label | 150.1.2.0/24 | 0 | Se0/0/0 | 183.1.0.3 |
| 19 | 19 | 150.1.3.0/24 | 0 | Se0/0/0 | 183.1.0.5 |
| 20 | No Label | 150.1.3.3/32 | 0 | Se0/0/0 | 183.1.0.3 |
| 21 | Pop Label | 150.1.5.5/32 | 1391 | Se0/0/0 | 183.1.0.5 |
| 22 | Pop Label | 150.1.6.6/32 | 1651 | Fa0/1 | 183.1.46.6 |
| 23 | 23 | 150.1.7.0/24 | 0 | Se0/0/0 | 183.1.0.5 |
| | No Label | 150.1.7.0/24 | 0 | Se0/0/0 | 183.1.0.3 |
| 24 | 24 | 150.1.8.0/24 | 0 | Se0/0/0 | 183.1.0.5 |
| | No Label | 150.1.8.0/24 | 0 | Se0/0/0 | 183.1.0.3 |
| 25 | 25 | 150.1.10.0/24 | 0 | Se0/0/0 | 183.1.0.5 |
| | No Label | 150.1.10.0/24 | 0 | Se0/0/0 | 183.1.0.3 |

Rack1R5#show mpls forwarding-table | e 150.1

| Local Label | Outgoing Label or VC | Prefix or Tunnel Id | Bytes Switched | Label | Outgoing interface | Next Hop |
|-------------|----------------------|---------------------|----------------|-------|--------------------|--------------|
| 16 | No Label | 54.1.1.0/24 | 0 | | Se0/0/0 | 183.1.0.4 |
| 26 | No Label | 183.1.6.0/24 | 0 | | Se0/0/0 | 183.1.0.4 |
| 27 | No Label | 183.1.17.0/24 | 0 | | Fa0/0 | 183.1.105.10 |
| 28 | No Label | 183.1.28.0/24 | 0 | | Fa0/0 | 183.1.105.10 |
| 29 | No Label | 183.1.39.0/24 | 0 | | Fa0/0 | 183.1.105.10 |
| 30 | No Label | 183.1.46.0/24 | 0 | | Se0/0/0 | 183.1.0.4 |
| 31 | No Label | 183.1.107.0/24 | 0 | | Fa0/0 | 183.1.105.10 |
| 32 | No Label | 183.1.123.0/24 | 0 | | Fa0/0 | 183.1.105.10 |
| 33 | No Label | 192.10.1.0/24 | 0 | | Se0/0/0 | 183.1.0.3 |
| 34 | No Label | 200.0.0.0/24 | 0 | | Se0/0/0 | 183.1.0.4 |
| 35 | No Label | 200.0.1.0/24 | 0 | | Se0/0/0 | 183.1.0.4 |
| 36 | No Label | 200.0.2.0/24 | 0 | | Se0/0/0 | 183.1.0.4 |
| 37 | No Label | 200.0.3.0/24 | 0 | | Se0/0/0 | 183.1.0.4 |
| 38 | No Label | 204.12.1.0/24 | 0 | | Fa0/0 | 183.1.105.10 |

Rack1R5#show mpls forwarding-table | i 150.1

| | | | | | | |
|----|-----------|---------------|---|--|---------|--------------|
| 17 | No Label | 150.1.1.0/24 | 0 | | Fa0/0 | 183.1.105.10 |
| 18 | No Label | 150.1.2.0/24 | 0 | | Fa0/0 | 183.1.105.10 |
| 19 | No Label | 150.1.3.0/24 | 0 | | Fa0/0 | 183.1.105.10 |
| 20 | No Label | 150.1.3.3/32 | 0 | | Se0/0/0 | 183.1.0.3 |
| 21 | Pop Label | 150.1.4.4/32 | 0 | | Se0/0/0 | 183.1.0.4 |
| 22 | 22 | 150.1.6.6/32 | 0 | | Se0/0/0 | 183.1.0.4 |
| 23 | No Label | 150.1.7.0/24 | 0 | | Fa0/0 | 183.1.105.10 |
| 24 | No Label | 150.1.8.0/24 | 0 | | Fa0/0 | 183.1.105.10 |
| 25 | No Label | 150.1.10.0/24 | 0 | | Fa0/0 | 183.1.105.10 |

Rack1R6#show mpls forwarding-table | e 150.1

| Local Label | Outgoing Label or VC | Prefix or Tunnel Id | Bytes Switched | Label | Outgoing interface | Next Hop |
|-------------|----------------------|---------------------|----------------|-------|--------------------|------------|
| 25 | No Label | 183.1.0.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 26 | No Label | 183.1.17.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 27 | No Label | 183.1.28.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 28 | No Label | 183.1.39.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 29 | No Label | 183.1.45.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 30 | No Label | 183.1.105.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 31 | No Label | 183.1.107.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 32 | No Label | 183.1.123.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 33 | No Label | 192.10.1.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 34 | No Label | 200.0.0.0/24 | 0 | | Se0/0/0 | 54.1.1.254 |
| 35 | No Label | 200.0.1.0/24 | 0 | | Se0/0/0 | 54.1.1.254 |
| 36 | No Label | 200.0.2.0/24 | 0 | | Se0/0/0 | 54.1.1.254 |
| 37 | No Label | 200.0.3.0/24 | 0 | | Se0/0/0 | 54.1.1.254 |
| 38 | No Label | 204.12.1.0/24 | 0 | | Fa0/0 | 183.1.46.4 |

Rack1R6#show mpls forwarding-table | i 150.1

| | | | | | | |
|----|-----------|---------------|---|--|-------|------------|
| 16 | 17 | 150.1.1.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 17 | 18 | 150.1.2.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 18 | 19 | 150.1.3.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 19 | 20 | 150.1.3.3/32 | 0 | | Fa0/0 | 183.1.46.4 |
| 20 | Pop Label | 150.1.4.4/32 | 0 | | Fa0/0 | 183.1.46.4 |
| 21 | 21 | 150.1.5.5/32 | 0 | | Fa0/0 | 183.1.46.4 |
| 22 | 23 | 150.1.7.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 23 | 24 | 150.1.8.0/24 | 0 | | Fa0/0 | 183.1.46.4 |
| 24 | 25 | 150.1.10.0/24 | 0 | | Fa0/0 | 183.1.46.4 |

Task 4.2 Solution

Before you Begin

If you require basic hands-on practice for MP-BGP, check out the MPLS VPN section in VOL1 for the scenarios named **MP-BGP VPNv4** and **MP-BGP Prefix Filtering**. Most likely you will not see complicated VPN scenarios in the R&S exam, but you should master the fundamental VRF and VPNv4 configuration settings.

```
R5:
!
! Configure VRF VPN_A to export RT 100:5 and import 100:6 routes
!
ip vrf VPN_A
  rd 100:5
  route-target export 100:5
  route-target import 100:6
!
! Assign the VRF to a new Loopback interface
!
interface Loopback 1
  ip vrf forwarding VPN_A
  ip address 172.16.5.5 255.255.255.0
!
! Enable VPNv4 prefixes exchange between R4 and R5
!
router bgp 100
  address-family vpnv4
  neighbor 150.1.6.6 activate
  neighbor 150.1.6.6 send-community extended
  address-family ipv4 unicast vrf VPN_A
  redistribute connected
```

R6:

```
!  
! Configure VRF VPN_B to export RT 100:6 and import 100:5 routes  
!  
ip vrf VPN_B  
  rd 100:6  
  route-target export 100:6  
  route-target import 100:5  
!  
! Assign the VRF to a new Loopback interface  
!  
  
interface Loopback 1  
  ip vrf forwarding VPN_B  
  ip address 192.168.6.6 255.255.255.0  
!  
! Enable VPNv4 prefixes exchange between R4 and R5  
!  
router bgp 100  
  address-family vpnv4  
    neighbor 150.1.5.5 activate  
    neighbor 150.1.5.5 send-community extended  
  address-family ipv4 unicast vrf VPN_B  
  redistribute connected
```

Task 4.2 Breakdown

The task calls for establishing a BGP VPNv4 session between R5 and R6 for the purpose of VPNv4 prefix exchange. Two VRFs are to be created – one in R5 and another in R6. Notice that VRF VPN_A uses RD/RT of 100:5 and VPN_B in R6 uses RD/RT of 100:6. Since the goal is to make the two VPNs communicate, every VRF should import other VRF routes based on RT.

Since there is no dynamic routing in the VPNs, both BGP processes are configured to redistribute connected routes under the respective VRF contexts. The Loopback interfaces are created according with the task requirements and assigned to the respective VRFs.

Further Learning

MP-BGP is the core of MPLS VPN and you may want to get a better understanding of this technology. MP-BGP allows BGP to carry any “routable” protocol’s reachability information in BGP. Best path selection is still performed based on the BGP attributes, only for the matching “multiprotocol prefixes”. To better understand how MP-BGP works, start by recalling the classic BGP (RFC1771) UPDATE message format. It consists of the following sections:

UPDATE = [Withdrawn prefixes (Optional)] + [Path Attributes] + [NLRIs].

The Withdrawn Prefixes and NLRIs are formatted as *IPv4 prefixes*, and their structure does not support any other routable protocols. The Path Attributes (e.g. AS_PATH, ORIGIN, LOCAL_PREF, and NEXT_HOP) are associated with all NLRIs packed in the update; prefixes sharing different set of path attributes should be carried in a separate UPDATE message. Also, notice that NEXT_HOP is a BGP attribute, and it contains an IPv4 address as well. In order to introduce support for non-IPv4 network protocols into BGP, two new optional transitive *Path Attributes* have been added to BGP. Notice that this does not affect the original UPDATE packet formatting at all. The first attribute is known as MP_REACH_NLRI. It has the following structure:

MP UPDATE = [AFI/SAFI] + [NEXT_HOP] + [NLRI].

Here NLRI stands for *Network Layer Reachability Information*. In this packet, both NEXT_HOP and NLRI are formatted according to the protocol encoded via the AFI/SAFI pair. This abbreviation stands for *Address Family Identifier* and *Subsequent Address Family Identifier*, respectively. For example, this could be an IPv6 or CLNS prefix. Thus, all information about non-IPv4 prefixes is encoded in a new BGP Path Attribute.

A typical MP BGP UPDATE message that contains MP_REACH_NLRI attributes would have no “classic” IPv4 NEXT_HOP attribute and no “Withdrawn Prefixes” or “NLRI” found in normal UPDATE messages – all information is conveyed in BGP attributes. For the next-hop calculations, the receiving BGP speaker should use the information found in the MP_REACH_NLRI attribute. However, the multi-protocol UPDATE message may contain other BGP path attributes such as AS_PATH, ORIGIN, MED, LOCAL_PREF and so on to allow the BGP speakers performing classic best-path selection. However, these attributes are associated with the non-IPv4 prefixes found in all attached MP_REACH_NLRI attributes.

The second attribute, MP_UNREACH_NLRI has a format similar to MP_REACH_NLRI, but lists the “multi-protocol” addresses to be removed – functions similar to the BGP Withdrawn Prefixes field. No other path attributes need to be sent with this attribute, an UPDATE message may simply contain the list of MP_UNREACH_NLRIs.

The list of supported AFIs may be found in RFC1700 (although it is obsolete now, it is still very informative). For example, AFI 1 stands for IPv4, AFI 2 stands for IPv6 etc. The subsequent AFI is needed to clarify the purpose of the information found in MP_REACH_NLRI. For example, SAFI value of 1 means the prefixes should be used for *unicast* forwarding, SAFI 2 means the prefixes are to be used for multicast RPF checks and SAFI 3 means the prefixes could be used for both purposes. Last, but not least – SAFI of 128 means MPLS labeled VPN address.

Just as a reminder, the BGP process would perform a separate best-path election process for “classic” IPv4 prefixes and every AFI/SAFI pair prefixes separately, based on the respective path attributes. This allows for independent route propagation for the addresses found in different families. Since a given BGP speaker may not support particular network protocols, the list of supported AFI/SAFI pairs is advertised using the BGP capabilities feature (another BGP extension), and the particular network protocol information is only propagated if both speakers support it.

 **Deep Dive**

- What are the benefits and drawbacks of using MP-BGP over running native IGP adjacencies across MPLS tunnels?
- How are MP-BGP labels used together with IGP labels to create the MPLS label stack?

Task 4.2 Verification

Check the BGP peering session between R6 and R5:

```
Rack1R6#show bgp vpv4 unicast all summary
BGP router identifier 150.1.6.6, local AS number 100
BGP table version is 5, main routing table version 5
3 network entries using 468 bytes of memory
3 path entries using 204 bytes of memory
6/2 BGP path/bestpath attribute entries using 1008 bytes of memory
1 BGP AS-PATH entries using 24 bytes of memory
2 BGP extended community entries using 48 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
Bitfield cache entries: current 1 (at peak 1) using 32 bytes of memory
BGP using 1784 total bytes of memory
BGP activity 14/0 prefixes, 20/6 paths, scan interval 15 secs

Neighbor          V          AS MsgRcvd MsgSent   TblVer  InQ  OutQ
Up/Down  State/PfxRcd
150.1.5.5        4           100     153     149       5    0    0
00:07:25         1
```

Check that prefixes have been exchanged over BGP:

```
Rack1R6#show bgp vpv4 unicast all
BGP table version is 5, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop          Metric LocPrf Weight Path
Route Distinguisher: 100:5
*>i172.16.5.0/24    150.1.5.5           0     100     0 ?
Route Distinguisher: 100:6 (default for vrf VPN_B)
*>i172.16.5.0/24    150.1.5.5           0     100     0 ?
*> 192.168.6.0      0.0.0.0             0           32768 ?
```

Rack1R5#show bgp vpnv4 unicast all summary

BGP router identifier 150.1.5.5, local AS number 100
 BGP table version is 5, main routing table version 5
 3 network entries using 468 bytes of memory
 3 path entries using 204 bytes of memory
 8/2 BGP path/bestpath attribute entries using 1344 bytes of memory
 3 BGP AS-PATH entries using 72 bytes of memory
 2 BGP extended community entries using 48 bytes of memory
 0 BGP route-map cache entries using 0 bytes of memory
 0 BGP filter-list cache entries using 0 bytes of memory
 Bitfield cache entries: current 3 (at peak 3) using 96 bytes of memory
 BGP using 2232 total bytes of memory
 BGP activity 24/0 prefixes, 24/0 paths, scan interval 15 secs

| Neighbor | V | AS | MsgRcvd | MsgSent | TblVer | InQ | OutQ |
|-----------|--------------|-----|---------|---------|--------|-----|------|
| Up/Down | State/PfxRcd | | | | | | |
| 150.1.6.6 | 4 | 100 | 150 | 155 | 5 | 0 | 0 |
| 00:08:47 | 1 | | | | | | |

Rack1R5#show bgp vpnv4 unicast all

BGP table version is 5, local router ID is 150.1.5.5
 Status codes: s suppressed, d damped, h history, * valid, > best, i -
 internal,
 r RIB-failure, S Stale
 Origin codes: i - IGP, e - EGP, ? - incomplete

| Network | Next Hop | Metric | LocPrf | Weight | Path |
|--|-----------|--------|--------|--------|------|
| Route Distinguisher: 100:5 (default for vrf VPN_A) | | | | | |
| *> 172.16.5.0/24 | 0.0.0.0 | 0 | | 32768 | ? |
| *>i192.168.6.0 | 150.1.6.6 | 0 | 100 | | 0 ? |
| Route Distinguisher: 100:6 | | | | | |
| *>i192.168.6.0 | 150.1.6.6 | 0 | 100 | | 0 ? |

Check the label stacks for VPN prefixes in R5 and R6 (there should be two labels on either side, due to an intermediate router between R5 and R6):

```
Rack1R6#show ip cef vrf VPN_B 172.16.5.5
172.16.5.0/24
  nexthop 183.1.46.4 FastEthernet0/0 label 17 29
```

```
Rack1R5#show ip cef vrf VPN_A 192.168.6.6
192.168.6.0/24
  nexthop 183.1.0.4 Serial0/0/0 label 18 21
```

Do a ping and a traceroute to VPN prefixes:

```
Rack1R5#ping vrf VPN_A 192.168.6.6 source loopback 1
```

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 192.168.6.6, timeout is 2 seconds:

Packet sent with a source address of 172.16.5.5

!!!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 60/60/64 ms

```
Rack1R5#traceroute vrf VPN_A 192.168.6.6 source loopback 1
```

Type escape sequence to abort.

Tracing the route to 192.168.6.6

```
 1 183.1.0.4 [MPLS: Labels 18/21 Exp 0] 64 msec 60 msec 60 msec
 2 192.168.6.6 32 msec * 28 msec
```

Task 5.1 Solution

Before you Begin

Notice that if you are doing VOL2 labs for the first time and feel like your hands-on experience is lacking, you may even choose to skip multicasting until the moment you feel solid with the core topics (L2/L3, IGP, BGP). This particular section assumes your working knowledge of PIM SM/DM and AutoRP. You may find these technologies covered in the VOL1's section **Multicast** scenarios **PIM Sparse Mode, PIM Sparse-Dense Mode, and Auto-RP**.

Next, before you start working with this scenario, the active multicast topology should be discovered using the commands **show ip pim interface** and **show ip pim neighbor**. These commands allow you to discover the transit and stub multicast interfaces.

```
Rack1R1#show ip pim interface
```

| Address | Interface | Ver/ Mode | Nbr Count | Query Intvl | DR Prior | DR |
|----------|-----------|--------------|--------------|----------------|-------------|----|
| Rack1R1# | | | | | | |

```
Rack1R2#show ip pim interface
```

| Address | Interface | Ver/ Mode | Nbr Count | Query Intvl | DR Prior | DR |
|---------------|-----------------|--------------|--------------|----------------|-------------|--------------|
| 183.1.1.28.2 | FastEthernet0/0 | v2/SD | 0 | 30 | 1 | 183.1.1.28.2 |
| 183.1.1.123.2 | Serial0/0 | v2/SD | 1 | 30 | 1 | |
| 183.1.1.123.3 | | | | | | |

```
Rack1R2#show ip pim neighbor
```

PIM Neighbor Table

Mode: B - Bidir Capable, DR - Designated Router, N - Default DR Priority,
S - State Refresh Capable

| Neighbor Address | Interface | Uptime/Expires | Ver | DR Prio/Mode |
|---------------------|-----------|-------------------|-----|-----------------|
| 183.1.1.123.3 | Serial0/0 | 05:26:02/00:01:20 | v2 | 1 / DR S |

R2 connects to R3:

```
Rack1R3#show ip pim interface
```

| Address | Interface | Ver/ Mode | Nbr Count | Query Intvl | DR Prior | DR |
|---------------|-----------------|--------------|--------------|----------------|-------------|------------|
| 204.12.1.3 | FastEthernet0/0 | v2/SD | 0 | 30 | 1 | 204.12.1.3 |
| 183.1.1.123.3 | Serial1/0 | v2/SD | 1 | 30 | 1 | |
| 183.1.1.123.3 | | | | | | |
| 183.1.1.0.3 | Serial1/1 | v2/SD | 2 | 30 | 1 | |
| 183.1.1.123.2 | | | | | | |

```
Rack1R3#show ip pim neighbor
```

```
PIM Neighbor Table
```

```
Mode: B - Bidir Capable, DR - Designated Router, N - Default DR Priority,
      S - State Refresh Capable
```

| Neighbor Address | Interface | Uptime/Expires | Ver | DR Prio/Mode |
|------------------|-----------|-------------------|-----|--------------|
| 183.1.123.2 | Serial1/0 | 05:26:15/00:01:38 | v2 | 1 / S |
| 183.1.123.2 | Serial1/1 | 02:51:28/00:01:38 | v2 | 1 / DR S |
| 183.1.0.5 | Serial1/1 | 02:51:56/00:01:28 | v2 | 1 / S |

R3 connects to R2 and R5:

```
Rack1R4#show ip pim interface
```

| Address | Interface | Ver/Mode | Nbr Count | Query Intvl | DR Prior | DR |
|---------|-----------|----------|-----------|-------------|----------|----|
| | | | | | | |

```
Rack1R5#show ip pim interface
```

| Address | Interface | Ver/Mode | Nbr Count | Query Intvl | DR Prior | DR |
|-------------|-----------------|----------|-----------|-------------|----------|----|
| 183.1.105.5 | FastEthernet0/0 | v2/SD | 0 | 30 | 1 | |
| 183.1.105.5 | | | | | | |
| 183.1.0.5 | Serial0/0/0 | v2/SD | 3 | 30 | 1 | |
| 183.1.123.3 | | | | | | |

```
Rack1R5#show ip pim neighbor
```

```
PIM Neighbor Table
```

```
Mode: B - Bidir Capable, DR - Designated Router, N - Default DR Priority,
      S - State Refresh Capable
```

| Neighbor Address | Interface | Uptime/Expires | Ver | DR Prio/Mode |
|------------------|-------------|-------------------|-----|--------------|
| 183.1.0.3 | Serial0/0/0 | 02:45:28/00:01:18 | v2 | 1 / S |
| 183.1.123.3 | Serial0/0/0 | 05:26:09/00:01:43 | v2 | 1 / DR S |
| 183.1.123.2 | Serial0/0/0 | 05:26:39/00:01:15 | v2 | 1 / S |

R5 connects to R2 and R3. R6 has no IPv4 multicast interfaces:

```
Rack1R6#show ip pim interface
```

| Address | Interface | Ver/Mode | Nbr Count | Query Intvl | DR Prior | DR |
|---------|-----------|----------|-----------|-------------|----------|----|
| | | | | | | |

You may want to mark the interfaces that are enabled for multicast on your diagram with a different color so you can quickly track the multicast traffic flow. Make sure you have IGPs outlined on the same diagram, this often helps spotting potential multicast problems.

```
R2:
!
! Configure R2 as the Auto-RP MA
!
interface Loopback0
 ip pim sparse-mode
!
ip pim send-rp-discovery Loopback0 scope 16
```

```
R3:
!
! Configure R3 as the RP advertising itself via Auto-RP
!
interface Loopback0
 ip pim sparse-mode
!
ip pim send-rp-announce Loopback0 scope 16
```

Task 5.1 Breakdown

The main work to be done in this task is discovering the topology. Setting R2 and R3 as the Auto-RP components is straight-forward and directly based on the task requirements.

Further Learning

Multicasting is not a major part of the CCIE R&S exam, but you should definitely expect to see it in your exam. To solidify your fundamental multicast knowledge, you may want to complete all scenarios from **VOL1's Multicast** section up to and including the **AutoRP and RP/MA Placement**. Covering these scenarios should make you comfortable with most multicast topics, and allow you to fully benefit from practicing the VOL2 multicast sections.

Deep Dive

- Why was Auto-RP used with PIMv1?
- Why do you need to enable PIM on the Loopback interfaces for Auto-RP?

Task 5.1 Verification

Verify that the RP mapping information has been disseminated to routers:

```
Rack1R2#show ip pim rp mapping
```

```
PIM Group-to-RP Mappings
```

```
This system is an RP-mapping agent (Loopback0)
```

```
Group(s) 224.0.0.0/4
```

```
RP 150.1.3.3 (?), v2v1
```

```
Info source: 150.1.3.3 (?), elected via Auto-RP
```

```
Uptime: 00:03:26, expires: 00:02:31
```

```
Rack1R3#show ip pim rp mapping
```

```
PIM Group-to-RP Mappings
```

```
This system is an RP (Auto-RP)
```

```
Group(s) 224.0.0.0/4
```

```
RP 150.1.3.3 (?), v2v1
```

```
Info source: 150.1.2.2 (?), elected via Auto-RP
```

```
Uptime: 00:04:03, expires: 00:02:53
```

```
Rack1R5#show ip pim rp mapping
```

```
PIM Group-to-RP Mappings
```

Note

R5 does not have any group-RP mapping information, although it receives the information. Mapping agent announcements are dropped by R5 because of the RPF failure towards the MA. You can see this by issuing a `debug ip pim` and wait for 60 seconds to catch on MA advertisement. The RPF check failure will be resolved in the following task anyways:

```
Rack1R5#debug ip pim
```

```
PIM debugging is on
```

```
IP(0): s=150.1.2.2 (Serial0/0/0) d=224.0.1.40 id=26632, ttl=15,  
prot=17, len=52(48), not RPF interface
```

```
PIM(0): Prune Serial0/0/0/224.0.1.40 from (150.1.2.2/32, 224.0.1.40)
```

```
Rack1R5#undebug ip pim
```

Task 5.2 Solution

Before you Begin

The scenario deals with Multicast RPF failure. If you seek better understanding of Multicast RPF failures, you should examine **VOL1's Multicast > RPF Failure** scenario as well as reading INE's blog post "Troubleshooting Multicast RPF Failure":

<http://blog.ine.com/2008/01/02/troubleshooting-multicast-rpf-failure/>

which provides some additional guidance on RPF issue troubleshooting.

R5:

```
!  
! Join R5's interface to the multicast route  
!  
interface FastEthernet0/0  
  ip igmp join-group 226.26.26.26  
!  
! Create static m-route to fix the RPF failure  
!  
ip mroute 0.0.0.0 0.0.0.0 183.1.0.3
```

Task 5.2 Breakdown

The only issue in this task is that multicast traffic is sourced off VLAN 28, which R5 prefers to reach via EIGRP. However, the multicast packets are coming from R1 and toward R5's Frame-Relay interface. This will cause RPF failure in R5, so there must be a workaround.

- 1) You may use simple static mroute in R5 and let all RPF checks be performed via R3. This is a viable solution if the task/lab does not prohibit static mroutes.
- 2) If using static multicast routes is not an option, you may tune IGP AD for the VLAN28 prefix or filter it in EIGRP. This will automatically make R5 prefer it via OSPF and R3.

In our case we use a static mroute, but you may experiment with other approaches as well.

Deep Dive

- How are mroutes different from static routes?
- Do you need to have PIM enabled on the interface using the IGMP join command?
- What is the common source of RPF problems in the lab?

Task 5.2 Verification

Before the static mroute is configured on R5:

```
Rack1R2#ping
Protocol [ip]:
Target IP address: 226.26.26.26
Repeat count [1]: 100
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Interface [All]: Serial0/0
Time to live [255]:
Source address: 183.1.28.2
...
Rack1R5# debug ip mpacket
IP(0): s=183.1.28.2 (Serial0/0/0) d=226.26.26.26 id=165, ttl=254,
prot=1, len=104(100), not RPF interface
IP(0): s=183.1.28.2 (Serial0/0/0) d=226.26.26.26 id=166, ttl=254,
prot=1, len=104(100), not RPF interface

Rack1R5#sh ip mroute
<output omitted>

(183.1.28.2, 226.26.26.26), 00:00:15/00:02:44, flags: L
  Incoming interface: FastEthernet0/0, RPF nbr 183.1.105.10
  Outgoing interface list:
    Serial0/0/0, Forward/Sparse-Dense, 00:00:16/00:00:00
```

After the static mroute is configured:

```
Rack1R2#ping
Protocol [ip]:
Target IP address: 226.26.26.26
Repeat count [1]: 100
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Interface [All]: Serial0/0
Time to live [255]:
Source address: 183.1.28.2

Reply to request 0 from 183.1.0.5, 64 ms
Reply to request 0 from 183.1.0.5, 192 ms
Reply to request 1 from 183.1.0.5, 60 ms
Reply to request 1 from 183.1.0.5, 188 ms

Rack1R5#sh ip mroute
<output omitted>

(183.1.28.2, 226.26.26.26), 00:00:15/00:02:59, flags: LJT
  Incoming interface: Serial0/0/0, RPF nbr 183.1.0.3, Mroute
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 00:00:16/00:02:54
```

Task 5.3 Solution

Before You Begin

If you are not familiar with IGMP filtering features, you may want to practice the VOL1's **Multicast > IGMP Filtering** scenario, which shows the intended use of this feature, as well as provides some additional information. Some additional features are available for the IGMPv3 case, e.g. filtering based on the traffic source.

R3:

```
!  
! Access-list to be used for IGMP filtering  
!  
access-list 1 deny 239.0.0.0 0.255.255.255  
access-list 1 permit any  
!  
! IGMP access-group (filter) applied to the interface  
!  
interface FastEthernet0/0  
 ip igmp access-group 1
```

Task 5.3 Breakdown

The key for this task is to recall the Administratively scoped multicast group range and create an access-list denying it. After this, the access-list should be used as an IGMP report filter on R3's interface connected to VLAN33.

Task 5.3 Verification

Check the IGMP access-group associated with the interface:

```
Rack1R3#show ip igmp interface FastEthernet 0/0 | include access
  Inbound IGMP access group is 1
```

```
Rack1R3#show ip access-lists 1
Standard IP access list 1
 10 deny 239.0.0.0, wildcard bits 0.255.255.255
 20 permit any (1 match)
```

Task 6.1 Solution

Before You Begin

This task assumes you have a good working knowledge of IOS access-lists and understand the nature of TCP SYN Flooding attacks. For initial practice with IOS access-lists, complete the **VOL1 > Security** scenarios named **Traffic Filtering with Standard Access-Lists**, **Traffic Filtering with Extended Access-List**, and **Logging with Access-Lists**.

If you are looking for the TCP SYN Flooding attack description, check the Wikipedia article named "SYN Flood" at http://en.wikipedia.org/wiki/SYN_flood.

R3:

```
ip access-list extended SYN_ATTACK
 permit tcp any host 183.1.28.100 eq www syn log-input
 permit ip any any
!
interface FastEthernet0/0
 ip access-group SYN_ATTACK in
```

SW4:

```
ip access-list extended SYN_ATTACK
 permit tcp any host 183.1.28.100 eq www syn log-input
 permit ip any any
!
interface Vlan102
 ip access-group SYN_ATTACK in
```

Task 6.1 Breakdown

This scenario requires logging potential DOS attack attempts, specifically registering the TCP SYN packets from a known host. In order to accomplish this, a special access-list is created with two permit entries. The first entry matches the suspicious packets and logs the source along with MAC-address (log-input) per the task requirements. The second entry simply permits both packets. We apply the access lists on R3 and SW4 on the mentioned interfaces.

Further Learning

If you are looking for additional features available with access-list logging, you may want to check **VOL1's Security > Logging with Access-Lists** that provides additional examples.

Deep Dive

- What is the purpose of a TCP SYN attack?
- Is there a host-based solution to protect against TCP SYN flooding?
- How can you distinguish a TCP SYN attacker from a normal session?

Task 6.1 Verification

Generate TCP SYN packets from BB2 and watch the ACL log hits on SW2:

```
BB2>telnet 183.1.28.100 80
Trying 183.1.28.100, 80 ...
```

```
Rack1SW2#show logging
```

```
<output omitted>
%SEC-6-IPACCESSLOGP: list SYN_ATTACK permitted tcp 192.10.1.254(18518) (Vlan102
0010.7b3a.14cc) -> 183.1.28.100(80), 1 packet
```

Task 6.2 Solution

Before You Begin

This task depends on the previous one, as they both use the same access-list. It is possible to complete just this scenario, without ever configuring the solution for the previous task, but you should notice the dependency.

R3:

```
!  
! Modify the existing access-list to deny packets coming from spoofed  
! addresses  
!  
no ip access-list extended SYN_ATTACK  
ip access-list extended SYN_ATTACK  
  deny ip 183.1.0.0 0.0.255.255 any  
  permit tcp any host 183.1.28.100 eq www syn log-input  
  permit ip any any
```

SW4:

```
!  
! Modify the existing access-list to deny packets coming from spoofed  
! addresses  
!  
no ip access-list extended SYN_ATTACK  
ip access-list extended SYN_ATTACK  
  deny ip 183.1.0.0 0.0.255.255 any  
  permit tcp any host 183.1.28.100 eq www syn log-input  
  permit ip any any
```

R6:

```
!  
! Create an access-list in R6 that drops spoofed packets  
!  
ip access-list extended SYN_ATTACK  
  deny ip 183.1.0.0 0.0.255.255 any  
  permit ip any any  
!  
interface Serial0/0/0  
  ip access-group SYN_ATTACK i
```

Task 6.2 Breakdown

The task requires you to configure ingress edge filtering, preventing packets coming from spoofed local addresses. The whole /16 range of your pod is being filtered when used as source on the external links, as you don't expect your network's own traffic to come from outside. Another way to implement this would be by using uRPF filtering. Notice that you need to delete and recreate access-lists in R3 and SW4, as the same access-lists are currently used for tracking.

Further Learning

You may refer to RFCs 2827 and RFC 3704 for more information on ingress filtering.

Deep Dive

- How could outside systems spoofing your addresses be dangerous for your network?
- What is the main problem with ingress filtering based on access-lists?

Task 6.2 Verification

Check the access-lists on every participating router:

```
Rack1R3#sh ip access-lists | beg SYN_ATTACK
Extended IP access list SYN_ATTACK
 10 deny ip 183.1.0.0 0.0.255.255 any
 20 permit tcp any host 183.1.28.100 eq www syn log-input
 30 permit ip any any (3 matches)
```

```
Rack1R6#sh ip access-lists | beg SYN_ATTACK
Extended IP access list SYN_ATTACK
 10 deny ip 183.1.0.0 0.0.255.255 any
 20 permit ip any any (20 matches)
```

```
Rack1SW2#sh ip access-lists | beg SYN_ATTACK
Extended IP access list SYN_ATTACK
 10 deny ip 183.1.0.0 0.0.255.255 any
 20 permit tcp any host 183.1.28.100 eq www syn log-input
 30 permit ip any any (19 matches)
```

Task 6.3 Solution

SW4:

```
interface Vlan102
  no ip unreachablees
  no ip mask-reply
```

Task 6.3 Breakdown

The two ICMP features that are enabled by default on any IP interfaces are sending ICMP mask replies and ICMP unreachable messages for dropped packets. The first feature allows the potential attacker to find information about the local network mask and figure out the default gateway/number of hosts on the subnet. The second feature allows for learning about filtered or un-routable destinations and thus find out about network map. Both should be normally disabled for security concerns on the interfaces facing public networks. You may read more about ICMP message format and types in RFC 792, and specifically about the purpose of the ICMP Mask Request/Reply in RFC 950.

Task 6.3 Verification

Check the interface settings to verify your configuration:

```
Rack1SW4#show ip interface vlan 102
Vlan102 is up, line protocol is up
<snip>
  ICMP unreachablees are never sent
  ICMP mask replies are never sent
<snip>
```

Task 6.4 Solution

Before You Begin

This scenario uses TTL filtering, which is a relatively new IOS feature. You need to completely understand how IOS decrements the TTL field. Also, keep in mind that outbound access-lists only apply to the transit packets, and not the packets originated by the router itself.

R4:

```
!  
! Create an access list to drop packets with TTL < 3  
!  
ip access-list extended TTL  
  deny ip any any ttl lt 3 log  
  permit ip any any  
  
!  
! Apply the access-list egress to every interface  
!  
interface FastEthernet0/0  
  ip access-group TTL out  
!  
interface FastEthernet0/1  
  ip access-group TTL out  
!  
interface Serial0/0/0  
  ip access-group TTL out  
!  
! The below settings are needed to enable informational logging  
! to the router's console and buffer  
!  
logging on  
logging console informational  
logging buffered informational  
no logging monitor
```

Task 6.4 Breakdown

The task calls to a new access-list feature that allows matching on the packet TTL field. A special access-list is created and applied to all interfaces in the outgoing direction. Notice that the TTL is decremented before the outgoing access-list check but remains unmodified on the ingress ACL check. Thus, if a packet with TTL 10 comes into a router, the ingress ACL will see it with a TTL of 10, while egress will see it with a TTL of 9.

Deep Dive

- How can TTL filtering be used to prevent spoofing?

Task 6.4 Verification

Send out traceroute packets with different TTL values. The first one using TTL up from 4:

```
Rack1R6#traceroute 150.1.1.1 ttl 4 10
```

```
Type escape sequence to abort.  
Tracing the route to 150.1.1.1
```

```
 4 183.1.107.7 28 msec 32 msec 32 msec  
 5 183.1.17.1 28 msec * 28 msec
```

The second one use TTL starting up from one

```
Rack1R6#traceroute 150.1.1.1 ttl 1 10
```

```
Type escape sequence to abort.  
Tracing the route to 150.1.1.1
```

```
 1 183.1.46.4 0 msec 4 msec 0 msec  
 2 183.1.46.4 !A * !A
```

```
Rack1R4#show logging
```

```
Syslog logging: enabled (0 messages dropped, 3 messages rate-limited,  
0 flushes, 0 overruns, xml disabled, filtering  
disabled)
```

```
<snip>
```

```
Log Buffer (4096 bytes):
```

```
%SEC-6-IPACCESSLOGP: list TTL denied udp 183.1.46.6(0) -> 150.1.5.5(0),  
1 packet  
%SEC-6-IPACCESSLOGP: list TTL denied udp 183.1.46.6(0) -> 150.1.1.1(0),  
1 packe
```

Task 7.1 Solution

Before You Begin

You need to be familiar with RMON concepts at a basic level, for example by completing **VOL1's System Management > RMON Alarms** section. It illustrates a detailed scenario featuring the use of RMON for SNMP MIB variable monitoring.

R2:

```
!  
! Create an alarm for rising and falling thresholds  
!  
rmon alarm 1 ifEntry.11.1 60 delta rising-threshold 15000 1 falling-  
threshold 5000 2  
  
!  
! Create two events generated on rising and falling thresholds  
!  
rmon event 1 trap IETRAP description "Above 15000 for ifInUcastPkts"  
rmon event 2 trap IETRAP description "Below 5000 for ifInUcastPkts"  
  
!  
! Configure the SNMP server to report the errors to  
!  
snmp-server host 183.1.17.100 IETRAP
```

Task 7.1 Breakdown

This task explicitly names the MIB object, so we only have to set up the RMON alarm and events. In the case where only the interface parameter name is given, we could have proceeded using the command **show snmp mib ifmib ifindex** to find out the interface index to be associated with the parameter name. Notice that the delta method is selected for the variable sampling, as we are monitoring a value which has its value accumulating and not oscillating. Normally, you would like to collect the rate of changes for such variables, as absolute values do not reveal much useful information.

Deep Dive

- What is the benefit of using RMON over SNMP polling?
- What is the main purpose of “delta” sampling for RMON?

Task 7.1 Verification

Verify the RMON configuration:

```
Rack1R2#show rmon alarms
```

```
Alarm 1 is active, owned by config
Monitors ifEntry.11.1 every 60 second(s)
Taking delta samples, last value was 0
Rising threshold is 15000, assigned to event 1
Falling threshold is 5000, assigned to event 2
On startup enable rising or falling alarm
```

```
Rack1R2#show rmon events
```

```
Event 1 is active, owned by config
Description is Above 15000 for ifInUcastPkts
Event firing causes trap to community IETRAP,
last event fired at 0y0w0d,00:00:00,
Current uptime 0y0w0d,06:11:00
Event 2 is active, owned by config
Description is Below 5000 for ifInUcastPkts
Event firing causes trap to community IETRAP,
last event fired at 0y0w0d,00:00:00,
Current uptime 0y0w0d,06:11:00
```

```
Rack1R2#show snmp host
```

```
Notification host: 183.1.17.100 udp-port: 162 type: trap
user: IETRAP security model: v1
```

Task 7.2 Solution

Before You Begin

NTP is an important service that you should know how to set up without referencing the documentation. If you are unfamiliar with the topic, check **VOL1's** section **System Management** for the scenario named **NTP**.

R3:

```
ntp server 204.12.1.254
```

R6:

```
ntp server 54.1.1.254
```

R1, R2, and SW1:

```
ntp server 150.1.3.3
```

R4, R5, and SW4:

```
ntp server 150.1.6.6
```

R1:

```
!  
! Set up NTP peering b/w R1 and R2  
!  
ntp peer 150.1.2.2
```

Task 7.2 Breakdown

The task wording should be read carefully to figure out the NTP relationships. R3 and R6 are the “primary” masters in this task. All other routers take their time from either R3 or R6 and the primary masters are set for NTP peering relations. Notice that the `ntp peer` command could be configured on just one router, e.g. R1, as this type of association is bi-directional (one node is passive, another active, but both sides adjust their clocks).

Take into account the amount of time that may be needed for NTP to converge. It may be significant, especially if the clock offset between the client and server is high.

Task 7.2 Verification

Verify the NTP status and associations. Use the same commands on all NTP-enabled routers:

Rack1R3#show ntp status

```
Clock is synchronized, stratum 6, reference is 204.12.1.254
nominal freq is 249.5901 Hz, actual freq is 249.5904 Hz, precision is 2**18
reference time is CF1BB8A2.6BA2EAE2 (10:34:10.420 UTC Tue Feb 9 2010)
clock offset is -10.1820 msec, root delay is 45.56 msec
root dispersion is 14.57 msec, peer dispersion is 0.81 msec
```

Rack1R3#show ntp associations detail

```
204.12.1.254 configured, our_master, sane, valid, stratum 5
ref ID 172.16.4.1, time CF1BB80E.EC16708A (10:31:42.922 UTC Tue Feb 9 2010)
our mode client, peer mode server, our poll intvl 64, peer poll intvl 64
root delay 39.86 msec, root disp 3.57, reach 377, sync dist 27.161
delay 5.71 msec, offset -10.1820 msec, dispersion 0.81
precision 2**18, version 3
org time CF1BB8A2.684C7781 (10:34:10.407 UTC Tue Feb 9 2010)
rcv time CF1BB8A2.6BA2EAE2 (10:34:10.420 UTC Tue Feb 9 2010)
xmt time CF1BB8A2.6A23167E (10:34:10.414 UTC Tue Feb 9 2010)
filtdelay =      5.71      5.77      5.69      6.12      5.77      5.68      6.67      7.16
filtoffset =    -10.18     -9.82     -9.49     -8.82     -8.42     -7.81     -6.49     -5.03
filterror =       0.02       0.99       1.97       2.94       3.92       4.90       5.87       6.85
```

Rack1R6#show ntp status

```
Clock is synchronized, stratum 5, reference is 54.1.1.254
nominal freq is 250.0000 Hz, actual freq is 250.0017 Hz, precision is 2**24
reference time is CF1BC033.592CA6BD (11:06:27.348 UTC Tue Feb 9 2010)
clock offset is -0.0340 msec, root delay is 0.02 msec
root dispersion is 0.49 msec, peer dispersion is 0.18 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is -0.000006889 s/s
system poll interval is 64, last update was 114 sec ago.
```

Rack1R6#show ntp associations detail

```
54.1.1.254 configured, our_master, sane, valid, stratum 4
ref ID 127.127.7.1 , time CF1BC06A.E8431DF8 (11:07:22.907 UTC Tue Feb 9 2010)
our mode client, peer mode server, our poll intvl 64, peer poll intvl 64
root delay 0.00 msec, root disp 0.03, reach 77, sync dist 0.22
delay 0.00 msec, offset -34.0443 msec, dispersion 189.79
precision 2**18, version 4
org time CF1BC074.4B250284 (11:07:32.293 UTC Tue Feb 9 2010)
rcv time CF1BC074.58DCB11F (11:07:32.347 UTC Tue Feb 9 2010)
xmt time CF1BC074.513824FA (11:07:32.317 UTC Tue Feb 9 2010)
filtdelay = 0.02 0.02 0.03 0.03 0.02 0.02 0.00 0.00
filtoffset = -0.03 -0.03 -0.02 -0.01 -0.01 -0.00 0.00 0.00
filtererror = 0.00 0.00 0.00 0.00 0.00 0.00 16.00 16.00
minpoll = 6, maxpoll = 10
```

Now check the NTP peering relations between R1 and R2:

Rack1R1#show ntp associations detail

```
150.1.3.3 configured, authenticated, our_master, sane, valid, stratum 6
ref ID 204.12.1.254, time CF1BBAE2.67A75A3D (10:43:46.404 UTC Tue Feb 9 2010)
our mode client, peer mode server, our poll intvl 64, peer poll intvl 64
root delay 43.87 msec, root disp 13.06, reach 377, sync dist 77.591
delay 84.67 msec, offset 3.6351 msec, dispersion 0.26
precision 2**18, version 3
org time CF1BBB00.226B782D (10:44:16.134 UTC Tue Feb 9 2010)
rcv time CF1BBB00.2C53D616 (10:44:16.173 UTC Tue Feb 9 2010)
xmt time CF1BBB00.16944EA5 (10:44:16.088 UTC Tue Feb 9 2010)
filtdelay = 84.67 85.53 84.61 85.36 84.53 84.93 85.02 84.79
filtoffset = 3.64 3.47 4.06 3.20 3.58 3.55 4.05 3.77
filtererror = 0.02 0.99 1.97 2.94 3.92 4.90 5.87 6.85
```

150.1.2.2 configured, selected, sane, valid, stratum 7

```
ref ID 150.1.3.3, time CF1BBAFB.73378C49 (10:44:11.450 UTC Tue Feb 9 2010)
our mode active, peer mode passive, our poll intvl 64, peer poll intvl 64
root delay 129.32 msec, root disp 21.56, reach 377, sync dist 109.650
delay 46.26 msec, offset 38.5184 msec, dispersion 0.26
precision 2**18, version 3
org time CF1BBAFD.26250B5C (10:44:13.149 UTC Tue Feb 9 2010)
rcv time CF1BBAFD.2234EC81 (10:44:13.133 UTC Tue Feb 9 2010)
xmt time CF1BBAFD.164DD878 (10:44:13.087 UTC Tue Feb 9 2010)
filtdelay = 46.26 46.65 46.52 46.14 46.16 47.59 46.16 46.75
filtoffset = 38.52 38.58 38.39 37.82 37.79 37.67 37.73 37.56
filtererror = 0.02 0.99 1.97 2.94 2.96 2.98 2.99 3.01
```

Task 7.3 Solution

Before You Begin

If you are not familiar with NTP service authentication in Cisco IOS, we recommend looking into VOL1's scenario **System Management > NTP Authentication** to get a better understanding of NTP authentication. The procedure is not complicated, but there could be some confusion when you are configuring it for the first time.

R3 and R6:

```
!  
! Only the key should be defined in "servers"  
! as their time is not changed by client  
!  
ntp authentication-key 1 md5 CISCO
```

Note

For more recent T-train IOS releases, in order for NTP authentication to work, you need the command **ntp trusted-key** configured on the NTP server as well.

R6:

```
ntp trusted key 1
```

R1, R2, and SW1:

```
!  
! The clients need to have the key and the server defined.  
! The key should be trusted and authentication enabled, as  
! clients should validate the identity of the server, which  
! changes their clocks.  
!  
ntp authentication-key 1 md5 CISCO  
ntp authenticate  
ntp trusted-key 1  
ntp server 150.1.3.3 key 1
```

R1:

```
!  
! Authenticate NTP peers on R1 and R2  
!  
ntp peer 150.1.2.2 key 1
```

R2:

```
!  
! Authenticate NTP peers on R1 and R2  
!  
ntp peer 150.1.1.1 key 1
```

R4, R5, and SW4:

```
!  
! The clients need to have the key and the server defined.  
! The key should be trusted and authentication enabled, as  
! clients should validate the identity of the server, which  
! changes their clocks.  
!  
ntp authentication-key 1 md5 CISCO  
ntp authenticate  
ntp trusted-key 1  
ntp server 150.1.6.6 key 1
```

Task 7.3 Breakdown

NTP authentication should be configured differently on clients and servers. Only the client really authenticates the incoming packets. This is because the server never changes its time in response to client queries, but the client changes its time based on the server's responses. Notice that the server still needs to have the key defined with the SAME key ID as used by the client. This allows the server to select the proper key when responding to queries, and properly provide authentication information in responses.

When configuring authentication on a client, make sure you created and trusted the authentication key. After this, associate the key with the server – this is needed so that the client may associate the proper key id with the outgoing polls for this server.

Authenticating NTP peerings is a bit tricky – you need to configure both sides with the respective key ids and associate the key with the peer. This is required because in peering relations both nodes synchronize each other.

Deep Dive

- Why are key IDs important with NTP authentication?
- When do you need to trust an NTP authentication key?

Task 7.3 Verification

Note

Check the NTP associations on every NTP-enabled node. The output below demonstrates the status of R1 and R4, where the servers show as authenticated:

Rack1R1#show ntp associations detail

```
150.1.3.3 configured, authenticated, our_master, sane, valid, stratum 6
ref ID 204.12.1.254, time CCEC61CE.6070F38F (04:06:38.376 UTC Fri Dec 12 2008)
our mode client, peer mode server, our poll intvl 64, peer poll intvl 64
root delay 47.26 msec, root disp 11.40, reach 377, sync dist 74.097
delay 70.27 msec, offset 0.8069 msec, dispersion 3.94
precision 2**18, version 3
org time CCEC6203.7CB0702A (04:07:31.487 UTC Fri Dec 12 2008)
rcv time CCEC6203.8729CADF (04:07:31.527 UTC Fri Dec 12 2008)
xmt time CCEC6203.715D99BE (04:07:31.442 UTC Fri Dec 12 2008)
filtdelay =    84.85    84.67    84.37    84.37    70.27    69.08    69.27    69.96
filtoffset =    1.52    0.88    -0.17    -0.67    0.81    0.86    0.21    0.04
filtererror =    0.02    0.99    1.97    2.62    3.60    4.58    5.55    5.57
```

Rack1R4#show ntp associations detail

```
150.1.6.6 configured, authenticated, our_master, sane, valid, stratum 5
ref ID 54.1.1.254, time CCEC6217.A1919786 (04:07:51.631 UTC Fri Dec 12 2008)
our mode client, peer mode server, our poll intvl 64, peer poll intvl 64
root delay 29.75 msec, root disp 2.81, reach 377, sync dist 19.302
delay 3.05 msec, offset -1.2642 msec, dispersion 0.09
precision 2**18, version 3
org time CCEC621B.BE0A1D73 (04:07:55.742 UTC Fri Dec 12 2008)
rcv time CCEC621B.BEC170E5 (04:07:55.745 UTC Fri Dec 12 2008)
xmt time CCEC621B.BDE7063D (04:07:55.741 UTC Fri Dec 12 2008)
filtdelay =    3.05    3.08    3.10    3.45    3.17    3.14    3.13    3.23
filtoffset =   -1.26   -1.26   -1.28   -1.03   -0.99   -0.75   -0.23   -0.19
filtererror =    0.02    0.99    1.97    2.61    3.59    4.56    5.54    5.55
```

The output below shows that R2 has three authenticated sessions: one for the server, another for the peer, and the third one shows the dynamic association formed on an active request from R1:

Rack1R2#show ntp associations detail

```
150.1.3.3 configured, authenticated, our_master, sane, valid, stratum 6
ref ID 204.12.1.254, time CF1BBE22.6132EF95 (10:57:38.379 UTC Tue Feb 9 2010)
our mode client, peer mode server, our poll intvl 64, peer poll intvl 64
root delay 50.64 msec, root disp 17.47, reach 377, sync dist 88.211
delay 85.11 msec, offset -10.3962 msec, dispersion 2.59
precision 2**18, version 3
org time CF1BBE3B.61054694 (10:58:03.378 UTC Tue Feb 9 2010)
rcv time CF1BBE3B.70E9B5FA (10:58:03.441 UTC Tue Feb 9 2010)
xmt time CF1BBE3B.58860224 (10:58:03.345 UTC Tue Feb 9 2010)
filtdelay =    94.99    85.11    84.58    84.93   116.87    84.87    84.69    85.14
filtoffset =   -14.58   -10.40    -9.12    -8.92   -24.92    -9.10    -9.07    -8.62
filtererror =    0.02    0.99    1.97    2.94    3.92    4.90    5.87    6.85
```

```
150.1.1.1 configured, authenticated, selected, sane, valid, stratum 7
ref ID 150.1.3.3, time CF1BBE26.2E9C9C8A (10:57:42.182 UTC Tue Feb 9 2010)
our mode active, peer mode passive, our poll intvl 64, peer poll intvl 128
root delay 136.31 msec, root disp 67.63, reach 376, sync dist 165.298
delay 56.38 msec, offset -39.8359 msec, dispersion 1.27
precision 2**18, version 3
org time CF1BBE49.17F3AAB6 (10:58:17.093 UTC Tue Feb 9 2010)
rcv time CF1BBE49.296B4AE0 (10:58:17.161 UTC Tue Feb 9 2010)
xmt time CF1BBE56.58562E2C (10:58:30.345 UTC Tue Feb 9 2010)
filtdelay =    56.58    56.38    56.56    56.15    56.27    56.49    56.29    56.92
filtoffset =   -39.94   -39.84   -39.03   -38.96   -38.98   -38.98   -38.97   -38.70
filtererror =    0.79    0.79    1.77    1.79    1.80    1.82    1.83    1.85
```

```
183.1.123.1 dynamic, authenticated, selected, sane, valid, stratum 7
ref ID 150.1.3.3, time CF1BBE26.2E9C9C8A (10:57:42.182 UTC Tue Feb 9 2010)
our mode passive, peer mode active, our poll intvl 64, peer poll intvl 128
root delay 136.31 msec, root disp 67.63, reach 17, sync dist 4043.762
delay 56.30 msec, offset -39.9679 msec, dispersion 3879.84
precision 2**18, version 3
org time CF1BBE4D.18A06324 (10:58:21.096 UTC Tue Feb 9 2010)
rcv time CF1BBE4D.2A10D322 (10:58:21.164 UTC Tue Feb 9 2010)
xmt time CF1BBE38.584C7C3D (10:58:00.344 UTC Tue Feb 9 2010)
filtdelay =    56.30    56.21    95.23    0.00    0.00    0.00    0.00    0.00
filtoffset =   -39.97   -39.94   -57.92    0.00    0.00    0.00    0.00    0.00
filtererror =    0.34    1.31    2.29 16000.0 16000.0 16000.0 16000.0 16000.0
```

Task 7.4 Solution

Before You Begin

For an introduction to the various IOS accounting features, check **VOL1's IP Services** section. Specifically, examine every **Accounting** scenario such as **IP Precedence Accounting** which covers this scenario's case.

R2:

```
!  
! Account for precedence values of incoming/outgoing packets  
!  
interface Serial0/0  
  ip accounting precedence input  
  ip accounting precedence output  
!  
ip accounting-threshold 50000
```

R3:

```
!  
! Account for precedence values of incoming/outgoing packets  
!  
interface Serial1/0  
  ip accounting precedence input  
  ip accounting precedence output  
!  
ip accounting-threshold 50000
```

Task 7.4 Breakdown

There are many ways to set up accounting features, such as using Netflow or IP packet accounting. However, this scenario specifically points to the feature that counts packets and distributes them into statistics bins based on precedence values. This feature is enabled on a per-interface basis and collects the precedence counter for incoming and outgoing packets.

Deep Dive

- When accounting for IP packets, why is setting a threshold so important?
- Give an example of using IP precedence accounting.

Task 7.4 Verification

Verify precedence accounting. The precedence value of 6 below corresponds to the routing protocols that Cisco IOS normally tags with IPP of 6:

```
Rack1R2#show interfaces serial 0/0 precedence
```

```
Serial0/0
```

```
Input
```

```
Precedence 6: 114 packets, 8737 bytes
```

```
Output
```

```
Precedence 0: 1 packets, 114 bytes
```

```
Precedence 6: 119 packets, 8051 bytes
```

```
Rack1R3#show interfaces serial 1/0 prec
```

```
Serial1/0
```

```
Input
```

```
Precedence 6: 35 packets, 2706 bytes
```

```
Output
```

```
Precedence 0: 1 packets, 114 bytes
```

```
Precedence 6: 98 packets, 6966 bytes
```

Task 7.5 Solution

Before You Begin

The scenario prompts you toward using basic HSRP settings. If you are unfamiliar with this technology, check **VOL1's IP Services** section for **HSRP**.

R5:

```
!  
! Configure the HSRP group settings  
!  
interface FastEthernet0/0  
  standby 1 ip 183.1.105.254  
  standby 1 preempt  
  standby 1 track Serial0/0/0 100
```

SW4:

```
interface FastEthernet0/18  
  standby 1 ip 183.1.105.254  
  standby 1 priority 50  
  standby 1 preempt
```

Task 7.5 Breakdown

One may reasonably assume that tracking a multipoint Frame-Relay connection makes no sense as this does not detect a loss of a particular VC. However, the task specifically requires tracking of the physical interface, which is not something you would do in real life. The real world scenarios would probably call for use of IP SLA operations and ICMP-echo based tracking. In our case, R5's priority is lowered by 100 when the Frame-Relay interface goes down.

Further Learning

It is also recommended to learn about VRRP and GLBP in parallel with HSRP to compare the technologies and see their differences. There are **VRRP** and **GLBP** scenarios in **VOL1**, which provide in-depth coverage of the respective gateway redundancy protocols. Additionally, you may also want to learn about IP SLA and object tracking, which are often used together with the gateway redundancy protocols. These are covered in **IP SLA** and **Object Tracking** scenarios respectively.

Deep Dive

- What HSRP messages do you know?
- What is the purpose of the HSRP Coup message?

Task 7.5 Verification

Verify the HSRP configuration:

Rack1R5#show standby

```
FastEthernet0/0 - Group 1
  State is Active
    2 state changes, last state change 00:01:16
  Virtual IP address is 183.1.105.254
  Active virtual MAC address is 0000.0c07.ac01
    Local virtual MAC address is 0000.0c07.ac01 (v1 default)
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 1.896 secs
  Preemption enabled
  Active router is local
  Standby router is 183.1.105.10, priority 50 (expires in 7.892 sec)
  Priority 100 (default 100)
    Track interface Serial0/0/0 state Up decrement 100
```

Rack1R5(config)#interface Serial 0/0/0

Rack1R5(config-if)#shutdown

<output omitted>

```
%HSRP-6-STATECHANGE: FastEthernet0/0 Grp 1 state Active -> Speak
```

Rack1R5(config-if)#do show standby

```
FastEthernet0/0 - Group 1
  State is Standby
  <output omitted>
  Active router is 183.1.105.10, priority 50 (expires in 8.200 sec)
  Standby router is local
  Priority 0 (default 100)
    Track interface Serial0/0/0 state Down decrement 100
  IP redundancy name is "hsrp-Fa0/0-1" (default)
```


Task 7.6 Breakdown

The task does not provide any specific requirements on the address translation. Therefore, we utilize the simple solution that translates the pod's interface subnets into the gateway's outside IP address using NAT overloading. The network to be used in the access-list is explicitly defined in the task.

Further Learning

If you are looking for an exhaustive list of NAT options, you may want to go through all **VOL1's IP Services** section **NAT** scenarios. Additionally, you may want to check the post named "The Inside and Outside of NAT"

<http://blog.ine.com/2008/02/15/the-inside-and-outside-of-nat/>

on the INE blog if you are interested in the intrinsic difference between NAT domains and the purpose of the NAT Virtual Interface.

 **Deep Dive**

- Modify this scenario to use an NVI.
- What are extendable NAT entries and how are they different from standard?

Task 7.6 Verification

Verify the NAT translations:

```
Rack1R1#ping 204.12.1.254
```

Type escape sequence to abort.

```
Sending 5, 100-byte ICMP Echos to 204.12.1.254, timeout is 2 seconds:  
!!!!!
```

```
Rack1R3#sh ip nat translations
```

| Pro | Inside global | Inside local | Outside local | Outside global |
|------|-----------------|------------------|-------------------|-------------------|
| icmp | 204.12.1.3:3179 | 183.1.123.1:3179 | 204.12.1.254:3179 | 204.12.1.254:3179 |
| icmp | 204.12.1.3:3180 | 183.1.123.1:3180 | 204.12.1.254:3180 | 204.12.1.254:3180 |
| icmp | 204.12.1.3:3181 | 183.1.123.1:3181 | 204.12.1.254:3181 | 204.12.1.254:3181 |
| icmp | 204.12.1.3:3182 | 183.1.123.1:3182 | 204.12.1.254:3182 | 204.12.1.254:3182 |
| icmp | 204.12.1.3:3183 | 183.1.123.1:3183 | 204.12.1.254:3183 | 204.12.1.254:3183 |

Check for the NAT-enabled interfaces:

```
Rack1R3#show ip nat statistics
```

```
Total active translations: 1 (0 static, 1 dynamic; 1 extended)
```

```
Outside interfaces:
```

```
FastEthernet0/0
```

```
Inside interfaces:
```

```
FastEthernet0/1, Serial1/0, Serial1/1
```

```
Hits: 9 Misses: 1
```

```
CEF Translated packets: 10, CEF Punted packets: 0
```

```
Expired translations: 0
```

```
Dynamic mappings:
```

```
-- Inside Source
```

```
[Id: 1] access-list 2 interface FastEthernet0/0 refcount 1
```

```
Queued Packets: 0
```

Task 7.7 Solution

Before You Begin

The solution utilizes the Embedded Event Management feature. EEM is a powerful and complex technology, and learning to program EEM applets can take a considerable amount of time. However, from the perspective of the CCIE R&S exam, you just need to understand the basic EEM architecture and the fundamental event detectors. The applet programming language is relatively simple and you may find a lot of examples in the documentation CD. If you are looking for a “gentle start”, check out **VOL1’s IP Services** section and look for the various **EEM** scenarios presented there. They cover both the theoretical concepts and hands-on examples you may try to familiarize yourself with for the topic. For this particular scenario you need to familiarize yourself with the VOL1 scenario called **IP Services > EEM Interface Events**.

R6:

```
!  
! Create an interface event detector based applet.  
! notice that the command "event interface..." should be entered  
! along on a single line - it defines a detector condition  
!  
event manager applet INTERFACE_MONITOR  
  event interface name Serial 0/0/0 parameter txload entry-op gt entry-  
va 204 entry-type value poll-interval 1  
  action 1.0 syslog msg "R6's $_interface_name output rate:  
$_interface_parameter = $_interface_value"  
!  
interface Serial 0/0/0  
  load-interval 30
```

Task 7.7 Breakdown

The solution uses interface event monitoring with an EEM applet to track the transmit load value. Since the interface counters are in fact N-minute average values of the actual rate, we need to set the interface **load-interval** as low as possible to ensure quick response to traffic changes.

The script is set to monitor the “txload” variable with the “entry value” of 204 and entry operation “gt” or “greater”. This means that the entry event will be triggered once the “txload” exceeds “204”, which corresponds to 80% of transmit load as the maximum value to “txload” is “255”. We set the poll interval to one second which allows the script to check the interface for changes every second. The task does not provide any specific value, so it’s up to you to select any value that is reasonably low.

Deep Dive

- o What other way can you signal an interface congestin condition?

Task 7.7 Verification

Generate a flood of ICMP packets across the Frame-Relay interface using the command:

```
Rack1R6#ping 54.1.1.254 size 1000 repeat 1000000 timeout 0
```

Observe syslog messages - notice that you may need to set R6's Frame-Relay interface bandwidth to a low value e.g. 64 using the command **bandwidth 64**:

```
%HA_EM-6-LOG: INTERFACE_MONITOR: R6's Serial0/0/0 output rate: txload = 231
```

You may also validate the registered policies with EEM and the published events:

```
Rack1R6#show event manager policy registered
```

```
No.  Class      Type      Event Type      Trap  Time Registered
Name
1    applet     user      interface        Off   Sun Dec 6 20:02:35
2009  INTERFACE_MONITOR
    name {Serial0/0/0} parameter {txload} entry_op gt entry_val 204
    entry_type value exit_op lt exit_val 102 exit_type value poll_interval
    30.000
    maxrun 20.000
    action 1.0 syslog msg "R6's $_interface_name output rate:
    $_interface_parameter = $_interface_value
```

```
Rack1R6#show event manager history events
```

```
No.  Job Id Proc Status  Time of Event      Event Type
Name
1    1      Actv success  Sun Dec 6 20:08:35 2009  interface
applet: INTERFACE_MONITOR
```

Task 8.1 Solution

Before You Begin

This scenario utilizes the Frame-Relay Traffic Shaping feature. If you want to get familiar with this technology, check out VOL1's QoS scenario named **Legacy Frame-Relay Traffic Shaping**,

R5:

```
map-class frame-relay DLCI_504
  frame-relay cir 512000
  frame-relay bc 25600
  frame-relay be 51200
  frame-relay mincir 384000
  frame-relay adaptive-shaping becn
!
map-class frame-relay DLCI_513
  frame-relay cir 128000
  frame-relay bc 6400
  frame-relay be 0
  frame-relay mincir 96000
  frame-relay adaptive-shaping becn
!
interface Serial0/0/0
  frame-relay traffic-shaping
  frame-relay interface-dlci 504
    class DLCI_504
  frame-relay interface-dlci 513
    class DLCI_513
```

Task 8.1 Breakdown

For Frame-Relay traffic-shaping we need to find out the Bc, Be, and CIR values for every PVC involved in the scenario. Those three core values should be discovered in the task wording. The only router performing shaping in this scenario is R5, so we have to deal with just one router.

First, we can see that the Tc is 50ms so we can find the CIR if we figure out the Bc and Be values. Next, the port's physical rate is 1536K which defines the maximum possible rate that PVC can send at. The Bc values can be found from the information in the task:

- 1) DLCI 504, CIR = 512Kbps, Tc=50ms, Bc = $512000 * 50 / 1000 = 25600$ bits.
- 2) DLCI 513, CIR = 128Kbps, Tc=50ms, Bc = $128000 * 50 / 1000 = 6400$ bits

The task wording says we should use adaptive shaping for both PVCs and throttle down the sending rate in response to BECNs, using the MinCIR values of 384Kbps for DLCI 504 and 96Kbps for DLCI 513.

This leaves us with only one value to find out – the Be for every PVC. The task explicitly says that DLCI 513 is not allowed to send above CIR, so we set Be to 0 for DLCI 513. The task further says that DLCI 504 is allowed to send up to the physical rate if it accumulates enough credit. What this means is that $(Bc + Be) / Tc = AIR$ (physical interface rate) = 1536Kbps. The equation expands to:

$(25600 + Be) = 1536000 * 50 / 1000$ and therefore $Be = 51200$. Now we have the values defined for all DLCIs:

- 1) DLCI 504: CIR = 512000, Bc = 25600, Be = 51200, MinCIR = 384000
- 2) DLCI 513: CIR = 128000, Bc = 6400, Be = 0, MinCIR = 96000

We use those parameters to define the map classes. It is possible to use either MQC, CBTS, or FRTS for traffic-shaping in this task, but we chose the legacy FRTS as the simplest method, which involves less configuration efforts. Notice that in the real exam your choice may be based on the other tasks and additional requirements.

Further Learning

There are four general ways to do frame-relay traffic shaping: legacy Generic Traffic Shaping, legacy Frame-Relay Traffic Shaping, MQC-based Frame-Relay traffic shaping and MQC-based Class-Based Traffic Shaping. Among these, the legacy FRTS used in this task is still probably one of the most widely used. For a detailed description of the differences between those flavors, you may want to work through the respective `VOL1 QoS` section scenarios:

`Legacy GTS for Frame-Relay,`
`Legacy Frame-Relay Traffic Shaping,`
`Legacy Adaptive FRTS,`
`MQC Based Frame-Relay Traffic Shaping`
`Using Class-Based GTS for FRTS.`

Deep Dive

- What is the main feature of Frame-Relay Traffic Shaping?
- Why would you need FRTS?
- What feature of Frame-Relay are service providers using in the core to enforce FR QoS policies?

Task 8.1 Verification

Check the FRTS configuration:

Rack1R5#show traffic-shape

```
Interface   Se0/0/0
           Access Target Byte  Sustain  Excess   Interval  Increment  Adapt
VC          List   Rate  Limit bits/int bits/int (ms)      (bytes)    Active
502         56000  875   7000    0        125      875       -
503         56000  875   7000    0        125      875       -
504         512000 9600  25600   51200    50       3200      BECN
513         128000 800   6400    0        50       800       BECN
501         56000  875   7000    0        125      875       -
```

Double-check for more detailed information:

Rack1R5#show frame-relay pvc 504

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

DLCI = 504, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0

<output omitted>

Shaping adapts to BECN

pvc create time 05:50:23, last time pvc status changed 01:50:51

cir 512000 bc 25600 be 51200 byte limit 9600 interval 50

mincir 384000 byte increment 3200 Adaptive Shaping BECN

<output omitted>

Note Be is set to 0, to disable bursting:

Rack1R5#show frame-relay pvc 513

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

DLCI = 513, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0

<output omitted>

Shaping adapts to BECN

pvc create time 05:50:56, last time pvc status changed 04:16:14

cir 128000 bc 6400 be 0 byte limit 800 interval 50

mincir 96000 byte increment 800 Adaptive Shaping BECN

<output omitted>

Task 8.2 Solution

Before You Begin

The solution is based on the MQC three-color policer feature. This technology is explained in-depth in the **VOL1's QoS** scenario named **MQC Single-Rate Three-Color Policer**, which you may use to refresh your knowledge.

```
R1:
!
! Match ICMP traffic
!
class-map match-all ICMP
  match protocol icmp
!
! Police the traffic using Bc of 1/4*128000/8 bytes
!
policy-map POLICE_ICMP
  class ICMP
    police cir 128000 bc 4000
!
interface FastEthernet0/0
  service-policy output POLICE_ICMP
```

Task 8.2 Breakdown

The task does not mention the specific rate-limiting method explicitly so we choose the MQC policer as a more flexible option with respect to configuration. The task explicitly requires using the burst value of 1/4 of the sending rate. What this means is that the Bc should be calculated as $1/4s*128000=32000$ bits. However, the MQC policer uses the burst value in bytes, and so the final burst value is 4000 bytes. In real life scenarios, where you use policing for the purpose of rate-limiting, your choice of the burst size should be based on empirical observations.

Further Learning

If you need more information on the fundamentals of traffic policing and rate-limiting, check out the blog post named **The meaning of Bc with Traffic Policing**

<http://blog.ine.com/2009/12/12/the-meaning-of-bc-with-traffic-policing/>

on INE's blog. Additionally, the following VOL1 QoS scenarios provide in-depth breakdown and hands-on examples for various applications of rate-limiting:

Legacy CAR for Admission Control,
Oversubscription with Legacy CAR and WFQ,
Frame-Relay Traffic-Policing and Congestion Management,
MQC Single-Rate Three-Color Policer.

These scenarios provide enough information to fully comprehend the use of traffic policing and rate-limiting in almost any practical scenario.

Task 8.2 Verification

Check policing parameters:

```
Rack1R1#show policy-map interface fastEthernet 0/0
FastEthernet0/0
```

```
Service-policy output: POLICE_ICMP
```

```
Class-map: ICMP (match-all)
  0 packets, 0 bytes
  5 minute offered rate 0 bps, drop rate 0 bps
Match: protocol icmp
police:
  cir 128000 bps, bc 4000 bytes
  conformed 0 packets, 0 bytes; actions:
    transmit
  exceeded 0 packets, 0 bytes; actions:
    drop
  conformed 0 bps, exceed 0 bps
```

Task 8.3 Solution

Before You Begin

This solution uses CBWFQ settings. CBWFQ technology might be confusing due to the lack of good explanations available in Cisco documentation. Furthermore, starting with IOS 12.4(20)T, the familiar CBWFQ has been replaced with HQF. For now, if you are looking toward better understanding of CBWFQ features, you may want to check out the post named “Insights on CBWFQ”

<http://blog.ine.com/2008/08/17/insights-on-cbwfq/>

on INE’s blog. We also recommend completing the following scenario from VOL1’s QoS section: **MQC Bandwidth Reservation and CBWFQ** prior to attempting this scenario.

```
R5:
!
! Use NBAR for protocol matching.
!
class-map match-all CITRIX
  match protocol citrix
!
! Match VoIP traffic based on pre-defined DSCP value
!
class-map match-all VOICE
  match dscp ef
!
! Define CBWFQ scheduling parameters
!
policy-map CBWFQ
  class VOICE
    priority 64
  class CITRIX
    bandwidth remaining percent 30
    queue-limit 16
  class class-default
    fair-queue
!
! Assign CBWFQ as the queueing policy for PVC
!
map-class frame-relay DLCI_504
  service-policy output CBWFQ
!
map-class frame-relay DLCI_513
  service-policy output CBWFQ
```

Task 8.3 Breakdown

The task clearly points toward using CBWFQ for traffic queueing and scheduling. There is an explicit value set for VoIP traffic which we use in the policy. We do not define any custom burst value as the task does not require that. Further, we need to classify Citrix traffic. Since this is an application that uses dynamic port numbers, the best option here is to use NBAR classification. The task mentions that Citrix traffic should be guaranteed 30 percents of the bandwidth remaining after voice traffic. This is accomplished using the command **bandwidth remaining percent** in the policy-map configuration. Finally, the class-default traffic uses simple flow-based scheduling, which utilizes precedence-weighted scheduling for WFQ and per-flow scheduling for HQF.

Deep Dive

- Name the main differences between CBWFQ and HQF.
- Why might WFQ queues starve with the default CBWFQ settings?

Task 8.3 Verification

Check the PVC parameters and the associated policy map:

Rack1R5#show frame-relay pvc 504

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

DLCI = 504, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE = Serial0/0/0

```

input pkts 6                output pkts 3                in bytes 204
out bytes 102              dropped pkts 0              in pkts dropped 0
out pkts dropped 0        out bytes dropped 0
in FECN pkts 0           in BECN pkts 0            out FECN pkts 0
out BECN pkts 0          in DE pkts 0              out DE pkts 0
out bcast pkts 3         out bcast bytes 102
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
Shaping adapts to BECN
pvc create time 01:01:30, last time pvc status changed 01:01:10
cir 512000   bc 25600   be 51200   byte limit 9600   interval 50
mincir 384000   byte increment 3200 Adaptive Shaping BECN
pkts 0        bytes 0          pkts delayed 0          bytes delayed 0
shaping inactive
traffic shaping drops 0
service policy CBWFQ
Serial0/0/0: DLCI 504 -

```

Service-policy output: CBWFQ

```

Class-map: VOICE (match-all)
 0 packets, 0 bytes
 5 minute offered rate 0 bps, drop rate 0 bps
Match: dscp ef (46)
Queueing
  Strict Priority
  Output Queue: Conversation 40
  Bandwidth 64 (kbps) Burst 1600 (Bytes)
  (pkts matched/bytes matched) 0/0
  (total drops/bytes drops) 0/0

Class-map: CITRIX (match-all)
 0 packets, 0 bytes
 5 minute offered rate 0 bps, drop rate 0 bps
Match: protocol citrix
Queueing
  Output Queue: Conversation 41
  Bandwidth remaining 30 (%)Max Threshold 16 (packets)
  (pkts matched/bytes matched) 0/0
  (depth/total drops/no-buffer drops) 0/0/0

Class-map: class-default (match-any)
 0 packets, 0 bytes
 5 minute offered rate 0 bps, drop rate 0 bps
Match: any
Queueing
  Flow Based Fair Queueing
  Maximum Number of Hashed Queues 32
  (total queued/total drops/no-buffer drops) 0/0/0
Output queue size 0/max total 600/drops 0

```

Task 8.4 Solution

Before You Begin

This scenario deals with Catalyst QoS features. The hardware-based nature of Catalyst QoS seriously affects the feature design, which may vary from platform to platform significantly. The current CCIE R&S exam only tests students on knowledge of the 3560/3750 QoS features. Policing and marking are two fundamental QoS stages, and if you feel like you need more information check out the blog post named “Comparing Traffic Policing Features in the 3550 and 3560 switches”

<http://blog.ine.com/2008/09/11/comparing-traffic-policing-features-in-the-3550-and-3560-switches/>

This post outlines the difference between the two platforms and provides in-depth configuration examples. Even though the 3550 has been removed from the lab exam, comparing the two platforms is a good way to understand the QoS features in general. The same approach is used in the VOL1 QoS section, where you may want to check the scenarios named:

**Catalyst QoS Port-Based Classification,
Catalyst 3560 Per-VLAN Classification**

SW4:

```
mls qos
mls qos map dscp-mutation OSPF_CHANGE 48 to 24
!
! Mutate the DSCP value on received OSPF packets
!
interface FastEthernet 0/4
 mls qos dscp-mutation OSPF_CHANGE
 mls qos trust dscp
```

```
SW2:
!
! Enable MLS-Based QoS on all trunk links (to SW1/SW3)
!
mls qos
!
interface FastEthernet 0/13
  mls qos vlan-based
!
interface FastEthernet 0/15
  mls qos vlan-based
!
interface FastEthernet 0/16
  mls qos vlan-based
!
interface FastEthernet 0/17
  mls qos vlan-based
!
interface FastEthernet 0/18
  mls qos vlan-based

!
! Classify HTTP traffic
!
ip access-list extended HTTP
  permit tcp any eq 80 any
!
class-map HTTP
  match access-group name HTTP
!
! Mark HTTP traffic with DSCP 16
!
policy-map VLAN46_MARK
  class HTTP
    set dscp 16
  class class-default
    trust dscp
!
interface vlan 46
  service-policy input VLAN46_MARK
```

```
!  
! Map HTTP traffic (DSCP 16=CS2) to queue 4  
!  
!  
mls qos srr-queue output dscp-map queue 4 16  
  
!  
! Set interface speed limit to 4 Mbps (40%*10Mps)  
! Shape queue 4 to 1/10 of the interface speed  
!  
interface FastEthernet 0/6  
  srr-queue bandwidth limit 40  
  speed 10  
  srr-queue bandwidth shape 0 0 0 10
```

Task 8.4 Breakdown

The task calls for the use of Catalyst QoS, by directly stating that no routers should be configured. These are the main goals:

- 1) Remark OSPF packets sent by R4
- 2) Limit the rate of packets sent to R6's VLAN 46 connection to 4Mbps
- 3) Limit the HTTP packet rate to 1Mbps
- 4) Mark the HTTP packets sent to R6 with CS2

The first requirement can be achieved using DSCP mutation maps. The simplest way is to place the DSCP mutation on the port connecting R4 to VLAN46. We accomplish this on SW4 as follows:

```
mls qos
mls qos map dscp-mutation OSPF_CHANGE 48 to 24
!
interface FastEthernet 0/4
 mls qos dscp-mutation OSPF_CHANGE
 mls qos trust dscp
```

The other requirements are interconnected.

First, we need to classify the HTTP packets sent to R6 over VLAN 46 and mark them with DSCP CS2. Since R6's VLAN 46 interface connects to SW2, this is the switch to configure. Marking is performed ingress on the catalyst switches and could be either interface-based or VLAN based. For simplicity, we've chosen the VLAN-based model as it allows avoiding replicating the same configuration on all trunk ports. We configure the trunk links for VLAN-based QoS: all links need to be configured. This is because VLAN 46 traffic may take different paths from SW4 to SW2 (e.g. across SW1 or SW3 – refer to the physical connections diagram in the beginning of the lab). After this, we apply the marking policy ingress on VLAN 46, matching HTTP traffic with an access-list and setting the DSCP value of CS2. Notice how the rest of the traffic is matched under the "class-default" and has its marking trusted. This is needed to preserve marking for all other traffic flows:

```
ip access-list extended HTTP
  permit tcp any eq 80 any
!
class-map HTTP
  match access-group name HTTP
!
policy-map VLAN46_MARK
  class HTTP
    set dscp 16
  class class-default
    trust dscp
!
interface vlan 46
  service-policy input VLAN46_MARK
```

The next task is to limit the packet rate to 4Mbps on the connection to R6. There is no egress policing in the 3560 switch and thus we may only use SRR queue shaping features. We configure the port physical speed to 10Mbps as SRR granularity does not allow to set the shaping rate to 4Mbps on 100Mbps links. We then use a SRR bandwidth limit of 40% relative to the port speed which results in the rate limit of 4Mbps:

```
interface FastEthernet 0/6
  srr-queue bandwidth limit 40
  speed 10
```

Following that task, we need to limit the HTTP traffic rate to 1Mbps. This could only be done by mapping this traffic flow to a separate queue and setting the SRR shape limit for this queue. Since HTTP traffic is marked with CS2, we configure the switch to map this DSCP value to queue-id 4 (this could be any queue, but we selected the highest-numbered one). The command to achieve this is **mls qos srr-queue output dscp-map queue 4 16**.

Finally, we only need to set the proper SRR queue shaping weight for Queue 4 on the port connecting to R6. This is done using the interface-level command **srr-queue bandwidth shape 0 0 0 10** which sets the queue 4 bandwidth limit to 1/10 of the interface physical rate of 10Mbps; resulting in a 1Mbps hard-limit.

Task 8.4 Verification

Configure R6 as follows to perform verification:

R6:

```
ip access-list extended HTTP
  permit tcp any eq www any
ip access-list extended OSPF
  permit ospf any any
!
class-map match-all HTTP_DSCP16
  match ip dscp cs2
  match access-group name HTTP
!
class-map match-all OSPF_DSCP24
  match ip dscp cs3
  match access-group name OSPF
!
!
policy-map METER
  class OSPF_DSCP24
  class HTTP_DSCP16
!
interface FastEthernet 0/0
  service-policy input METER
  load-interval 30
```

Now check the policy-map stats for OSPF packet matches:

```
Rack1R6#show policy-map interface fastEthernet 0/0
FastEthernet0/0
```

```
Service-policy input: METER
```

```
Class-map: OSPF_DSCP24 (match-all)
 18 packets, 1684 bytes
 5 minute offered rate 0 bps
Match: ip dscp cs3 (24)
Match: access-group name OSPF
```

```
Class-map: HTTP_DSCP16 (match-all)
 0 packets, 0 bytes
 5 minute offered rate 0 bps
Match: ip dscp cs2 (16)
Match: access-group name HTTP
```

```
Class-map: class-default (match-any)
 32 packets, 2699 bytes
 5 minute offered rate 0 bps, drop rate 0 bps
Match: any
```

Now configure R4 as a web-server and transfer the IOS image in R4 to R6:

R4:

```
ip http server
ip http path flash:
ip http authentication enable
```

```
Rack1R6#copy http://cisco:cisco@150.1.4.4/c1841-adventerprisek9-mz.124-24.T.bi$
```

```
Loading http://*****:*****@150.1.4.4/c1841-adventerprisek9-mz.124-24.T.bin !!!!!!!!
```

```
Rack1R6#sh policy-map interface fastEthernet 0/0
FastEthernet0/0
```

```
Service-policy input: METER
```

```
Class-map: OSPF_DSCP24 (match-all)
  87 packets, 8162 bytes
  30 second offered rate 0 bps
  Match: ip dscp cs3 (24)
  Match: access-group name OSPF
```

```
Class-map: HTTP_DSCP16 (match-all)
  26989 packets, 15906896 bytes
  30 second offered rate 979000 bps
  Match: ip dscp cs2 (16)
  Match: access-group name HTTP
```

```
Class-map: class-default (match-any)
  99 packets, 6871 bytes
  30 second offered rate 0 bps, drop rate 0 bps
  Match: any
```

Notice the HTTP class matches and the traffic rate is close to 1Mbps:

VOL1 Scenario Reference

The following is the list of VOL1 labs that are pre-requisites for this mock lab scenario.

- Bridging and Switching > Private VLANs
- OSPF > OSPF over Non-Broadcast Media
- OSPF > OSPF Interface Timers
- OSPF > OSPF Path Selection with Non-Backbone Transit Areas
- OSPF > Repairing Discontiguous OSPF Areas with Virtual links.
- BGP > BGP Bestpath Selection using Local Preference
- BGP > BGP Bestpath Selection using MED
- IPv6 > EIGRPv6
- IPv6 > IPv6 Tunneling
- IPv6 > IPv6 PIM and MLD
- IPv6 > IPv6 PIM BSR
- MPLS VPN > MPLS LDP
- MPLS VPN > MPLS Label Filtering
- MPLS VPN > MP-BGP VPNv4
- MPLS VPN > MP-BGP Prefix Filtering
- Multicast > RPF Failure
- Multicast > PIM Sparse Mode
- Multicast > PIM Sparse-Dense Mode
- Multicast > Auto-RP
- Multicast > IGMP Filtering
- Security > Traffic Filtering with Standard Access-Lists
- Security > Traffic Filtering with Extended Access-List
- Security > Logging with Access-Lists
- System Management > RMON Alarms
- System Management > NTP
- System Management > NTP Authentication
- IP Services > IP Precedence Accounting
- IP Services > HSRP
- IP Services > NAT Overload
- IP Services > EEM Interface Events
- QoS > Legacy Frame-Relay Traffic Shaping
- QoS > MQC Single-Rate Three-Color Policer
- QoS > MQC Bandwidth Reservation and CBWFQ
- QoS > Catalyst QoS Port-Based Classification
- QoS > Catalyst 3560 Per-VLAN Classification

For additional training, you may practice some or all of these scenarios before or after completing this lab.

Deep Dive Answers

- Why would you avoid VLAN stripping off trunks in real-world scenarios?
 - This prevents flexible path manipulation and may result in STP/VLAN topology inconsistency if using MSTP.
- If required to maximize link bandwidth usage, what alternative to STP may you use to provide redundancy and bandwidth optimization?
 - Port-Channels are an effective way to maximize link utilization and provide redundancy. However, their effectiveness depends on statistical traffic pattern properties, e.g. having enough flows to balance across the member links.
- Why would you prefer BPDU Guard over BPDU Filter at the edge of your network?
 - BPDU Filter would stop STP formation and topology changes, but may result in loop formation if the connected topology is looped. BPDU Guard will shutdown the offending link and send a notification message, requiring operator intervention. This is normally safer in most environments, though less dynamic and flexible.
- Why is it not sufficient to simply raise an OSPF router's priority to ensure it is elected as a DR?
 - Since elections are non-preemptive, the first router to be elected the DR stays that way until it loses connection to the segment. Thus, if a lower priority router is booted before the higher priority, the former will take the role of the DR. Setting priority to zero ensures a router is never elected as DR.
- What are the drawbacks of OSPF non-deterministic DR election?
 - Unpredictable elections sometimes makes troubleshooting harder. Normally it is recommended fixing the DR and BDR roles and assigning other routers zero priorities.
- What is the difference between an OSPF virtual link and a tunnel?
 - Even though OSPF treats virtual links as a point-to-point connection in area 0, it is not used to tunnel packets. The link is used to calculate the SPF however. Forwarding is based on distance-vector computations for summary routes learned over the virtual link.
- Can you peer OSPF routers that are using broadcast and non-broadcast network types respectively?
 - Yes, provided that you have adjusted the timers to match. Both topologies elect a DR and thus are compatible in their vision of the topology.

- What are the large-scale limitations of using OSPF fast-hello timers? What could be an alternative?
 - With many OSPF adjacencies, fast hellos could become a burden to a router's CPU. An alternative would be using BFD which could run on the line-card's CPU and report link/adjacent neighbor changes to upper level protocols.
- What is the main difference of the redistribute function between IPv6 and IPv4 routing protocols?
 - The IPv6 redistribute command does not include local connected routes even if they are part of the IGP advertisements. The same feature among IPv4 IGP processes is only found for IS-IS.
- What is the best-practice for setting AD in IGP routing protocols?
 - Normally you would set an external route's AD higher than ANY native internal distance for other routing protocols.
- How could you fix the lack of external AD in RIP?
 - You may use an access-list and selective AD changes or selective metric adjustments.
- Why are routing loops possible in scenarios with redistribution?
 - The routes learned from other routing protocols are treated in a distance-vector manner, and no topology information is being learned. Similarly, EIGRP cannot send diffusing computations in the other routing domain and determine loop free paths. Therefore, it is possible for the routing domain to prefer its own prefixes via another routing domain. Routing loops could be manually detected by means of tagging and prevented by use of filtering, summarization and AD manipulation.
- What is the purpose of the MED attribute in BGP?
 - The metric attribute is supposed to be the neighboring AS internal cost to reach the advertised prefix. It suggests how close the advertising BGP speaker was to the advertised prefix in the terms of other system's IGP metric. Normally, MED is considered by receiving AS before the local IGP metric to reach the local border BGP peer that learned this prefix.
- How is "cold potato" routing different from "hot potato" routing in BGP?
 - Cold potato routing is based on preferring "foreign" MED attribute values over local IGP metrics to the border peer. This in effect, allows the local system to find the exit point closest to the advertised prefix in terms of "other" AS IGP metrics. The local AS metrics are effectively being ignored. Hot potato, in turn, attempts to deliver the packet for the other AS prefix as fast as possible in terms of local IGP metric. This is achieved by resetting the MED attribute in received prefixes to the same value.

- Why does comparing MED's between multiple Autonomous Systems not work?
 - MED effectively reflects the other AS's IGP metric value. If two different ASs use different IGPs, comparing their MED values makes little sense.
- Why does IPv6 PIM use a special tunnel interface built for every RP?
 - This tunnel interface is used to send initial multicast packets to the RP when registering the multicast source. This procedure makes the multicast forwarding code unified. Instead of encapsulation, the original multicast PIM register message is sent over the tunnel with proper multicast state. The tunnel traffic is later pruned after the SPT to the source is built by the RP.
- What version of IGMP could be thought of as equivalent to MLD?
 - IGMPv2 was the IPv4 equivalent to MLD. Notice that it does not support source specific multicast.
- What label distribution modes do you know of?
 - LDP label distribution modes are: downstream on-demand and downstream unsolicited. The latter is the prevalent mode used in IP networks for LDP. The first mode only allocates label upon the request of the upstream neighbor (with respect to the prefix in question) while the second mode simply advertises labels for all locally known prefixes.
- Why is a /32 prefix required for PE Loopback interfaces?
 - The use of a /32 prefix allows the local router to advertise a NULL label for the prefix. Any non-host route will generate a special label that the local router will use to perform an aggregate lookup based on the destination IP address in the packet. With the NULL label advertisement, the label lookup will be performed based on the VPNv4 label, in the case of MPLS VPN scenario.
- How does LDP validate label advertisement against the local RIB?
 - Every prefix learned via LDP should have an exact match in the local RIB. Otherwise, the advertisement is ignored. This prevents prefix summarization in OSPF networks.
- What are the benefits and drawbacks of using MP-BGP over running native IGP adjacencies across MPLS tunnels?
 - The main benefit is practically unlimited scalability and extendibility of the carrier protocol. The main drawback is slow convergence of BGP, which affect MPLS VPN customers.
- How are MP-BGP labels used together with IGP labels to create the MPLS label stack?

- The next-hop found in an MP-BGP update is looked up in the RIB, and the corresponding IGP (normally LDP-learned) label is found. Next, the VPNv4 label is stacked on top of the label associated with the MP-BGP next-hop and the label stack is ready.
- Why was Auto-RP used with PIMv1?
 - There was no mechanism to distribute the RP information. Using special dense-mode groups was a natural and simple way for out-of-band signaling in this situation. The use of a Mapping Agent allows for consistent RP selection among all multicast routers.
- Why do you need to enable PIM on the Loopback interfaces for Auto-RP?
 - Auto-RP sends multicast packets out of the interface selected as RP or MA candidate. In order for the multicast packet to be sent, any form of PIM should be enabled on the interface. In the case of Loopback, the multicast packet returns to the router itself and is flooded out of all other multicast-enabled interfaces.
- How are mroutes different from static routes?
 - Multicast routes specify manual RPF check information of a given prefix, not the next-hop to route traffic to.
- Do you need to have PIM enabled on the interface using the IGMP join command?
 - Yes, to enable multicast switching out this interface.
- What is the common source of RPF problems in the lab?
 - Redistribution is the most common problem, due to incompatible metrics and suboptimal routing paths.
- What is the purpose of a TCP SYN attack?
 - To exhaust the TCP connection table on the target host.
- Is there a host-based solution to protect against TCP SYN flooding?
 - Known as SYN cookies, it uses TCP sequence number to encode the connection parameters. This allows for dropping the connection entries for newly received SYN packets and later recovering them if TCP ACK is being received based on the sequence number. The drawback is lack of supported MSS and TCP options.
- How could you distinguish a TCP SYN attacker from a normal session?
 - Normally, there is no clear criterion. Most of the time, if a server does not receive TCP ACK segment within a defined time window the connection is suspected to be a part of the attack. However, sometimes, these delays could be a result of network congestion or propagation delays.

- How could outside systems spoofing your addresses be dangerous for your network?
 - If your devices are using access-control lists based on source IP addresses, they could be bypassed by spoofed packets. Furthermore, some type of network attacks such as SMURF use spoofed source IP addresses to expose the spoofed victims to a springboard (bounce) attack.
- What is the main problem with ingress filtering based on access-lists?
 - It is not flexible, and you may have problems correctly defining all address ranges belonging to your network, especially if you are merging two different networks. An alternate solution could be using uRPF – unicast Reverse Path Failure check for automatic spoofing prevention.
- How could TTL filtering be used to prevent spoofing?
 - Since most IP implementations set TTL of 255 in outgoing packets, you may use the TTL in received packet to figure out the distance to the sender. If a packet sourced off your local address appears to be 30 hops away this most likely signified a spoofed connection.
- What is the benefit of using RMON over SNMP polling?
 - RMON runs independently in the device and may collect statistics even if the device is temporarily unavailable. Furthermore, it puts less stress on the NMS. However, the main benefit of NMS-based polling is centralized control and reporting.
- What is the main purpose of “delta” sampling for RMON?
 - Delta sampling is used to determine the rate (speed) of changes. For example, delta-sampling the number of input bytes collected for an interface will reveal the input traffic rate.
- Why are key IDs important with NTP authentication?
 - NTP packets bear key IDs in their headers to allow the receiving device picking up the right key for packet authentication.
- When do you need to trust an NTP authentication key?
 - When the received packet is used to change the value of local clock, e.g. when a client receives a server update.
- When accounting for IP packets, why is setting threshold so important?
 - Memory consumption may grow in unlimited fashion, since the number of sources and destination could be huge, especially if there is a virus program spoofing IP addressing. Therefore, defining a limit is very important.

- Give an example of using IP precedence accounting.
 - IP precedence account could be used to estimate the use of existing QoS marking policing, e.g. determine how many packets are getting remarked.
- What HSRP messages do you know?
 - The three main messages are Hello, Coup and Resign.
- What is the purpose of the HSRP Coup message?
 - HSRP Coup message is used by a router that detects itself having the highest priority among others, but not being active. The Coup message tells the currently active router to resign its role.
- What are extendable NAT entries and how are they different from standard?
 - Extendable entries store port number and protocol information in addition to the original/translated IP address. This allows for the same inside host to have multiple outside addresses (e.g. multihoming) or allows mapping different ports of the same global IP to different inside IPs.
- What other ways can you use to signal interface congestion condition with EEM?
 - You may use RMON monitoring and generate an alarm upon crossing the threshold.
- What is the main feature of Frame-Relay Traffic Shaping?
 - Per-VC adaptive shaping. Shaping applies per logical circuit and allows for changing the shaping rate based on reception of explicit congestion notifications.
- Why would you need FRTS?
 - FRTS is normally used to conform to the service-provider's traffic contract. Aside from that, it could be used by the hub site to avoid oversubscribing slow spoke sites.
- What Frame-Relay feature do SPs use in the core to enforce FR QoS policies?
 - Frame-Relay Traffic Policing: metering ingress traffic and remarking it with DE bit or dropping per the traffic contract. The core switches further use different queue thresholds for non-DE and DE-marked packet as well as insert congestion notification upon queue fill-up.
- Name the main differences between CBWFQ and HQF.
 - CBWFQ is based on WFQ scheduling, and HQF uses the more optimized proprietary fair-queueing algorithm. Unclassified traffic always has some bandwidth allocated, unlike it was in CBWFQ. Lastly, fair-queueing is now purely flow-based, ignoring the precedence, and could be used under user-defined classes.

- Why might WFQ queues starve with the default CBWFQ settings?
 - WFQ dynamic flows have less preferred weight values compared to any user-defined class in CBWFQ, and thus may starve in the presence of user-defined classes.

