Copyright Information

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 Lab Workbook VOL2

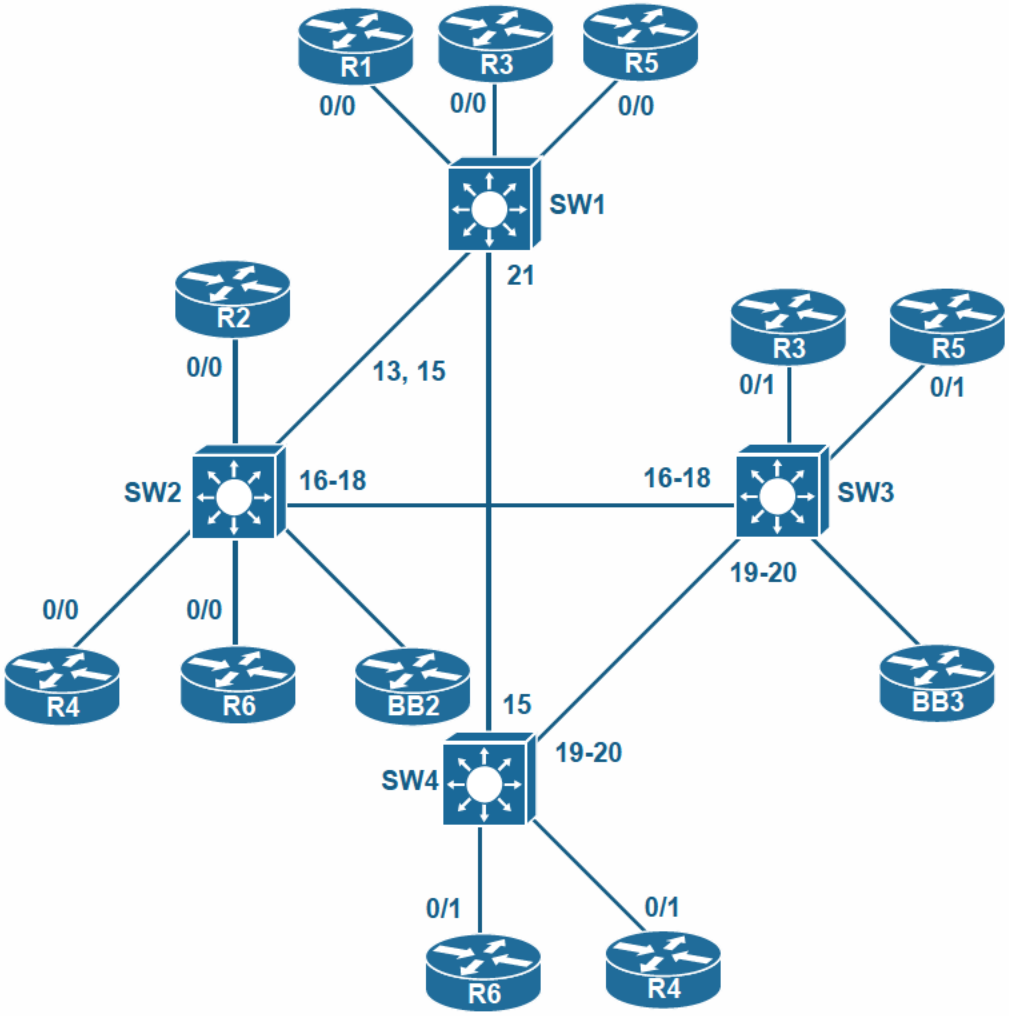
INE's CCIE Routing & Switching Lab Workbook Volume II is designed to be used as a supplement to other self-paced and instructor-led training materials in preparation for Cisco Systems' CCIE Routing & Switching Lab Exam. Please, refer to the company website http://www.INE.com for more information on additional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

VOL2 workbook 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 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.

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 you 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 straight-forward 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 discontiguous 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 Discontiguous 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_TO_OSPF subnets
!
ip prefix-list VLAN_6 permit 183.1.6.0/24
!
route-map CONNECTED_TO_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 in 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 "Undestanding 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_TO_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_TO_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 that 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

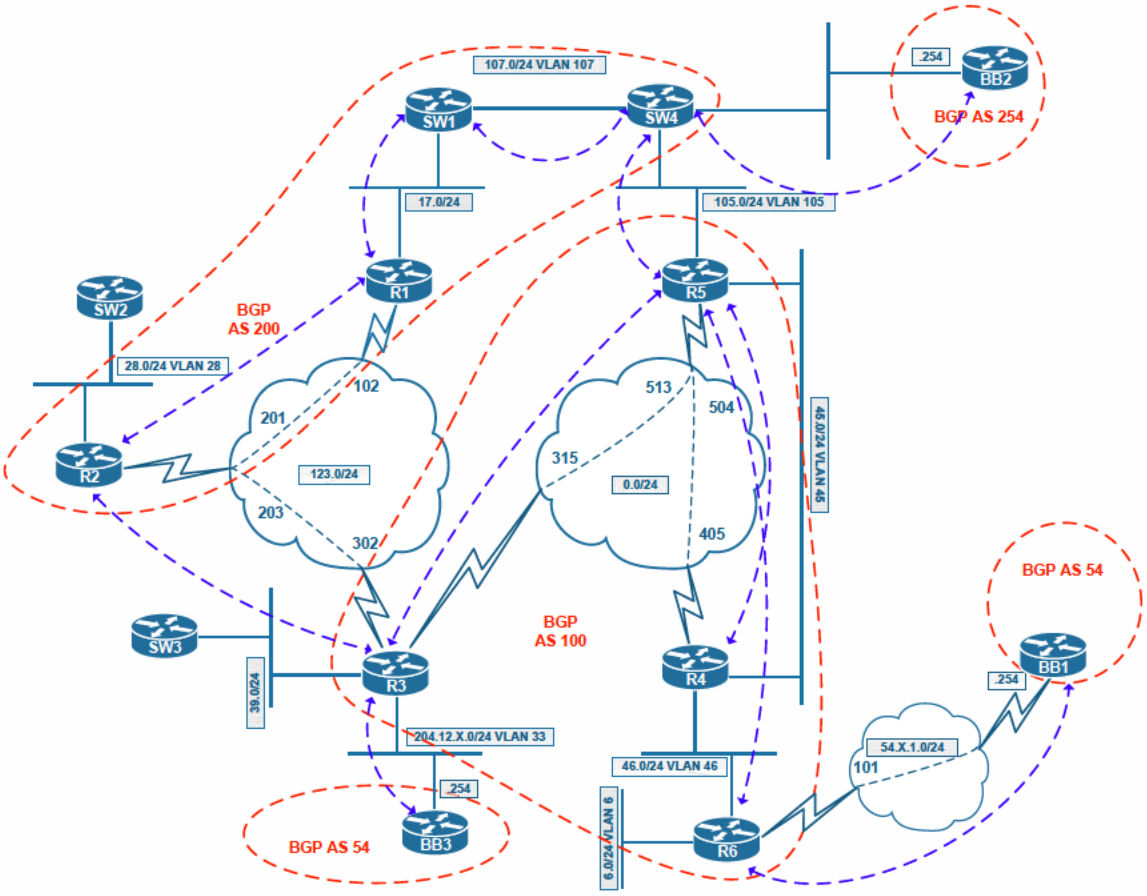
Older Catalyst IOS versions do not support TCL scripting. A smartport macro can be used in place of the TCL shell for ping tests on the switches as follows.

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

For inbound updates:

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

For outbound updates:

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

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

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  Outgoing      Prefix           Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id     Switched      interface
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  Outgoing      Prefix           Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id     Switched      interface
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  Outgoing      Prefix           Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id     Switched      interface
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 "NLRIs" 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 benfits 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 vpnv4 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 vpnv4 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/   Nbr    Query  DR     DR
                                          Mode   Count  Intvl  Prior
Rack1R1#

Rack1R2#show ip pim interface

Address          Interface                Ver/   Nbr    Query  DR     DR
                                          Mode   Count  Intvl  Prior
183.1.28.2       FastEthernet0/0          v2/SD  0      30     1      183.1.28.2
183.1.123.2      Serial0/0                v2/SD  1      30     1
183.1.123.3

Rack1R2#show ip pim neighbor
PIM Neighbor Table
Mode: B - Bidir Capable, DR - Designated Router, N - Default DR Priority,
      S - State Refresh Capable
Neighbor          Interface                Uptime/Expires    Ver   DR
Address                                                            Prio/Mode
183.1.123.3       Serial0/0                05:26:02/00:01:20 v2    1 / DR S
```

R2 connects to R3:

```
Rack1R3#show ip pim interface

Address          Interface                Ver/   Nbr    Query  DR     DR
                                          Mode   Count  Intvl  Prior
204.12.1.3       FastEthernet0/0          v2/SD  0      30     1      204.12.1.3
183.1.123.3      Serial1/0                v2/SD  1      30     1
183.1.123.3
183.1.0.3        Serial1/1                v2/SD  2      30     1
183.1.123.2
```

```
Rack1R3#show ip pim neighbor
PIM Neighbor Table
Mode: B - Bidir Capable, DR - Designated Router, N - Default DR Priority,
      S - State Refresh Capable
Neighbor          Interface                Uptime/Expires    Ver   DR
Address                                                            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/   Nbr    Query  DR     DR
                                          Mode   Count  Intvl  Prior
```

```
Rack1R5#show ip pim interface

Address          Interface                Ver/   Nbr    Query  DR     DR
                                          Mode   Count  Intvl  Prior
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          Interface                Uptime/Expires    Ver   DR
Address                                                            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/   Nbr    Query  DR     DR
                                          Mode   Count  Intvl  Prior
```

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 infromation. 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 outsite 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 acess-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 unreachables
 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 unreachables 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 feauture. 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)
rec 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
filterror =     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
filterror =     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
filterror =     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 realeases, 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
filterror =     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
filterror =     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
filterror =     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
filterror =     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
filterror =     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 basic 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 oftern 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 Solution

🕮 Before You Begin

There are numerous ways that you can use NAT in Cisco IOS. This particular scenario uses a basic configuration (PAT overloading) which is outlined in **VOL1's** scenario **IP Services > NAT Overload**.

R3:

```
!
! Standard ACL to match the pod's traffic based on source IPs
!
access-list 2 permit 183.1.0.0 0.0.255.255
!
! NAT translation rule
!
ip nat inside source list 2 interface FastEthernet0/0 overload
!
! Inside and outside interfaces defined
!
interface FastEthernet0/0
 ip nat outside
!
interface FastEthernet0/1
 ip nat inside
!
interface Serial1/0
 ip nat inside
!
interface Serial1/1
 ip nat inside
```


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 technolgy, 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

- 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 you 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 =
Serial0/0/0
  <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 100Mpbs 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 lets 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 if 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.

Lab 2 Solutions

Task 1.1 Solutions

🕮 Before You Begin

Completing this scenario requires preliminary knowledge of port-channel technology, specifically LACP. You may want to practice VOL1 scenario named **`Bridging and Switching > Layer 2 Etherchannel with LACP`** and read the Wikipedia post article named "Link Aggregation" for introductory material.

http://en.wikipedia.org/wiki/Link_Aggregation

SW1:

```
!
! Make SW1's priority higher to allow it control bundle creation
!
lacp system-priority 1
!
! Ports 19 through 21 connect to SW4
!
interface FastEthernet0/19
 no shutdown
 switchport mode dynamic desirable
 channel-group 3 mode active
!
interface FastEthernet0/20
 no shutdown
 switchport mode dynamic desirable
 channel-group 3 mode active
!
interface FastEthernet0/21
 no shutdown
 switchport mode dynamic desirable
 channel-group 3 mode active
!
interface Port-channel3
 switchport mode dynamic desirable
```



```
SW4:
!
! Port 13 through 15 connect to SW1
!
interface FastEthernet0/13
 no shutdown
 switchport mode dynamic desirable
 channel-group 3 mode passive
!
interface FastEthernet0/14
 no shutdown
 switchport mode dynamic desirable
 channel-group 3 mode passive
!
interface FastEthernet0/15
 no shutdown
 switchport mode dynamic desirable
 channel-group 3 mode passive
!
interface Port-channel3
 switchport mode dynamic desirable
```

Task 1.1 Breakdown

The task requires the use of standard link bundling protocol, which is IEEE LACP. This protocol differs from PaGP in the way that it supports dynamic link aggregation. LACP supports up to 16 links in the bundle with max 8 active. The links are selected for the bundle based on their priority and the maximum allowed number of links. Every link has a priority value and a port number assigned – the numerically lowest value priority:portnumber is more likely to be elected for the bundle. The links that are not elected as part of the bundle are only eligible to join the bundle if any of the primary links fail. The bundling decision based on port priority is made by the switch with lowest LACP system-id which is formed by concatenation LACP priority with switch's MAC address. Thus, we configure SW1 with lower LACP system priority to make it the "deciding" device.

🕮 Further Learning

- How STP interacts with Layer 2 Ether-Channels?
- What happens if one end of the eherchannel unbinds while other remains bonded?

Task 1.1 Verification

Check the port-channel status:

```
Rack1SW1#show etherchannel 3 summary
<output omitted>
Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
3      Po3(SU)         LACP      Fa0/19(P)   Fa0/20(P)   Fa0/21(P)

Rack1SW4#show etherchannel 3 summary
<output omitted>
Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
3      Po3(SU)         LACP      Fa0/13(P)   Fa0/14(P)   Fa0/15(P)
```

Verify the trunk:

```
Rack1SW1#show interface po3 trunk

Port        Mode             Encapsulation  Status        Native vlan
Po3         desirable        n-isl          trunking      1

Port        Vlans allowed on trunk
Po3         1-4094

Port        Vlans allowed and active in management domain
Po3         1,3,5-6,8,10,26,33,52,255,783

Port        Vlans in spanning tree forwarding state and not pruned
Po3         none

Rack1SW4#show interface po3 trunk

Port        Mode             Encapsulation  Status        Native vlan
Po3         desirable        n-isl          trunking      1

Port        Vlans allowed on trunk
Po3         1-4094

Port        Vlans allowed and active in management domain
Po3         1,3,5-6,8,10,26,33,52,255,783

Port        Vlans in spanning tree forwarding state and not pruned
Po3         1,3,5-6,8,10,26,33,52,255,783
```


Verify the dot1q LACP priority:

```
Rack1SW1#show lacp sys-id
1, 001e.bdaa.ba80
Rack1SW1#

Rack1SW4#show lacp sys-id
32768, 000f.f769.3a80
```


Task 1.2 Solution

🕮 Before You Begin

802.1x authentiacation should not be a large topic within the scope of CCIE R&S exam, mostly because you are not involved in configuring access policies in the RADIUS server. To completely understand this scenario, you need to have basic understanding of RADIUS authentication and authorization process. You may find introductory information to RADIUS in the following Wikipedia article:

http://en.wikipedia.org/wiki/RADIUS

And practice the following AAA scenarios from VOL1:

```
Security > AAA Authentication Lists

Security > AAA Exec Authorization

Bridging and Switching > PPP AAA Authentication
```

While not directly related to 802.1x they give you some solid backbround in using AAA and RADIUS. As for 802.1x, for brief introduction we recommend reading the following free sample “IEEE 802.1X: Practical Port Control for Switches” from the Cisco Press book “Cisco Field Manual: Catalyst Switch Configuration”:

http://www.ciscopress.com/articles/article.asp?p=29600

SW1:

```
aaa new-model
!
! The following command ensures no authentication for login process
!
aaa authentication login default none
aaa authentication dot1x default group radius
!
! Enable 802.1x globally
!
dot1x system-auth-control
!
! Enable per-port authentication
!
interface FastEthernet0/9
 switchport mode access
 dot1x port-control auto
!
interface FastEthernet0/10
 switchport mode access
 dot1x port-control auto
!
! RADIUS server settings
!
ip radius source-interface Loopback0
!
radius-server host 204.12.1.100
radius-server key CISCO
```


Task 1.2 Breakdown

In order to provide added security at the access layer of the network, 802.1x defines username and password based authentication for Ethernet switches. To enable 802.1x authentication, first issue the global configuration command **dot1x system-auth-control** (prior to 12.1(14)EA1 this command is not required). Next, enable dot1x must be enabled on a per interface basis by issuing the interface level command **dot1x port-control [mode],** where *mode* is either auto, forced-authorized, or forced-unauthorized. Forced-authorized is the default mode, and indicated that authorization is not required for access into the network. Forced-unauthorized is the opposite, and dictates that clients can never access the network through this port. When the state is set to auto, dot1x is enabled for username and password authentication.

In order to centrally manage users, dot1x integrates with Authentication Authorization and Accounting (AAA) to offload username and password databases to either TACACS or RADIUS. Therefore, to enable dot1x authentication, AAA must be enabled. The first step in enabling AAA is to issue the global command **aaa new-model**. This command starts the AAA process. Next, either the TACACS or RADIUS server should be defined, along with its corresponding key value. This is accomplished with the **radius-server** or **tacacs-server** global configuration command. Additionally, since network devices typically have multiple interfaces running IP, it is common practice to force the router/switch to generate radius or tacacs packets from a single interface instead of relying on what the routing table dictates the outgoing interface to be. This is accomplished with the **ip [tacacs | radius] source-interface** command.

After AAA is enabled, the authentication policy must be defined. This is accomplished by issuing the **aaa authentication dot1x** command. In the above case, the *default* group is used. The default group applies to all interfaces and lines of the device in question.

🕮 Deep Dive

- What is the main idea of EAP protocol used with 802.1X?
- How RADIUS performs authorization if it does not have explicit authorization page?

Task 1.2 Verification

Verify dot1x port control:

```
Rack1SW1#show dot1x all
Sysauthcontrol                Enabled
Dot1x Protocol Version              2
Critical Recovery Delay           100
Critical EAPOL                Disabled

Dot1x Info for FastEthernet0/9
----------------------------------
PAE                         = AUTHENTICATOR
PortControl                 = AUTO
ControlDirection            = Both
HostMode                    = SINGLE_HOST
Violation Mode              = PROTECT
ReAuthentication            = Disabled
QuietPeriod                 = 60
ServerTimeout               = 0
SuppTimeout                 = 30
ReAuthPeriod                = 3600 (Locally configured)
ReAuthMax                   = 2
MaxReq                      = 2
TxPeriod                    = 30
RateLimitPeriod             = 0

Dot1x Info for FastEthernet0/10

----------------------------------
PAE                         = AUTHENTICATOR
PortControl                 = AUTO
ControlDirection            = Both
HostMode                    = SINGLE_HOST
Violation Mode              = PROTECT
ReAuthentication            = Disabled
QuietPeriod                 = 60
ServerTimeout               = 0
SuppTimeout                 = 30
ReAuthPeriod                = 3600 (Locally configured)
ReAuthMax                   = 2
MaxReq                      = 2
TxPeriod                    = 30
RateLimitPeriod             = 0
```


See if RADIUS is configured:

```
Rack1SW1#show aaa servers

RADIUS: id 1, priority 1, host 204.12.1.100, auth-port 1645, acct-port
1646
     State: current UP, duration 991s, previous duration 0s
     Dead: total time 0s, count 0
     Quarantined: No
     Authen: request 0, timeouts 0
             Response: unexpected 0, server error 0, incorrect 0, time
0ms
             Transaction: success 0, failure 0
     Author: request 0, timeouts 0
             Response: unexpected 0, server error 0, incorrect 0, time
0ms
             Transaction: success 0, failure 0
     Account: request 0, timeouts 0
             Response: unexpected 0, server error 0, incorrect 0, time
0ms
             Transaction: success 0, failure 0
     Elapsed time since counters last cleared: 2m
```

Task 1.3 Solution

🕮 Before You Begin

To fully understand this scenario, you may need to go read the following article at Cisco's website "Understanding and Configuring Switching Database Manager on Catalyst 3750 Series Switches"

http://www.cisco.com/en/US/products/hw/switches/ps5023/products_technote09186a00801e7bb9.shtml

SW1 and SW2:

```
!
! You need to reload the switch after this command
! in order for changes to have effect
!
sdm prefer routing
```

Task 1.3 Breakdown

The Switch Database Template (SDM) is used to alter the default allocation of resources (unicast routes, MAC addresses, etc) for the 3550 and 3560 series switches. By default the 3560 will support 8,000 unicast routes (6,000 directly connected and 2,000 non-directly connected). Since the new company's network already has 4,000 routes, the SDM will need to be altered to prefer *routing* to allow SW1 and SW2 to contain over 4,000 non-directly connected routes in their routing tables.

Task 1.3 Verification

Look at the default SDM:

```
Rack1SW1#show sdm prefer | begin unicast routes
  number of IPv4 unicast routes:                    8K
    number of directly-connected IPv4 hosts:        6K
    number of indirect IPv4 routes:                 2K
  number of IPv4 policy based routing aces:         0
  number of IPv4/MAC qos aces:                      0.5K
  number of IPv4/MAC security aces:                 1K
```

After the SDM has been changed to prefer routing and reloaded:

```
Rack1SW1#show sdm prefer | beg unicast route
  number of IPv4 unicast routes:                    11K
    number of directly-connected IPv4 hosts:        3K
    number of indirect IPv4 routes:                 8K
  number of IPv4 policy based routing aces:         0.5K
  number of IPv4/MAC qos aces:                      0.5K
  number of IPv4/MAC security aces:                 1K
```


Task 2.1 Solution

🕮 Before You Begin

You need good understanding of OSPF network types prior to attempting this scenario. Specifically, if you need more basic information, try the following two scenarios from VOL1:

OSPF > Network Point to Multipoint
OSPF > Network Point to Multipoint Non-Broadcast

🖉 Note

Notice that your OSPF adjacency may not form with the initial configurations because of superfluous or incomplete Frame-Relay Inverse-ARP mappings. You may need to reload your routers (R1, R2, R3 and R4) to fix that problem.

R1:

```
!
! Change OSPF network type
!
interface Serial0/0
 ip ospf network point-to-multipoint
!
! Enable OSPF authentication
!
interface FastEthernet0/0
 ip ospf authentication-key CISCO
!
router ospf 1
 area 17 authentication
```

R2:

```
interface Serial0/0
 ip ospf network point-to-multipoint
```


R3:
```
interface Serial1/0
 ip ospf network point-to-multipoint
```
R4:
```
interface Serial0/0/0
 ip ospf network point-to-multipoint
```

SW1:
```
interface FastEthernet0/1
 ip ospf authentication-key CISCO
!
router ospf 1
 area 17 authentication
```


Task 2.1 Breakdown

The task requires no DR election on the segment R2, R3 and R4 – this translates into the point-to-multipoint OSPF network type as this one does not rely on DR. All other requirements are straight-forward. Since there is nothing special about OSPF authentication, the choice has been made to configure it under OSPF process. You may alternatively enable OSPF authentication at interface level, this will not affect grading as it is being done based on the **`show ip ospf interface`** command output.

🕮 Deep Dive

- What is one special benetif of P2MP network type, not available with other OSPF networks?
- From routing perspective, what is the drawback of using P2MP network type?

Task 2.1 Verification

Verify the OSPF neighbors and ensure there is not DR/BDR election. For instance on R1:

```
Rack1R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.2.2         0   FULL/  -        00:01:46    132.1.0.2       Serial0/0
150.1.4.4         0   FULL/  -        00:01:36    132.1.0.4       Serial0/0
150.1.3.3         0   FULL/  -        00:01:31    132.1.0.3       Serial0/0
150.1.7.7         1   FULL/DR         00:00:38    132.1.17.7 FastEthernet0/0
```

Verify the area and network type of the interface:

```
Rack1R1#show ip ospf interface Serial0/0
Serial0/0 is up, line protocol is up
  Internet Address 132.1.0.1/24, Area 0
  Process ID 1, Router ID 150.1.1.1, Network Type POINT_TO_MULTIPOINT,
Cost: 64
  <output omitted>
```

Verify that the OSPF adjacencies in area 17 are being authenticated:

```
Rack1R1#show ip ospf | begin Area 17
    Area 17
        Number of interfaces in this area is 1
        Area has simple password authentication
```

Check if the interface is configured for authentication:

```
Rack1R1#show ip ospf interface fa0/0 | inc auth
  Simple password authentication enabled
```


Task 2.2 Solution

🕮 Before You Begin

You need to understand unicast EIGRP neighbors and EIGRP summarization to complete this task. The required information could be found in the following VOL1 labs:

EIGRP > EIGRP Unicast Updates
EIGRP > EIGRP Summarization

R2:

```
!
! Enable Static EIGRP neighbors
!
router eigrp 10
 network 132.1.26.2 0.0.0.0
 neighbor 132.1.26.6 FastEthernet0/0
```

R6:

```
router eigrp 10
 network 132.1.26.6 0.0.0.0
 neighbor 132.1.26.2 FastEthernet0/0.26
!
! Configure EIGRP summarization
!
interface FastEthernet0/0.26
 ip summary-address eigrp 10 200.0.0.0 255.255.252.0 5
```


Task 2.2 Breakdown

There task states that no other devices on VLAN 26 should be able to hear R2 and R6 exchanging EIGRP packets. This means that both routers should not use EIGRP multicast packets. This translates into using static neighbors over the segment. Notice that unlike with RIP you should not make VLAN26 interface passive in EIGRP. The second part is concerned with EIGRP summarization. The following prefixes are received from BB1:

```
D    200.0.0.0/24 [90/2297856] via 54.1.2.254, 22:40:25, Serial0/0/0
D    200.0.1.0/24 [90/2297856] via 54.1.2.254, 22:40:25, Serial0/0/0
D    200.0.2.0/24 [90/2297856] via 54.1.2.254, 22:40:25, Serial0/0/0
D    200.0.3.0/24 [90/2297856] via 54.1.2.254, 22:40:25, Serial0/0/0
```

The best summary route for those three would be 200.0.0.0/22 which borrows exactly two bits from the third octet.

🕮 Further Learning

You may want to read the post named “A Simple IPv4 Summarization Procedure” at the INE’s blog:

http://blog.ine.com/2010/03/17/a-simple-ipv4-prefix-summarization-procedure/

Which documents a process for IPv4prefix summarization.

🕮 Deep Dive

- Why would you need to use unicast EIGRP neighbors?
- Why summarization is so important in EIGRP?

Task 2.2 Verification

Verify that the EIGRP packets are being sent to the unicast address (protocol 88 is EIGRP):

```
Rack1R2#debug interface fa0/0
Rack1R2#debug ip packet detail
IP: s=132.1.26.6 (FastEthernet0/0), d=132.1.26.2 (FastEthernet0/0), len
60, rcvd 3, proto=88
IP: s=132.1.26.2 (local), d=132.1.26.6 (FastEthernet0/0), len 60,
sending, proto=88
Rack1R2#undebug all
Rack1R2#no debug interface fa0/0
```

This condition could also be verified using show commands

```
Rack1R6#show ip eigrp interfaces detail fastEthernet 0/0.26
IP-EIGRP interfaces for process 10

                        Xmit Queue   Mean   Pacing Time   Multicast
Pending
Interface        Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer
Routes
Fa0/0.26           1        0/0         4       0/1           50
0
  Hello interval is 5 sec
  Next xmit serial <none>
  Un/reliable mcasts: 0/0  Un/reliable ucasts: 3/4
  Mcast exceptions: 0  CR packets: 0  ACKs suppressed: 1
  Retransmissions sent: 0  Out-of-sequence rcvd: 0
  Authentication mode is not set
  Use unicast

Rack1R2#show ip eigrp interfaces detail fastEthernet 0/0
IP-EIGRP interfaces for process 10

                        Xmit Queue   Mean   Pacing Time   Multicast
Pending
Interface        Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer
Routes
Fa0/0              1        0/0         5       0/1           50
0
  Hello interval is 5 sec
  Next xmit serial <none>
  Un/reliable mcasts: 0/0  Un/reliable ucasts: 2/4
  Mcast exceptions: 0  CR packets: 0  ACKs suppressed: 1
  Retransmissions sent: 0  Out-of-sequence rcvd: 0
  Authentication mode is not set
  Use unicast
```


Verify that EIGRP adjacencies have been formed:

```
Rack1R2#show ip eigrp neighbors
IP-EIGRP neighbors for process 10
H   Address         Interface       Hold Uptime   SRTT   RTO  Q  Seq Type
                                    (sec)         (ms)       Cnt Num
1   132.1.26.6      Fa0/0             14 13:42:44    1   200  0  25   S
0   132.1.23.3      Se0/1             14 13:43:08   43   258  0  61
```

Verify that the EIGRP summary is generated on R6:

```
Rack1R6#show ip route | include Null0
D    200.0.0.0/22 is a summary, 00:00:30, Null0
```

Check that the other EIGRP enabled routers see the summary:

```
Rack1R2#show ip route eigrp | include 200.0
D    200.0.0.0/22 [90/2300416] via 132.1.26.6,00:01:38, FastEthernet0/0
```


Task 2.3 Solution

🕮 Before You Begin

There are multiple ways to filter routing updates in RIP. This scenario uses offset-based filering. This procedure is covered in details in **VOL1** scenario named **RIPv2 > RIPv2 Offset List.**

SW1:

```
router rip
 offset-list EVEN_SECOND_OCTET in 16 Vlan783
!
! Care should be taken with the access-list as
! SW1 may filter out SW2's Loopback on rack with
! even rack number
!
ip access-list standard EVEN_SECOND_OCTET
 permit 30.0.0.0 1.254.255.255
```

Task 2.3 Breakdown

The least significant bit of a binary number determines whether the number is even or odd. If the least significant bit is not set the number must be even. If the least significant bit is set the number must be odd. This always holds true since all other places in the binary table are even numbers, and any combination of even numbers plus an odd number results in an odd number. Likewise any combination of even numbers results in an even number.

Place	128	64	32	16	8	4	2	1
Even	X	X	X	X	X	X	X	0
Odd	X	X	X	X	X	X	X	1

Where "X" is either 0 or 1.

Since only the least significant bit determines whether a number is even or odd it is the only bit that needs to be checked. Therefore the resulting wildcard mask is 254, or in binary as follows:

Place	128	64	32	16	8	4	2	1
Wildcard	1	1	1	1	1	1	1	0

Where "0" is check and "1" is ignore.

The most common way to filter off a routing prefix in a distance vector protocol is to use the **`distribute-list`** command. A distribute-list is a way to apply an access-list to routing protocol updates. A routing prefix may also be filtered out by poisoning the metric or distance of the route.

To change the metric of a distance vector prefix use the routing process level command **`offset-list`**. In RIP a metric of 16 is "infinite". When a prefix has a metric of 16 it is considered unreachable, and cannot be installed in the routing table. The first solution to this task adds a metric of 16 to the incoming prefixes, hence invalidating them.

The second solution is to use the distance command. A distance of 255 is infinite. Any prefix with a distance of 255 is considered unreachable, and cannot be installed in the routing table. To change the distance of a prefix use the **`distance [distance] [neighbor] [wildcard] [access-list]`** where *distance* is the desired distance, *neighbor* is the originating address of the prefix, *wildcard* is a wildcard mask used to check the *neighbor* field, and *access-list* is a standard access-list number.

Like mentioned in the configuration part, we need to be careful with the prefix filtering as the distribute list applies to all prefixes received on VLAN783. You may either use extended access-list and specify the route source or just modify the standard access-list used in the solution and make sure it only affects the 30.X.X.X prefixes that BB3 originates.

🕮 Deep Dive

- Why is it important to keep maximum hop count for RIP low?

Task 2.3 Verification

Verify that the RIP networks with an even second octet are being filtered:

```
Rack1SW1#debug ip rip
<output omitted>
RIP: received v2 update from 204.12.1.254 on Vlan783
      30.0.0.0/16 via 0.0.0.0 in 17 hops  (inaccessible)
      30.1.0.0/16 via 0.0.0.0 in 1 hops
      30.2.0.0/16 via 0.0.0.0 in 17 hops  (inaccessible)
      30.3.0.0/16 via 0.0.0.0 in 1 hops
      31.0.0.0/16 via 0.0.0.0 in 17 hops  (inaccessible)
      31.1.0.0/16 via 0.0.0.0 in 1 hops
      31.2.0.0/16 via 0.0.0.0 in 17 hops  (inaccessible)
      31.3.0.0/16

Rack1SW1#undebug ip rip
```

Verify RIP routes in the routing table and make sure the second octets are all odd-valued:

```
Rack1SW1#show ip route rip | i 204.12.1.254
R       31.3.0.0 [120/1] via 204.12.1.254, 00:00:26, Vlan783
R       31.1.0.0 [120/1] via 204.12.1.254, 00:00:26, Vlan783
R       30.3.0.0 [120/1] via 204.12.1.254, 00:00:26, Vlan783
R       30.1.0.0 [120/1] via 204.12.1.254, 00:00:26, Vlan783
```


Task 2.4 Solution

🕮 Before You Begin

Redistribution topology has multiple points of mutual redistribution. This is a complex scenario, which requires good understanding of redistribution process. The breakdown for this scenario provides enough information on the subject, but we recommend reading the following post on the INE blog to get better understanding of redistribution: “Understanding Redistribution Part I, II and III”:

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

http://blog.ine.com/2008/02/19/understanding-redistribution-part-ii/

http://blog.ine.com/2008/03/17/understanding-redistribution-part-iii/

SW1:

```
!
! See the breakdown for explanation on the ADs
!
router ospf 1
 redistribute rip subnets
 distance ospf external 169
!
router rip
 redistribute ospf 1 metric 1
```

R2:

```
router eigrp 10
!
! EIGRP metric values chosen are arbitrary
!
redistribute ospf 1 metric 1 1 1 1 1
!
router ospf 1
 redistribute eigrp 10 subnets metric 20
```

R3:

```
router eigrp 10
 redistribute ospf 1 metric 1 1 1 1 1
!
router ospf 1
 redistribute eigrp 10 subnets metric 30
```

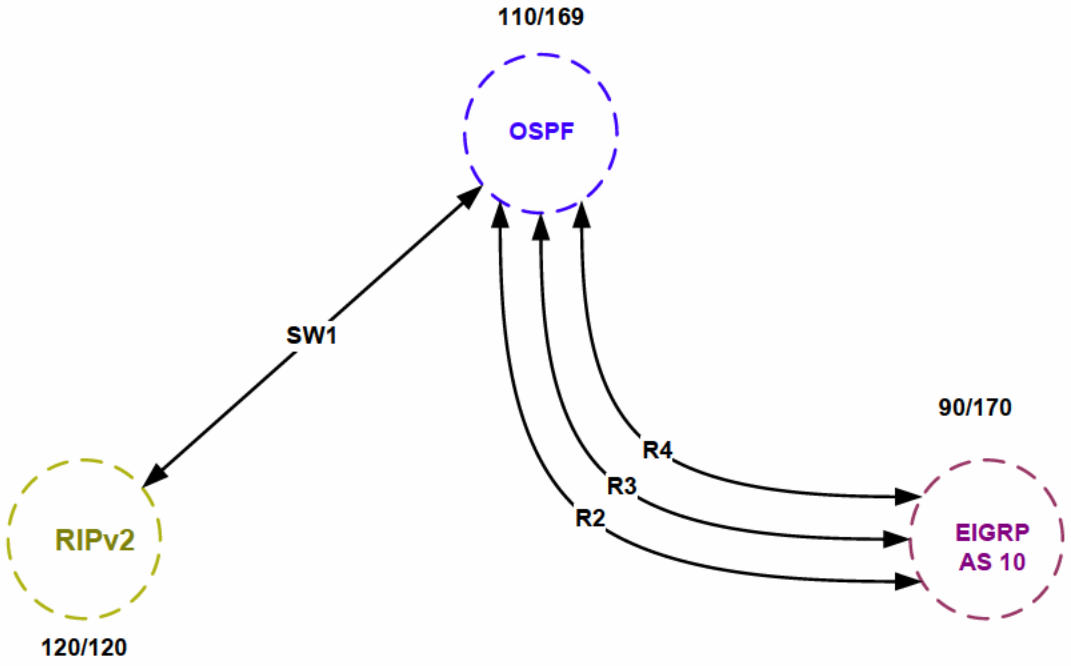
R4:

```
router eigrp 10
 redistribute ospf 1 metric 1 1 1 1 1
!
router ospf 1
 redistribute eigrp 10 subnets metric 40
```


Task 2.4 Breakdown

The redistribution part is slightly complicated in this scenario. We have three points of redistribution configured between OSPF and EIGRP AS 10 routing domains. To perform quick analysis, draw a “redistribution” diagram as follows:

1) Outline all routing domains used in the scenarios along with their names. In our case there are three domains: RIPv2, OSPF and EIGRP AS 10
2) Connect the routing domains using lines that represent routers performing redistribution. In our case those are SW1, R2, R3 and R4.



The arrow endpoints represent the redistribution direction. The diagram shown above clearly demonstrates that there is one central domain – OSPF – that carries routes between RIPv2 and EIGRP AS 10. Before you even begin the analysis, you need to decide on the administrative distance for all routing protocols. The general rule is as follows:

Rule 1: *For every IGP, the distance for non-native (external) prefixes should be higher than any other native distance among other routing protocols.*

Here "native" prefixes means prefixes that have been originated inside the routing domain, for example R5's connected interfaces are native to EIGRP AS 10 in this case. The purpose of this rule is to ensure that any router running two IGPs will prefer the "native" path to every IGP's native prefixes, thus avoiding sub-optimal routing or routing loops. Strictly speaking you need to apply this rule only if two IGP domains have more than one point of redistribution.

EIGRP by default sets the AD for all external prefixes to 170, which is a reflection of the above rule. OSPF's external AD is the same as the AD for internal routes – 110. RIP has no notion of external routes, but you may enforce distance using the distance command coupled with and access-list.

What would be the choices in our case? Should we change OSPF's external AD to ensure it does not preempt any other protocols native prefixes? The answer is "No", based on the following rule:

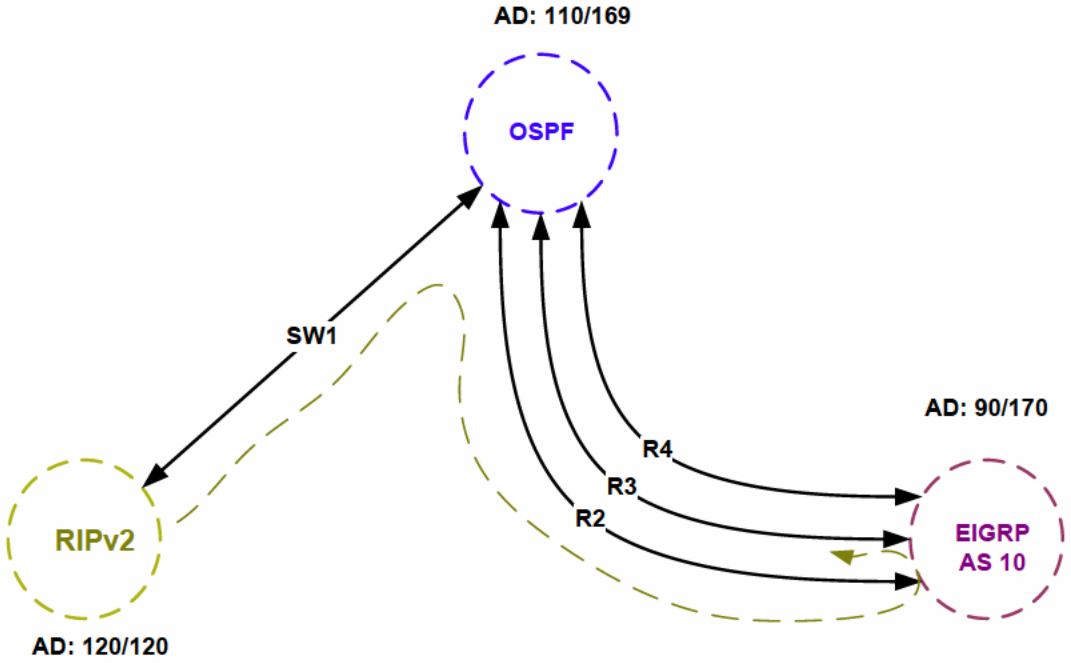
Rule 2: *If there is a central routing domain providing transit services to other "edge" routing domains its external AD should be the lowest among other routing domain ADs.*

This rule ensures that every router participating into two or more IGPs will chose the "central" IGP to reach other domain's external prefixes. This rule works well in topologies that that a central IGP domain, i.e. in star-like topologies. If all domains connect using "full-mesh" the choice is less obvious and based on the "primary" and "secondary" path. In our case, since OSPF is the "core" routing domain, we don't have to bother changing its external AD, since its already above the EIGRP's native AD of 70.

After we decided on the administrative distance values, our final step is to check if there are any routing loops remaining. The simplest way to detect this is to perform route-tracing using the following rule:

Rule 3: *For every routing domain, check if its native prefixes loop back through the redistribution topology. Start with a given domain and follow along the arrows noticing if the routes may ever loop back.*

For example, start with RIPv2 domain as shown on the figure below. RIPv2 prefixes travel across OSPF domain and may enter EIGRP domain by either of three ways. Let's say we select the path via R2. The prefixes may then loop back to OSPF via R3 or R4. However, the OSPF external AD on R3 and R4 is better than EIGRP's external AD and thus R3 and R4 would prefer to reach RIPv2 prefixes via OSPF. The same principle applies to OSPF and EIGRP prefixes looped back through the "connected" domain. No prefixes could ever loop through RIPv2 as this domain has only one connection to OSPF.



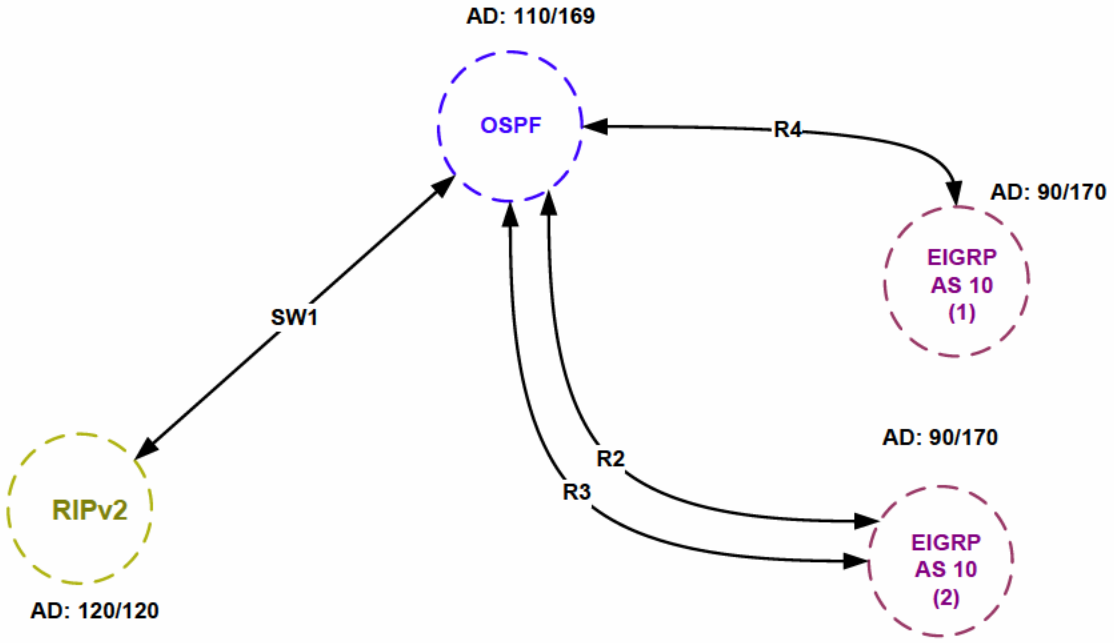
Notice that if OSPF's external AD was higher than EIGRP's AD, then RIPv2 prefixes may loop back to OSPF and return to SW1. However, OSPF's external AD would still be higher than RIPv2's native AD and RIPv2 prefixes will not be preempted.

The above loop tracking procedure will allow you to spot any prefixes that may loop back to the domain of origin. You may then use filtering techniques, such as based on tagging to prevent those feedback loops. But before you do that, you need one addition check – look for anomalies.

Rule 4: *Anomalies are non-standard configurations which do either of the following:*

1) *Originate native prefixes into the IGP domain as "external" that is via redistribution*
2) *Perform routing information hiding by means of summarization or filtering, effectively preventing complete routing information propagation through the given IGP*
3) *Unintended filtering scenarios, possible with distance-vector protocols. For example if R3 would learn R6's prefixes via OSPF and prefer them for some reason it would not propagate them down to R5. This is also considered an "information hiding" scenario*
4) *Additional best-path manipulation needed for the purpose of traffic engineering, e.g. for multicast section.*

Every of the above cases require special consideration and handling. So far we only concern ourselves with the current scenario, which apparently has none of those special cases – there are no native prefixes originated via redistribution, which is a good thing, and no other special cases. Thus, simply adjust the ADs per the analysis performed will do the trick. In the event if R3's connection to R5 fails, R4 still connects EIGRP domain to the OSPF cloud and just EIGRP domain becomes fragmentary. No routing loops form and full-reachability remains in place.



Task 2.4 Verification

Verify full IGP reachability using the following TCL script:

```
tclsh
foreach i {
132.1.0.1
132.1.17.1
150.1.1.1
132.1.0.2
132.1.23.2
150.1.2.2
132.1.26.2
132.1.3.3
132.1.0.3
132.1.23.3
150.1.3.3
132.1.35.3
132.1.33.3
132.1.0.4
150.1.4.4
132.1.255.4
132.1.45.4
132.1.5.5
150.1.5.5
132.1.35.5
132.1.45.5
192.10.1.5
54.1.2.6
150.1.6.6
132.1.26.6
132.1.17.7
150.1.7.7
204.12.1.7
132.1.8.8
150.1.8.8
204.12.1.8
132.1.255.9
132.1.255.10
} { puts [ exec "ping $i timeout 1" ] }
```


Do not worry if you can not ping local unmapped IP addresses on Frame Relay multipoint and physical interfaces. If you are uncertain as to the requirement for your particular lab ask the proctor for clarification. Also for now ignore the 132.1.45.0/24 subnet as it's the backup link between R4 and R5.

In order to test full-reachability in situation where the link between R3 and R5 is down, remove the DLCI on any of the respective point-to-point sub-interfaces. This will cause backup link to come up after the interface is declared down. It may take some time for the new EIGRP adjacency to establish and the IGP protocols to converge. It may even require issuing the command **`clear ip route *`** on R2 and R3 to give the IGP protocols a signal to update RIB.

As additional verification bring down the Frame Relay link between R3 and R5 by removing the DLCI from either side's subinterface. Once the backup interface is out of the standby state, rerun the ping script.

Although the 3550's and 3560's do not support the TCL shell they do support macros. The macro below can be used for testing from the switches.


```
conf t
macro name ping_internal
do ping 132.1.0.1
do ping 132.1.17.1
do ping 150.1.1.1
do ping 132.1.0.2
do ping 132.1.23.2
do ping 150.1.2.2
do ping 132.1.26.2
do ping 132.1.3.3
do ping 132.1.0.3
do ping 132.1.23.3
do ping 150.1.3.3
do ping 132.1.35.3
do ping 132.1.33.3
do ping 132.1.0.4
do ping 132.1.255.4
do ping 150.1.4.4
do ping 132.1.5.5
do ping 150.1.5.5
do ping 132.1.35.5
do ping 192.10.1.5
do ping 54.1.2.6
do ping 150.1.6.6
do ping 132.1.26.6
do ping 132.1.17.7
do ping 150.1.7.7
do ping 204.12.1.7
do ping 132.1.8.8
do ping 150.1.8.8
do ping 204.12.1.8
do ping 132.1.255.9
do ping 132.1.255.10
@
macro global apply ping_internal
```

Although points are not taken away for additional configuration it is advisable to remove the macros from the configuration prior to leaving the lab.

Lastly, verify reachability to the backbone IGP networks with following TCL script and macro:


```
tclsh
foreach i {
200.0.0.1
200.0.1.1
200.0.2.1
200.0.3.1
31.3.0.1
31.1.0.1
30.3.0.1
30.1.0.1
} { puts [ exec "ping $i" ] }

SW1 and SW2:
conf t
macro name ping_external
do ping 200.0.0.1
do ping 200.0.1.1
do ping 200.0.2.1
do ping 200.0.3.1
do ping 31.3.0.1
do ping 31.1.0.1
do ping 30.3.0.1
do ping 30.1.0.1
@
macro global apply ping_external
```


Task 2.5 Solution

🕮 Before You Begin

This BGP porting of this lab is relatively straight-forward and does not affect any routers but R5. Therefore, there is no need for crearting a separate detailed BGP diagram.

The first task uses BGP authentication. For more information on BGP authentication which is based on TCP MD5 option you may look at VOL1's scenario: **`BGP > Authenticating BGP Peering.`** Local-AS feature is very helpful in AS merging and migration, as it allows for retaining the old AS# on some connection. The use of this feature is illustrated in VOL1 lab **`BGP > BGP Local AS`**.

R5:

```
router bgp 200
 neighbor 192.10.1.254 remote-as 254
 neighbor 192.10.1.254 password CISCO
```

SW1:

```
router bgp 400
!
! The no-prepend is required to make sure AS 100 learns AS 54 prefixes
! Otherwise they will be rejected due to the local AS# in the AS_PATH
!
 neighbor 204.12.1.254 local-as 100 no-prepend
```


Task 2.5 Breakdown

The task points toward using BGP authentication on R5 and Local-AS feature on SW1. There is a hidden catch here. By default, when Local-AS is enabled, BGP peer will prepend the local AS# number to the routes send toward iBGP peers. The paths will propagate across AS400 and AS 3000 but will fail to enter AS 100 due to AS_PATH loop prevention rule. Therefore, in order to make real AS 100 accept those prefixes the **no-prepend** option is needed when setting up local-AS.

🕮 Further Learning

An additional mechanism for BGP security is know as session TTL security, which prevents session spoofing. You may find a detailed outline of the security process in VOL1's scenario **BGP> BGP TTL Security.** If you are looking for more information on Local-AS feature, check the addition VOL1 scenario named **BGP > BGP Local-AS/Replace-AS Dual-AS**.

🕮 Deep Dive

- What other protocol you know that uses TCP MD5 option for authentication?
- How could firewalls react on the presence of TCP MD5 option?
- What is the purpose of the "replace-as" option to local-as .. no-prepend command?

Task 2.5 Verification

Try setting wrong BGP password and see results:

```
Rack1R5#conf t
Rack1R5(config)#router bgp 200
Rack1R5(config-router)#no neighbor 192.10.1.254 password CISCO
Rack1R5(config-router)#neighbor 192.10.1.254 password CISCO1
Rack1R5(config-router)#do clear ip bgp 192.10.1.254
%BGP-5-ADJCHANGE: neighbor 192.10.1.254 Down User reset
%TCP-6-BADAUTH: Invalid MD5 digest from 192.10.1.254(179) to
192.10.1.5(49258)
%TCP-6-BADAUTH: Invalid MD5 digest from 192.10.1.254(179) to
192.10.1.5(49258)
```

Verify that local-AS is configured:

```
Rack1SW1#show ip bgp neighbors 204.12.1.254 | inc local
BGP neighbor is 204.12.1.254,  remote AS 54,  local AS 100 no-prepend,
external link
```

Verify that the local-AS is not prepended on iBGP peering session:

```
Rack1SW1#show ip bgp neighbors 204.12.1.8 advertised-routes
<output omitted>

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   204.12.1.254             0             0 54 i
*> 28.119.17.0/24   204.12.1.254             0             0 54 i
*> 112.0.0.0        204.12.1.254                           0 54 50 60 i
*> 113.0.0.0        204.12.1.254                           0 54 50 60 i
```

Remove the “no-prepend” keyword to see the difference:

```
Rack1SW1#conf t
Rack1SW1(config)#router bgp 400
Rack1SW1(config-router)#no neighbor 204.12.1.254 local-as 100 no-
prepend
Rack1SW1(config-router)#neighbor 204.12.1.254 local-as 100 no-prepend
%BGP-5-ADJCHANGE: neighbor 204.12.1.254 Down Local AS change
Rack1SW1(config-router)#neighbor 204.12.1.254 local-as 100
Rack1SW1#show ip bgp neighbors 204.12.1.8 advertised-routes
<output omitted>
   Network          Next Hop       Metric LocPrf Weight Path
*> 28.119.16.0/24   204.12.1.254        0             0 100 54 i
*> 28.119.17.0/24   204.12.1.254        0             0 100 54 i
*> 112.0.0.0        204.12.1.254                      0 100 54 50 60 i
*> 113.0.0.0        204.12.1.254                      0 100 54 50 60 i
*> 114.0.0.0        204.12.1.254                      0 100 54 i
<output omitted>
```


Task 2.6 Solution

🕮 Before You Begin

This scenario employs filtering based on AS PATH matching. AS_PATH regular expressions have been covered previously in Lab1, and you may want to check VOL1's scenario **BGP > BGP Regular Expressions** for more hands-on practice.

SW1:

```
router bgp 400
 neighbor 204.12.1.254 route-map STOP_TRANSIT_TO_AS_254 out
!
ip as-path access-list 1 permit _254$
!
route-map STOP_TRANSIT_TO_AS_254 deny 10
 match as-path 1
!
route-map STOP_TRANSIT_TO_AS_254 permit 20
```

Task 2.6 Verification

The task inplicitly requires filtering AS 254 prefixes from entering AS 400 in orde to prevent AS 400 from ever accessing AS 254. This is achieved by using BGP regular expression for the prefixes originated in AS 254. A route map is then used ingress on AS 400's connection to BB3. Notice that using a route-map is more flexible approach than using a filter-list as it allows for accumulating the whole policy in a single route map and avoiding order of operation collisions.

🕮 Deep Dive

- How could you implement the same "non-transit" filtering in scalable manner without using AS_PATH matching?

Task 2.6 Verification

Check if we have AS 254 originated routes in BGP table:

```
Rack1SW1#show ip bgp regexp _254$
<output omitted>
   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      132.1.17.1                             0 300 200 254 ?
*> 220.20.3.0       132.1.17.1                             0 300 200 254 ?
*> 222.22.2.0       132.1.17.1                             0 300 200 254 ?
```

Make sure they are not advertised to AS 54

```
Rack1SW1#show ip bgp neighbor 204.12.1.254 advertised-routes

Total number of prefixes 0
Rack1SW1#
```

Task 2.7 Solution

🕮 Before You Begin

This scenario utilizes BGP route aggregation and prefix-filtering to block the aggregate. If you need introductory information, check VOL1 scenarios named

BGP > BGP Aggregation
BGP > BGP Aggregation - Summary-Only
BGP > BGP Filtering With Prefix-Lists

R5:

```
!
! Notice that a network should be in BGP table for aggregation
! to work properly.
!
router bgp 200
 network 132.1.5.0 mask 255.255.255.0
 aggregate-address 132.1.0.0 255.255.0.0 summary-only
 neighbor 132.1.35.3 route-map DENY_AGGREGATE out
 neighbor 132.1.45.4 route-map DENY_AGGREGATE out
!
! This configuration blocks aggregate from being sent to iBGP peers
!
ip prefix-list DENY_AGGREGATE seq 5 permit 132.1.0.0/16
!
route-map DENY_AGGREGATE deny 10
 match ip address prefix-list DENY_AGGREGATE
!
route-map DENY_AGGREGATE permit 20
```


Task 2.7 Breakdown

The question explicitly specifies the summary prefix length. The rest of the configuration is basic, involving creating route-map to filter the summary prefix sent to the internal peers. Notice that a prefix needs to be in BGP table before the aggregation could be configured.

🕮 Deep Dive

- What other aggregation methods could be used?
- What is could be a disadvantage of BGP table aggregation?
- Why it could be important to filter summary from being advertised to internal peers?

Task 2.7 Verification

Verify that the aggregate is being generated:

```
Rack1R5#show ip bgp
<output omitted>
*> 132.1.0.0         0.0.0.0                            32768 i
s> 132.1.5.0/24      0.0.0.0                  0         32768 i
```

Check that we send only the summary route to BB2:

```
Rack1R5#show ip bgp neig 192.10.1.254 advertised-routes | inc 0.0.0.0
<snip>
*> 132.1.0.0         0.0.0.0                            32768 i
```

Check that R5 does not send the summary to R3 and R4:

```
Rack1R5#show ip bgp neighbors 132.1.35.3 advertised-routes | inc
132.1.0.0

Rack1R5#
```

Task 2.8 Solution

🕮 Before You Begin

This scenario requires knowledge of BGP performance features, which are briefly explained in the breakdown section for this task. You may additionally want to check the **VOL1** scenario named **BGP > BGP Next-Hop Trigger** for additional information on the next-hop tracking feature.

```
R5:
!
! Enable BGP next-hop trigger delay and change advertisement interval
!
router bgp 200
 bgp nexthop trigger delay 15
 neighbor 192.10.1.254 advertisement-interval 3
```

Task 2.8 Breakdown

The task calls for the use of BGP even-driven best-path computation. In the past, BGP would be updating next-hop values and verifying their validity every 60 seconds by default, with the run of BGP scanner. The 12.3T feature allows for registering BGP next-hops with IGP event handlers that wake up BGP scanner process on respective IGP changes (e.g. metric change or prefix loss). This allows for reducing CPU load as periodic BGP scanner no longer needs to process ALL prefixes and make BGP more responsive to IGP change. The command **bgp nexthop trigge delay X** instructs BGP to wake up the BGP scanner only X seconds after the first event notification was received. This allows for IGP protocol to converge and any additional IGP event to be taken in account, thus avoiding excessive reaction to changes and improving stability.

The **advertisement-interval** N command instructs the local BGP speaker to wait for N seconds before sending a consecutive update to the peer. This allows for better grouping of changes and improved stability as the updates are now batched and processed optimally on the receiving router.

Task 2.8 Verification

Check the advertisement timer set for BB2:

```
RSRack1R5#sh ip bgp neighbors 192.10.1.254 | include advertisement
  Default minimum time between advertisement runs is 30 seconds
  Minimum time between advertisement runs is 3 seconds
```


Task 3.1 Solution

🕮 Before You Begin

This scenario requires you configuring IPv6 over Frame-Relay. There are some caveats involved in the process, and if you don't feel yourself solid with IPv6 you may need to try VOL1's scenarion **IPv6 > IPv6 Link Local Addressing** and **IPv6 > IPv6 Unique Local Addressing**.

R2:

```
ipv6 unicast-routing
!
interface Loopback0
 ipv6 address 2001:CC1E:1::2/128
!
! Map IPv6 addresses on Frame-Relay
!
interface Serial0/0
 ipv6 address 2001:CC1E:1:2323::2/64
 frame-relay map ipv6 2001:CC1E:1:2323::3 203 broadcast
!
ipv6 route 2001:CC1E:1::3/128 Serial0/0 2001:CC1E:1:2323::3
```

R3:

```
ipv6 unicast-routing
!
interface Loopback0
 ipv6 address 2001:CC1E:1::3/128
!
! Map IPv6 addresses on Frame-Relay
!
interface Serial1/0
 ipv6 address 2001:CC1E:1:2323::3/64
 frame-relay map ipv6 2001:CC1E:1:2323::2 302 broadcast
!
ipv6 route 2001:CC1E:1::2/128 Serial1/0 2001:CC1E:1:2323::2
```

Task 3.1 Breakdown

Frame Relay is a non-broadcast multi-access (NBMA) media. This implies that for multipoint configurations layer 3 to layer 2 resolution must be obtained. Since only static routing is used, a mapping is not required to the remote link-local address. If dynamic IPv6 routing were configured a mapping for the remote link-local address would be required.

Task 3.1 Verification

Verify the Frame Relay IPv6 layer 3 to layer 2 mappings:

```
Rack1R3#show frame-relay map
<output omitted>
Serial1/0 (up): ipv6 2001:CC1E:1:2323::2 dlci 302(0x12E,0x48E0),
static,
              broadcast,
              CISCO, status defined, active
```

Verify L3 reachability:

```
Rack1R3#ping 2001:CC1E:1::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/31/32 ms
```


Task 4.1 Solution

🕮 Before You Begin

This scenario uses AToM for Layer 2 VPNs. Even though it's not a likely candidate in the lab exam, knowing layer 2 VPNs is helpful in understanding MPLS technologies overall. You may find the necessary fundamental information by completing the following VOL1 scenarios:

MPLS VPN > MPLS LDP
MPLS VPN> AToM

R4:

```
mpls ip
!
interface Loopback100
 ip address 150.1.44.44 255.255.255.255
!
mpls ldp router-id Loopback100 force
!
! The tunnel link is needed to traverse the non-MPLS cloud
! MPLS should be enabled to accept tagged packets
!
interface Tunnel 46
 tunnel source Loopback0
 tunnel destination 150.1.6.6
 ip address 132.1.46.4 255.255.255.0
 mpls ip

!
! We use static routes as the simplest solution
!
ip route 150.1.66.66 255.255.255.255 Tunnel46

!
! VC types must match on both ends, i.e. both should be VLAN
!
interface FastEthernet 0/0.4
 encapsulation dot1q 4 native
 xconnect 150.1.66.66 46 encapsulation mpls
!
! The interface was shut down in the initial config
!
interface FastEthernet 0/0
 no shutdown
```


R6:

```
mpls ip
!
interface Loopback100
 ip address 150.1.66.66 255.255.255.255
!
mpls ldp router-id Loopback100 force
!
interface Tunnel 46
 tunnel source Loopback0
 tunnel destination 150.1.4.4
 ip address 132.1.46.6 255.255.255.0
 mpls ip
!
ip route 150.1.44.44 255.255.255.255 Tunnel46

!
interface FastEthernet 0/0.6
 xconnect 150.1.44.44 46 encapsulation mpls
```


Task 4.1 Breakdown

This task involves a technology that may not be a part of the real CCIE exam, but still very useful to know as an example of MPLS application. AToM allows for direct encapsulation and transporting of Layer 2 frames over a packet switched network. It does so by establishing two LSPs between the endpoints and allocating labels identifying locally attached circuits on every node. The resulting stack consists of two labels – transport and VC label.

In our case, there is no MPLS cloud between R4 and R6 and therefore we cannot establish and end-to-end transport LSP. A solution to this problem is using a GRE tunnel between R4 and R6 and running LDP over the tunnel. Since we use the existing Loopback0 interfaces for the tunnel we need additional Loopbacks for LDP endpoints. We create two of those and force LDP to use them, plus add two static routes over the tunnel to make the new interfaces reachable.

After MPLS session has been established you may configure xconnect statements using mpls encapsulation. The rest should come up automatically, provided that all interfaces are in no shutdown state.

🕮 Deep Dive

- What problems you may see using L2 VPN solutions?
- Why P2P L2 VPNs are so effective for hardware implementation?
- Can you use L2 VPN P2P tunnels to create multipoint connectivity?

Task 4.1 Verification

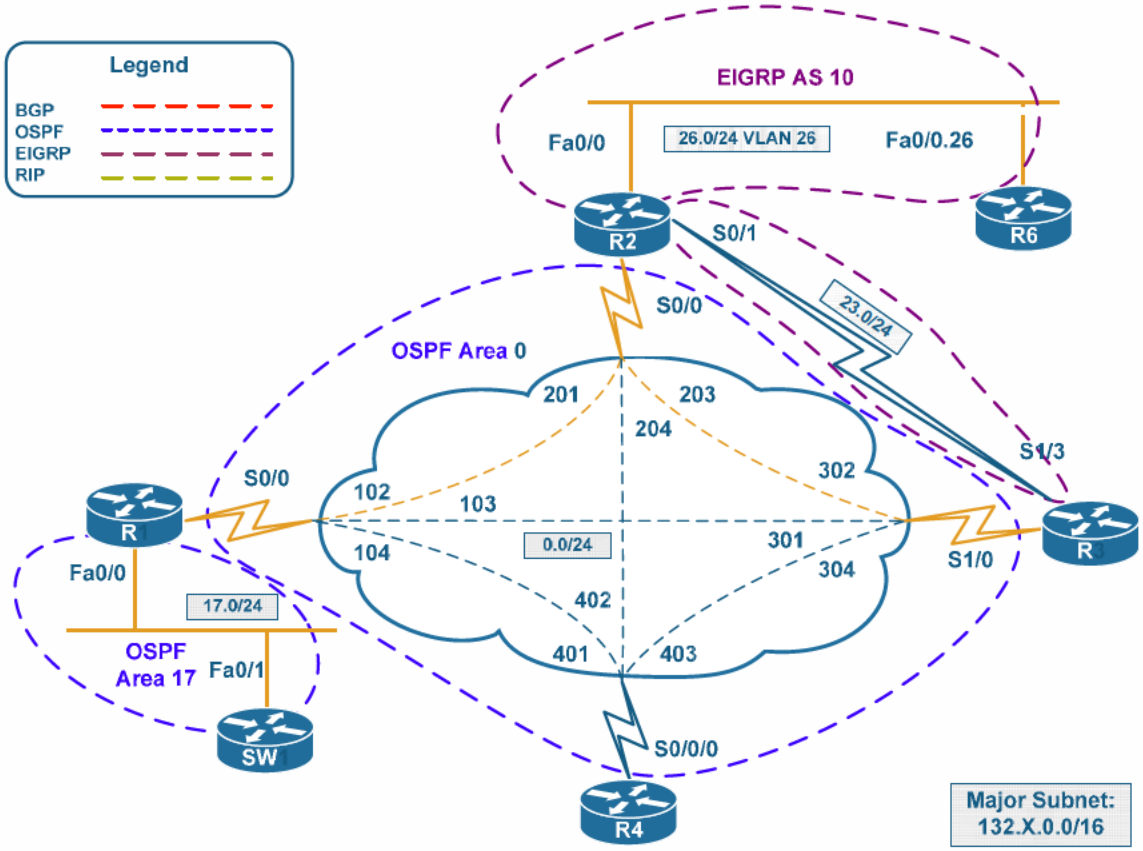
Check LDP peering session – AToM uses LDP for signaling.

```
RSRack1R4#show mpls ldp neighbor
    Peer LDP Ident: 150.1.66.66:0; Local LDP Ident 150.1.44.44:0
        TCP connection: 150.1.66.66.16608 - 150.1.44.44.646
        State: Oper; Msgs sent/rcvd: 38/42; Downstream
        Up time: 00:05:09
        LDP discovery sources:
          Targeted Hello 150.1.44.44 -> 150.1.66.66, active, passive
        Addresses bound to peer LDP Ident:
          132.1.26.6      54.1.2.6        150.1.6.6       150.1.66.66
          132.1.46.6

RSRack1R4#show mpls l2transport binding
  Destination Address: 150.1.66.66,  VC ID: 46
    Local Label:  65
        Cbit: 1,    VC Type: Eth VLAN,    GroupID: 0
        MTU: 1500,   Interface Desc: n/a
        VCCV: CC Type: CW [1], RA [2]
              CV Type: LSPV [2]
    Remote Label: 43
        Cbit: 1,    VC Type: Eth VLAN,    GroupID: 0
        MTU: 1500,   Interface Desc: n/a
        VCCV: CC Type: CW [1], RA [2]
              CV Type: LSPV [2]

Rack1R4#show mpls l2transport vc

Local intf     Local circuit              Dest address    VC ID
Status
-------------  -------------------------- --------------- ---------- --
Fa0/0.4        Eth VLAN 4                 150.1.66.66     46         UP

RSRack1R4#show mpls l2transport vc detail
Local interface: Fa0/0.4 up, line protocol up, Eth VLAN 4 up
  Destination address: 150.1.66.66, VC ID: 46, VC status: up
    Output interface: Tu46, imposed label stack {43}
    Preferred path: not configured
    Default path: active
    Next hop: point2point
  Create time: 00:05:26, last status change time: 00:05:25
  Signaling protocol: LDP, peer 150.1.66.66:0 up
    MPLS VC labels: local 65, remote 43
    Group ID: local 0, remote 0
    MTU: local 1500, remote 1500
    Remote interface description:
  Sequencing: receive disabled, send disabled
  VC statistics:
    packet totals: receive 275, send 211
    byte totals:   receive 16000, send 18321
    packet drops:  receive 0, seq error 0, send 0
```


Task 5.1 Solutions

🕮 Before You Being

Prior to starting with this task you need to discover the multicast topology using the PIM show commands (**`show ip pim neighbor`**, **`show ip pim interface`**). You will find that multicast is running on R1, R2, R3, R6 and SW1 as well as on the connecting interfaces. The following is the multicast topology diagram for this scenario (multicast-enabled links are highlighted).



If you need more fundamental information about the multicast technologies used in this scenario, you may practice VOL1 labs named **`Multicast > PIM Sparse Mode`** and look at the scenarios **`Multicast > PIM NBMA Mode`** and **`Multicast > RPF Failure`**.

```
R1, R2, R3 and R6:
!
! Configure static RP
!
ip pim rp-address 150.1.2.2

SW1:
ip pim rp-address 150.1.2.2
!
! Join the group on SW1
!
interface Vlan783
 ip igmp join-group 228.28.28.28
```

Task 5.1 Breakdown

Notice that the solution does not utilize PIM NBMA mode on the Frame-Relay interface. This will force R2 to send replica of the same multicast packet to all neighbors that have broadcasts mapped. You may observe the following on R3, even though it is not on the SPT toward the source:

```
RSRack1R3#debug ip mpacket fastswitch
IP multicast fastswitch packets debugging is on
RSRack1R3#
FS(0): Receive s=132.30.26.6 d=228.28.28.28 id=2211 prot=1
size=104(100) ttl=253 from Serial1/0 , dropped
RSRack1R3#
FS(0): Receive s=132.30.26.6 d=228.28.28.28 id=2212 prot=1
size=104(100) ttl=253 from Serial1/0 , dropped
RSRack1R3#
FS(0): Receive s=132.30.26.6 d=228.28.28.28 id=2213 prot=1
size=104(100) ttl=253 from Serial1/0 , dropped
RSRack1R3#un all
All possible debugging has been turned off
```

Based on this diagram you may expect to have potential for RPF failure on R3 or R1. The reason could be the fact that

- R3 sees the source on VLAN26 as a EIGRP prefix
- R1 see EIGRP redistributed routes with the next-hop IP address of R3.

However, this issues have been taken care of during the redistribution section. Firstly, R3's redistribution metric for EIGRP routes is higher than that of R2. Secondly, it does not really matter if R3 is accepting multicast packets or not – we are not required to ensure this.

🕮 Deep Dive

- What could be the benefits of using Static RP?
- Why should you use the command **`ip igmp join-group`** with caution?
- How PIM interprets the NBMA segments by default?

Task 5.1 Verification

Verify the joined groups and multicast routes:

```
Rack1SW1#show ip igmp groups
IGMP Connected Group Membership
Group Address    Interface                Uptime    Expires   Last
Reporter   Group Accounted
228.28.28.28     Vlan783                  00:09:14  00:02:09
204.12.1.7
224.0.1.40       FastEthernet0/1          01:38:20  00:02:21
132.1.17.1
224.0.1.40       Vlan783                  01:43:58  00:02:07
204.12.1.7

Rack1SW1#show ip mroute
<output omitted>
(*, 228.28.28.28), 00:00:41/00:02:18, RP 150.1.2.2, flags: SJCL
  Incoming interface: FastEthernet0/1, RPF nbr 132.1.17.1
  Outgoing interface list:
    Vlan783, Forward/Sparse, 00:00:41/00:02:18, H
```

Use mtrace to see how the packets should flow through the network:

```
Rack1SW1#mtrace 132.1.26.6 228.28.28.28
Type escape sequence to abort.
Mtrace from 132.1.26.6 to 132.1.17.7 via group 228.28.28.28
From source (?) to destination (?)
Querying full reverse path...
 0  132.1.17.7
-1  132.1.17.7 PIM  [132.1.26.0/24]
-2  132.1.17.1 PIM  [132.1.26.0/24]
-3  132.1.0.2 PIM Reached RP/Core [132.1.26.0/24]
```

Use ping to verify the configuration:

```
Rack1R6#debug ip mpacket
Rack1R6#ping 228.28.28.28

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 228.28.28.28, timeout is 2 seconds:

IP(0): s=132.1.26.6 (FastEthernet0/0.26) d=228.28.28.28 id=498,
ttl=254, prot=1, len=114(100), mroute olist null
Reply to request 0 from 132.1.17.7, 8 ms
Reply to request 0 from 132.1.17.7, 12 ms
```


Finally look at the output of the multicast routing table:

```
Rack1R6#show ip mroute
<output omitted>

(132.1.26.6, 228.28.28.28), 00:01:09/00:02:24, flags: FT
  Incoming interface: FastEthernet0/0.26, RPF nbr 0.0.0.0, Registering
  Outgoing interface list:
    FastEthernet0/0.26, Forward/Sparse, 00:01:09/00:03:17
```

Task 5.2 Solution

🕮 Before You Begin

Even though the solution is just one command, the technology behind it is a little convoluted, requiring your understanding of multicast switching fundamentals. You may find enough information in VOL1's scenario **Multicast > PIM NMBA Mode**.

```
R2:
interface Serial0/0
 ip pim nbma-mode
```

Task 5.2 Breakdown

PIM NBMA mode is based on PIM SM explicit Join messages. This allows the upstream node to detect downstream neighbors that actually want to receive multicast traffic and optimally switch the traffic in the fast-path as now all the downstream receivers are know and there is no need for process-mode pseudo-broadcast. When PIM NBMA mode is configured, the router treats each PIM neighbor as if it was reachable via a point-to-point link/sub-interface.

🕮 Deep Dive

- What could be an alternative to using PIM NBMA mode?
- What is the possible drawback of PIM NBMA mode in large mesh topologies?

Task 5.2 Verification

Ping the group joined by SW1:

```
Rack1R6#ping 228.28.28.28 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 228.28.28.28, timeout is 2 seconds:

Reply to request 0 from 132.1.17.7, 60 ms
Reply to request 1 from 132.1.17.7, 60 ms
Reply to request 2 from 132.1.17.7, 60 ms
```

Verify that per-destination mroute states appear on R2:

```
Rack1R2#show ip mroute 228.28.28.28
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 228.28.28.28), 00:29:41/00:03:24, RP 150.1.2.2, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0, 132.1.0.1, Forward/Sparse, 00:00:06/00:03:24

(132.1.26.6, 228.28.28.28), 00:09:37/00:03:26, flags: T
  Incoming interface: FastEthernet0/0, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0, 132.1.0.1, Forward/Sparse, 00:00:13/00:03:24
```



```
Rack1R2#show ip pim interface serial 0/0 detail
Serial0/0 is up, line protocol is up
  Internet address is 132.1.0.2/24
  Multicast switching: fast
  Multicast packets in/out: 0/45
  Multicast TTL threshold: 0
  PIM: enabled
    PIM version: 2, mode: sparse
    PIM DR: 132.1.0.3
    PIM neighbor count: 2
    PIM Hello/Query interval: 30 seconds
    PIM Hello packets in/out: 347/183
    PIM State-Refresh processing: enabled
    PIM State-Refresh origination: disabled
    PIM NBMA mode: enabled
    PIM ATM multipoint signalling: disabled
    PIM domain border: disabled
  Multicast Tagswitching: disabled
```


Task 6.1 Solution

🕮 Before You Begin

This scenario involves tuning a group of the technologies, such as IP srouce route, BOOTP (not DHCP) server, disabling Proxy ARP and CDP and setting banners. You may want to practice the following VOL1 scenarios (if not already) for additional knowledge on the subject:

IP Services > Proxy ARP
IP Services > DHCP

And read the following Wikipedia articles:

http://en.wikipedia.org/wiki/Source_routing
http://en.wikipedia.org/wiki/Bootstrap_Protocol

R5:

```
!
! Disable services per the task requirements
!
no ip source-route
no ip bootp server
!
interface FastEthernet0/1
 no ip proxy-arp
 no cdp enable
!
banner login "Access to this device or the attached networks is
prohibited without express written permission.  Violators will be shot
on sight."
```


Task 6.1 Breakdown

By default, IOS supports source-based routing which opens potential for IP address spoofing as an attacker may instruct the routers to pick up arbitrary path for the traffic. This feature should normally be disabled, along with the BOOTP process which is on by default along with DHCP. You don't really need BOOTP unless you're doing AutoInstall over Frame-Relay, so it's safe to disable it.

The last two interface-based services are Proxy-ARP and CDP. Proxy ARP is the feature that you normally don't need unless you want misconfigured (e.g. hosts with wrong netmask) to has connectivity. Additionally it opens routes for potential attacks based on ARP cache overflow as router responds to every unanswered ARP query. You don't need Proxy ARP on a properly configured segment so it is safe to disable.

Lastly, CDP is normally disable on interfaces connecting to the public networks, to prevent other people from learning any information about your router.

🕮 Deep Dive

- What could be the use of Proxy ARP?
- Why proxy ARP could be dangerous?
- How could an attacker use IP source routing?
- What good use you could think of for IP source routing?

Task 6.1 Verification

Verify that CDP is disabled on interface FastEthernet0/1 – compare the outputs for the two interfaces:

```
Rack1R5#show cdp interface Fa0/0
FastEthernet0/0 is up, line protocol is up
  Encapsulation ARPA
  Sending CDP packets every 60 seconds
  Holdtime is 180 seconds

Rack1R5#show cdp interface Fa0/1

Rack1R5#
```

Verify that the commands are in the configuration:

```
Rack1R5#show run | include (source-route|bootp)
no ip source-route
no ip bootp server
```

If you really want to see how source-routing works, try the following command from R4:

```
Rack1R5#show ip traffic | section Opts
  Opts:  0 end, 0 nop, 0 basic security, 0 loose source route
         0 timestamp, 0 extended security, 0 record route
         0 stream ID, 0 strict source route, 0 alert, 0 cipso, 0 ump
         0 other

Rack1R4#traceroute
Protocol [ip]:
Target IP address: 222.22.2.1
Source address:
Numeric display [n]:
Timeout in seconds [3]:
Probe count [3]:
Minimum Time to Live [1]:
Maximum Time to Live [30]:
Port Number [33434]:
Loose, Strict, Record, Timestamp, Verbose[none]: L
Source route: 132.1.0.1 132.1.0.2 132.1.23.3 132.1.35.5 192.10.1.254
Loose, Strict, Record, Timestamp, Verbose[LV]:

Rack1R5#show ip traffic | section Opts
  Opts:  16 end, 0 nop, 0 basic security, 16 loose source route
         0 timestamp, 0 extended security, 0 record route
         0 stream ID, 0 strict source route, 0 alert, 0 cipso, 0 ump
         0 other
```


Now try it with source-routing enabled on R5.

```
Rack1R5#show running-config interface Fa0/1

interface FastEthernet0/1
 ip address 192.10.1.5 255.255.255.0
 no ip proxy-arp
 half-duplex
 no cdp enable
end
```

To be sure that Proxy-ARP is disabled, issue the following command:

```
Rack1R5#sh ip interface Fa0/1 | include Proxy
  Proxy ARP is disabled
  Local Proxy ARP is disabled
```

Verify the login banner:

```
Rack1R3#telnet 150.1.5.5
Trying 150.1.5.5 ... Open
Access to this device or the attached networks is
prohibited without express written permission. Violators will be shot
on sight.

User Access Verification

Password:
```


Task 6.2 Solutions

🕮 Before You Begin

The objective of this scenario is dropping SNMP packets coming from outside of the network. For a relatively simple goal, Zone-Based Firewall is being used, which introduces configuration complexity. ZFW is known for it's highly modularity and configuration re-usability, but features a little cumbersome MQC-style syntax. For introduction to ZFW, you may want to practice the following VOL1 scenario:

`Security > Zone Based Firewall`

R5:

```
!
! ACL for SNMP classification
!
ip access-list extended ACL_SNMP
 permit udp any any eq SNMP

!
! Class-map for SNMP traffic
!
class-map type inspect CMAP_SNMP
 match access-group name ACL_SNMP

!
! Inspection policy for Outside to Inside Traffic
!
policy-map type inspect PMAP_FROM_OUTSIDE_TO_INSIDE
 class type inspect CMAP_SNMP
  drop
 class class-default
  pass

!
! Inspection policy for Inside to Outside Traffic
!
policy-map type inspect PMAP_FROM_INSIDE_TO_OUTSIDE
 class class-default
  pass
!
zone security OUTSIDE
zone security INSIDE

!
! Zone-pair for Outside to Inside Traffic
!
zone-pair security ZP_OUTSIDE_TO_INSIDE source OUTSIDE destination
INSIDE
  service-policy type inspect PMAP_FROM_OUTSIDE_TO_INSIDE
!
! Zone-Pair for Inside to Outside Traffic
!
zone-pair security ZP_INSIDE_TO_OUTSIDE source INSIDE destination
OUTSIDE
  service-policy type inspect PMAP_FROM_INSIDE_TO_OUTSIDE

!
interface FastEthernet0/1
 zone-member security OUTSIDE
!
interface FastEthernet0/0
 zone-member security INSIDE
!
interface Serial0/1/0
 zone-member security INSIDE

interface Serial0/0/0.1
 zone-member security INSIDE
```


R6:

```
!
! ACL for SNMP classification
!
ip access-list extended ACL_SNMP
 permit udp any any eq SNMP

!
! Class-map for SNMP traffic
!
class-map type inspect CMAP_SNMP
 match access-group name ACL_SNMP

!
! Inspection policy for Outside to Inside Traffic
!
policy-map type inspect PMAP_FROM_OUTSIDE_TO_INSIDE
 class type inspect CMAP_SNMP
  drop
 class class-default
  pass

!
! Inspection policy for Inside to Outside Traffic
!
policy-map type inspect PMAP_FROM_INSIDE_TO_OUTSIDE
 class class-default
  pass
!
zone security OUTSIDE
zone security INSIDE

zone-pair security ZP_OUTSIDE_TO_INSIDE source OUTSIDE destination
INSIDE
  service-policy type inspect PMAP_FROM_OUTSIDE_TO_INSIDE

!
! Associate policy with zone pair
!
zone-pair security ZP_INSIDE_TO_OUTSIDE source INSIDE destination
OUTSIDE
  service-policy type inspect PMAP_FROM_INSIDE_TO_OUTSIDE

!
interface Serial0/0/0
 zone-member security OUTSIDE
!
interface FastEthernet0/0.26
 zone-member security INSIDE
```


Task 6.2 Breakdown

Normally, you would solve this task by creating an interface access-list and applying it ingress on R5 and R6. Using ZBFW is little more complicated for "small" scenarios but pays back for large and complex configurations. In our case, we have to crate an access-list and a class-map to match the dangerous traffic. We then create two inspect policy maps, one that applies to the traffic going from inside to outside and another that applies in the opposite direction. The outside to inside policy drops the SNMP packets and passes everything else. The inside to outside policy simple passes all traffic out.

For the permitted traffic we could as well do "inspection" instead of "pass". Since the task does not mention anything about the allowed traffic it makes more sense to just allow it as it was before the ZBF policy was applied.

🕮 Further Learning

Zone-Based Firewall is an advanced concept, which intends to simplify configuration complexity for large firewall scenarios. The problem at the moment is that Cisco does not position IOS routers as a replacement for large or even medium-size firewalls – IOS firewall feature is being mainly used in small and branch offices, where CBAC syntax is enough. However, if you are interested in this technology you may find and read the "Zone Based Firewall Design Guide" on Cisco's website for more in-depth information.

🕮 Deep Dive

- What main benefits of ZFW you may name over CBAC?
- How is zone-based policing different from MQC policing?

Task 6.2 Verification

Check for the zones created in the firewall:

```
Rack1R5#show zone security
zone self
  Description: System defined zone


zone OUTSIDE
  Member Interfaces:
    FastEthernet0/1


zone INSIDE
  Member Interfaces:
    FastEthernet0/0
    Serial0/1/0
    Serial0/0/0.1
```

Verify the inspection policy maps:

```
Rack1R5#show policy-map type inspect
  Policy Map type inspect PMAP_FROM_OUTSIDE_TO_INSIDE
    Class CMAP_SNMP
      Drop
    Class class-default
      Pass

  Policy Map type inspect PMAP_FROM_INSIDE_TO_OUTSIDE
    Class class-default
      Pass

Rack1R5#show policy-map type inspect zone-pair

policy exists on zp ZP_OUTSIDE_TO_INSIDE
 Zone-pair: ZP_OUTSIDE_TO_INSIDE

  Service-policy inspect : PMAP_FROM_OUTSIDE_TO_INSIDE

    Class-map: CMAP_SNMP (match-all)
      Match: access-group name ACL_SNMP
      Drop
        0 packets, 0 bytes
```



```
    Class-map: class-default (match-any)
      Match: any
      Pass
        0 packets, 0 bytes

policy exists on zp ZP_INSIDE_TO_OUTSIDE
 Zone-pair: ZP_INSIDE_TO_OUTSIDE

  Service-policy inspect : PMAP_FROM_INSIDE_TO_OUTSIDE

    Class-map: class-default (match-any)
      Match: any
      Pass
        0 packets, 0 bytes

Rack1R6#show zone security
zone self
  Description: System defined zone


zone OUTSIDE
  Member Interfaces:
    Serial0/0/0


zone INSIDE
  Member Interfaces:
    FastEthernet0/0.26

Rack1R6#show policy-map type inspect
  Policy Map type inspect PMAP_FROM_OUTSIDE_TO_INSIDE
    Class CMAP_SNMP
      Drop
    Class class-default
      Pass

  Policy Map type inspect PMAP_FROM_INSIDE_TO_OUTSIDE
    Class class-default
      Pass
```



```
Rack1R6#show policy-map type inspect zone-pair

policy exists on zp ZP_OUTSIDE_TO_INSIDE
 Zone-pair: ZP_OUTSIDE_TO_INSIDE

  Service-policy inspect : PMAP_FROM_OUTSIDE_TO_INSIDE

    Class-map: CMAP_SNMP (match-all)
      Match: access-group name ACL_SNMP
      Drop
        0 packets, 0 bytes

    Class-map: class-default (match-any)
      Match: any
      Pass
        0 packets, 0 bytes

policy exists on zp ZP_INSIDE_TO_OUTSIDE
 Zone-pair: ZP_INSIDE_TO_OUTSIDE

  Service-policy inspect : PMAP_FROM_INSIDE_TO_OUTSIDE

    Class-map: class-default (match-any)
      Match: any
      Pass
        0 packets, 0 bytes
```

Task 6.3 Solution

🕮Before You Begin

The complexity of SNMP lies in NMS configuration and proper reading interpretation and correlation. The protocol configuration itself is not hard, unless it involves access-control configuration for SNMPv3. If you need more fundamental information about SNMPv2c (v2 with commuity authentication) check the following VOL1's scenarios:

System Management > SNMPv2 Server
System Management > SNMPv2 Access Control.

For introduction to the protocol itself, check the following Wikipedia article:

http://en.wikipedia.org/wiki/Simple_Network_Management_Protocol

R2 and R4:

```
snmp-server community public RO 1
access-list 1 deny any log
logging 132.1.33.100
```

Task 6.3 Breakdown

The key to this section is to create an access-list that denies all IP address and includes the *log* keyword. The access-list is then bound to the RO community string of *public*. This is a useful technique to track down the source of a host attempting to poll a device.

o 🕮Deep Dive

- Can you report syslog messages via SNMP?
- Why is it so important to use access-lists with SNMPv2c?
- Name the SNMP ports and their use.

Task 6.3 Verification

Check for the configured SNMP community strings and then looks for messages logged via syslog:

```
Rack1R2#show snmp community

Community name: ILMI
Community Index: cisco0
Community SecurityName: ILMI
storage-type: read-only  active

Community name: public
Community Index: cisco1
Community SecurityName: public
storage-type: nonvolatile        active access-list: 1

Rack1R2#show logging | begin Trap
    Trap logging: level informational, 68 message lines logged
        Logging to 132.1.33.100 (udp port 514, audit disabled, link
up), 2 message lines logged, xml disabled,
              filtering disabled
```


Task 6.4 Solution

🕮 Before You Begin

The solution uses simple reflexive access-lists. As a introduction to this type of ACL, you may want to practice VOL1's scenario **`Security > Traffic Filtering using Reflexive Access-Lists`**. To fully understand what ICMP messages and UDP port numbers need to be open for the traceroute command, you may want to read the Wikipedia article about this utility:

http://en.wikipedia.org/wiki/Traceroute

R1:

```
interface FastEthernet0/0
 ip access-group TO_SW1 out
 ip access-group FROM_SW1 in
!
no ip access-list extended FROM_SW1
ip access-list extended FROM_SW1
 permit icmp any any time-exceeded
 permit icmp any any port-unreachable
!
! Permit returning multicast traffic as required by previous task
!
 permit icmp host 132.1.17.7 any echo
 evaluate ICMP
 deny   icmp any any
 permit ip any any
!
ip access-list extended TO_SW1
 permit icmp any host 228.28.28.28 echo
 permit icmp any any reflect ICMP
 permit ip any any
```


Task 6.4 Breakdown

The task has multiple solutions, including reflexive access-list, CBAC and zone-based firewall. The reflexive lists are selected for their relative simplicity and minimal side-effects on the configuration (e.g. CBAC would enable TCP/UDP intercept). The access-list in the solution reflects ICMP traffic only and explicitly prohibits any ICMP packets from every coming back. All other IP protocols are permitted. Notice that ICMP time-exceeded and Port-Unreachable messages are allowed to go back as those are required for traceroute, but there is no need to permit the returning UDP packets, as in traceroute their flow is unidirectional, and we are not required to permit inboud traceroute.

Pay attention to the fact that multicast traffic is not reflected in the reflexive ACLs. Thus, the multicast test will no longer work – the packets will reach SW1 but will be able to come back to R6. In the real exam you may want to consult with the proctor and ask if permitting the responses for the multicast probes is required.

🕮 Deep Dive

- What is the default port range used by UNIX-style traceroute?
- How is the Winwows tracroute utility different from UNIX-one?

Task 6.4 Verification

To verify our reflective ACL ping from R3 to BB2:

```
Rack1R2#ping 150.1.8.8 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 150.1.8.8, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!
```

```
Rack1R1#sh ip access-lists ICMP
Reflexive IP access list ICMP
     permit icmp host 150.1.8.8 host 132.1.0.2  (352 matches) (time
left 299)
```


Ensure multicast section is still working:

```
Rack1R6#ping 228.28.28.28 repeat 5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 228.28.28.28, timeout is 2 seconds:

Reply to request 0 from 132.1.17.7, 104 ms
Reply to request 1 from 132.1.17.7, 104 ms
Reply to request 2 from 132.1.17.7, 104 ms
Reply to request 3 from 132.1.17.7, 112 ms
Reply to request 4 from 132.1.17.7, 116 ms
```

Verify that traceroute from behind R1 is functional:

```
Rack1R2#traceroute 150.1.8.8

Type escape sequence to abort.
Tracing the route to 150.1.8.8

  1 132.1.0.1 32 msec 28 msec 28 msec
  2 132.1.17.7 28 msec 32 msec 28 msec
  3 204.12.1.8 32 msec *  28 msec

Rack1R1#show ip access-lists FROM_SW1
Extended IP access list FROM_SW1
    10 permit icmp any any time-exceeded (3 matches)
    20 permit icmp any any port-unreachable (2 matches)
    30 permit icmp host 132.1.17.7 any echo-reply (5 matches)
    40 evaluate ICMP
    50 deny icmp any any
    60 permit ip any any (50 matches)

Rack1R1#show ip access-lists TO_SW1
Extended IP access list TO_SW1
    10 permit icmp any host 228.28.28.28 echo (5 matches)
    20 permit icmp any any reflect ICMP (10 matches)
    30 permit ip any any (6 matches)
```


Task 7.1 Solution

🕮 Before You Begin

RMON has been introduced in Lab 1. This scenario uses absolute sampling, though. For basic introduction to RMON features check VOL1 scenario **System Management > RMON Alarms**.

R5 and R6:

```
rmon event 1 trap IETRAP description "Five Minute CPU Average Above 75%"
rmon event 2 trap IETRAP description "Five Minute CPU Average Below 40%"
rmon alarm 1 lsystem.58.0 60 absolute rising-threshold 75 1 falling threshold
40 2
!
snmp-server host 132.1.33.100 IETRAP
```

Task 7.1 Breakdown

The “absolute” sampling method is selected in this case because the value monitored is changing in a limited range and may go either up or down. Notice that we may use other IOS features to reach the task’s objective, e.g. CPU thresholds or EEM, but the task explicitly requires RMON.

Task 7.1 Verification

Verify RMON configuration using the following commands:

```
Rack1R6#show rmon alarms
Alarm 1 is active, owned by config
 Monitors lsystem.58.0 every 60 second(s)
 Taking absolute samples, last value was 0
 Rising threshold is 75, assigned to event 1
 Falling threshold is 40, assigned to event 2
 On startup enable rising or falling alarm

Rack1R6#show rmon events
Event 1 is active, owned by config
 Description is Five Minute CPU Average Above 75%
 Event firing causes trap to community IETRAP,
 last event fired at  0y0w0d,00:00:00,
 Current uptime       0y0w0d,18:12:47
Event 2 is active, owned by config
 Description is Five Minute CPU Average Below 40%
 Event firing causes trap to community IETRAP,
 last event fired at  0y0w0d,18:12:04,
 Current uptime       0y0w0d,18:12:47
```

Task 7.2 Solution

🕮 Before You Begin

To learn the necessary information about terminal line timers you may want to practice VOL1 scenario **System Management > Terminal Line Settings**.

R4:

```
username NOC password 0 CISCO
!
line vty 0 807
 exec-timeout 5 0
 logout-warning 60
 absolute-timeout 15
 login local
```


Task 7.2 Breakdown

The task configuration requires good understanding of the terminal line timers. In this case three of them are being used: the exec-timeout, which closes exec-shell after N minutes of inactivity, the absolute timeout which closes the session no matter what after a given amount of time expires and the logout-warning messages that is displayed on the terminal screen X seconds before user is logged out.

Make sure to apply policies to all VTY lines available in the router. Common mistake is applying configuration to the first 5 (0-4) or 16 (0-15) lines.

Task 7.3 Solution

R4:

```
no username NOC password CISCO
username NOC secret CISCO
```

Task 7.3 Breakdown

IOS supports both “scrambled” (type 7) and “hashed” (secret) password. The first type is normally needed for some authentication methods, such as CHAP. The secret passwords are perfect to protect authentication information for remote users, as they store it using MD5 hash.

🕮 Further Learning

For better understanding of IOS encryption methods you may read the Cisco technology note named “Cisco IOS Password Encryption Facts”

http://www.cisco.com/en/US/tech/tk59/technologies_tech_note09186a00809d38a7.shtml

Task 7.4 Solution

🕮 Before You Begin

For general information on the syslog feature found in Cisco IOS you may reference VOL1’s scenarios named

System Management > System Logging
System Management > Syslog Logging

R3:

```
interface Serial1/0
 no logging event link-status
 logging event dlci-status-change
!
logging 132.1.33.100
logging trap debugging
```


Task 7.4 Breakdown

This task may require you to know some commands that you may not hear about before. Normally, syslog registers some interface events and you may quickly find a list by going under the interface and issuing: "**logging ?**". This should be the hint to find the related information and figure out about possible options you may configure. If you really get stuck with this type of tasks, simply leave them alone and go further in the lab.

🕮 Deep Dive

- How would you explain the difference in the purpose of SNMP and Syslog?

Task 7.4 Verification

To verify the logging configuration, use the **show logging** exec command.

```
Rack1R3#show logging
Syslog logging: enabled (0 messages dropped, 1 messages rate-limited, 0
flushes, 0 overruns, xml disabled)
    Console logging: level debugging, 26 messages logged, xml disabled
    Monitor logging: level debugging, 0 messages logged, xml disabled
    Buffer logging: disabled, xml disabled
    Logging Exception size (4096 bytes)
    Count and timestamp logging messages: disabled
    Trap logging: level debugging, 31 message lines logged
        Logging to 132.1.33.100, 2 message lines logged, xml disabled
```


Task 7.5 Solution

SW1, SW2, SW3, SW4:

```
no setup express
```

Task 7.5 Breakdown

Express Setup feature allows you configuring the switch using a Web-based utility. Normally, you enable this feature on the switches with empty startup-configuration by pressing and holding the Mode button for 3 seconds after the switch boots. Normally you should see all signal LEDs turn green to signal that the feature is enabled. You may then connect your PC/laptop, assign an IP address of 10.0.0.2 and access the switch on the IP 10.0.0.1.

If the switch already has configuration, the LED will blink signaling you that the Express Setup cannot be started. However, if you hold the Mode button for 7 seconds or more in total, all existing configuration will be erased and Express Setup started. This is a good thing to recover a switch, but poses serious security threat if the switch becomes physically accessible for an intruder. Thus, you may want to disable this feature in running configuration using the command above.

🕮 Further Learning

For more information about this feature refer to the following article at Cisco's website: "Using Express Setup on a Catalyst 2950 Series Switch for Initial Installation"

http://www.cisco.com/en/US/products/hw/switches/ps628/products_configuration_example09186a00801cac4c.shtml

Task 7.5 Verification

```
RSRack1SW1#show setup express
express setup mode is not active
```


Task 8.1 Solution

🕮 Before You Begin

The task uses access-list based classification and CBWFQ weight tuning. For introduction to this concepts you may check the following VOL1 scenarios:

QoS > MQC Classification and Marking
QoS > MQC Bandwidth Reservations and CBWFQ

R3:

```
!
! Packets from the mail server
!
ip access-list extended SMTP_FROM_SERVER
 permit tcp host 132.1.3.100 eq smtp any
!
! Match packets from the server
!
class-map match-all SMTP_FROM_SERVER
 match access-group name SMTP_FROM_SERVER
!
policy-map CBWFQ
 class SMTP_FROM_SERVER
  bandwidth 256
!
interface Serial1/1
 bandwidth 512
 service-policy output CBWFQ
```

R5:

```
!
! Packets to the mail server
!
ip access-list extended SMTP_TO_SERVER
 permit tcp any host 132.1.3.100 eq smtp
!
! Match the access-list for classification
!
class-map match-all SMTP_TO_SERVER
 match access-group name SMTP_TO_SERVER
!
policy-map CBWFQ
 class SMTP_TO_SERVER
  bandwidth 256
!
interface Serial0/0/0
 bandwidth 512
 service-policy output CBWFQ
```


Task 8.1 Breakdown

This is a task that requires you applying CBWFQ queueing to a given traffic flow. The key thing is to properly match the scheduled packets using access lists on R3 and R5. R5 needs to match packets TO the server (e.g. from any host to the SMTP port) and R3 needs to match the packets FROM the server (server's SMTP port to any destination). Of course, you need to know SMTP port either by name or by protocol and port numerical value of TCP 25.

🕮 Deep Dive

- What other classification options could you use to match SMTP traffic?
- What other additional option you may use for SMTP traffic to improve performance characteristics under congestion?

Task 8.1 Verification

Verify that the policy-map is configured, applied, and working.To do this, first simulate SMTP traffic from R5:

```
Rack1R5#telnet 132.1.3.100 25 /source-interface Fa0/1
Trying 132.1.3.100, 25 ...

Check out policy-map status:

Rack1R5#show policy-map interface s0/0/0

 Serial0/0/0

  Service-policy output: CBWFQ

    Class-map: SMTP_TO_SERVER (match-all)
      4 packets, 192 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name SMTP_TO_SERVER
      Queueing
        Output Queue: Conversation 137
        Bandwidth 256 (kbps) Max Threshold 64 (packets)
        (pkts matched/bytes matched) 4/192
        (depth/total drops/no-buffer drops) 0/0/0

    Class-map: class-default (match-any)
      51 packets, 3292 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```


Task 8.2 Solution

🕮 Before You Begin

Policy routing allows for taffic engineering in order to optimize network resource utilization. You may want to practice VOL1 scenario **`IP Routing > Policy Routing`** to get a grasp of basic policy routing features.

R2:

```
!
! FTP traffic FROM VLAN 26
!
ip access-list extended FTP_FROM_VLAN26
 permit tcp 132.1.26.0 0.0.0.255 host 132.1.33.33 eq ftp
 permit tcp 132.1.26.0 0.0.0.255 host 132.1.33.33 eq ftp-data

!
! Change the next-hop for the matching traffic
!
route-map POLICY-ROUTE permit 10
 match ip address FTP_FROM_VLAN26
 set ip next-hop 132.1.23.3

!
! Policy route applied to the interface
!
interface FastEthernet0/0
 ip policy route-map POLICY-ROUTE
```

R3:

```
!
ip access-list extended FTP_FROM_SERVER
 permit tcp host 132.1.33.33 eq ftp 132.1.26.0 0.0.0.255
 permit tcp host 132.1.33.33 eq ftp-data 132.1.26.0 0.0.0.255
!
route-map POLICY-ROUTE permit 10
 match ip address FTP_FROM_SERVER
 set ip next-hop 132.1.23.2
!
interface FastEthernet0/1
 ip policy route-map POLICY-ROUTE
```


Task 8.2 Breakdown

The task requires to forcefully re-route traffic across the Serial link. If you inspect routing tables you will notice that R3 already prefers to use the Serial link to reach VLAN26 as it's the native EIGRP prefix. R2 prefer routing to VLAN 33 over the Frame-Relay link. The solution applies the policy routing to both R2 and R3, diverting FTP packets in both direction. Even though it not really needed on R3, the solution requires that this decision does not depend on IGP routing information.

Notice the way FTP packets are matched. The task mentions that FTP Passive mode is not enabled in the server and thus FTP active mode is in use. This means the solution should match FTP data and control ports which are 20 and 21 respectively. Pay attention to the directions used in the access-list – it's the same logic that was used in the previous scenario.

🕮Deep Dive

- What are the limitations for using PBR for traffic engineering?
- What other features beside routing does PBR support?
- How could you avoid matching traffic with access-lists in the core for classification with PBR?

Task 8.2 Verification

Simulate a TCP connection on port 21, emulation FTP control connection

```
Rack1R6#telnet 132.1.33.33 21
Trying 132.1.33.33, 21 ...

Rack1R2#show route-map
route-map POLICY-ROUTE, permit, sequence 10
  Match clauses:
    ip address (access-lists): FTP_FROM_VLAN26
  Set clauses:
    ip next-hop 132.1.23.3
  Policy routing matches: 2 packets, 120 bytes
```


Task 8.3 Solution

🕮 Before You Begin

This task augments the previous one, by adding CBWFQ bandwidth reservation. As usual, more information could be found in VOL1's scenario **QoS > MQC Badwidth Reservations and CBWFQ**.

R2:

```
!
! Classify FTP traffic using existing access-lists
!
class-map match-all FTP_FROM_VLAN26
  match access-group name FTP_FROM_VLAN26
!
policy-map RESERVE_FTP
 class FTP_FROM_VLAN26
  bandwidth 256
!
! Apply the CBWFQ policy
!
interface Serial0/1
 bandwidth 1536
 service-policy output RESERVE_FTP
```

R3:

```
class-map match-all FTP_FROM_SERVER
  match access-group name FTP_FROM_SERVER
!
policy-map RESERVE_FTP
  class FTP_FROM_SERVER
   bandwidth 256
!
interface Serial1/3
 bandwidth 1536
 service-policy output RESERVE_FTP
```


Task 8.3 Breakdown

This task is logical continuation of the previous one and reserves bandwidth for FTP session in both directions. Even though FTP transfers are asymmetric, the task still requires reserving 256K symmetrically. The same access-lists that were used for policy-routing are used for classification in this task. Notice that bandwidth values on the interfaces are set to T1 1536K per the task requirement.

Task 8.3 Verification

Verify the policy-map configuration. Notice the matched due to traffic generated using the telnet command previously (verification was actually applied after both scenarios have been configured).

```
Rack1R2#show policy-map interface serial 0/1
 Serial0/1

  Service-policy output: RESERVE_FTP

    Class-map: FTP_FROM_VLAN26 (match-all)
      2 packets, 96 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name FTP_FROM_VLAN26
      Queueing
        Output Queue: Conversation 265
        Bandwidth 256 (kbps)Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0

    Class-map: class-default (match-any)
      2 packets, 136 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

Rack1R3#show policy-map interface s1/3
 Serial1/3

  Service-policy output: RESERVE_FTP

    Class-map: FTP_FROM_SERVER (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name FTP_FROM_SERVER
      Queueing
        Output Queue: Conversation 265
        Bandwidth 256 (kbps) Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0

    Class-map: class-default (match-any)
      2 packets, 88 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```


Task 8.4 Solution

🕮 Before You Begin

You may find the preliminary information about FRTS in VOL1 scenarios named **QoS > Legacy Frame Relay Traffic Shaping**

R2:

```
!
! Map class for DLCI 204 only
!
map-class frame-relay DLCI_204
 frame-relay cir 128000
 frame-relay bc 1280
!
! Map class used for all other DLCIs.
!
map-class frame-relay REMAINING_BW
 frame-relay cir 192000
 frame-relay bc 24000

!
! Apply map classes to the physical interface
!
interface Serial0/0
 frame-relay traffic-shaping
 frame-relay class REMAINING_BW
 frame-relay interface-dlci 204
  class DLCI_204
```

R4:

```
interface Serial0/0/0
 frame-relay class REMAINING_BW
 frame-relay traffic-shaping
 frame-relay interface-dlci 402
  class DLCI_402
!
map-class frame-relay DLCI_402
 frame-relay cir 128000
 frame-relay bc 1280
!
map-class frame-relay REMAINING_BW
 frame-relay cir 192000
 frame-relay bc 24000
```


Task 8.4 Breakdown

The task is about figuring out the Frame-Relay traffic-shaping parameters. First, notice the port speeds on R2 and R4 – those are 512Kbps. We can quickly find that the PVC between R2 and R4 has provisioned rate of 128Kbps in both directions. The task further says that R2 and R4 should use the lowest Tc possible for the PVC between them, which is Tc=10ms. This gives us enough information to figure out the shaping rate for DLCIs 402 and 204:

CIR =128Kbps, Bc = 128000*10/1000 = 12880

Now for the other DLCIs. The task says that every remaining DLCI should have the CIR equal to the half ot 512Kbps (port speed) – 128Kbps (PVC b/w R2 and R4) that is: (512 – 128)/2 = 192Kbps. The task further says to use the default Tc = 125Ms for those remaining DLCIs, which gives us the Bc = 192000*125/1000 = 24000 bits. Notice that in both cases Be remains the default of zero and the task does not requires to implement extended bursting functionality.

Next, the solution creates two Frame-Relay map-classes on R2 and R4. The first map-class is for the PVC between R2 and R4 - that is for the DLCI 204 and 402. Another map-class – named REMAINING_BW - for all remaining DLCIs. Further, the map-class REMAINING_BW is applied to the physical interface, which means all DLCIs by default inherit it. And finally, the map-class for DLCIs 204 and 402 applies to the specific DLCIs only.

Notice that the resulting sum of all PVCs' bandwidth would exceed the interface physical rate, so the interface is effectively oversubscribed. This is not a problem, since we are dealing with a lab scenario, not a real-world configuration.

🕮 Deep Dive

- If the SP provider gives you information about CIR and PIR rates, how would these translate to your legacy FRTS parameters?

Task 8.4 Verification

Verify the FRTS parameters:

```
Rack1R2#show traffic-shape

Interface   Se0/0
       Access Target     Byte   Sustain   Excess    Interval  Increment Adapt
VC     List   Rate       Limit  bits/int  bits/int  (ms)      (bytes)   Active
213           192000     3000   24000     0         125       3000      -
201           192000     3000   24000     0         125       3000      -
203           192000     3000   24000     0         125       3000      -
204           128000     160    1280      0         10        160       -
205           192000     3000   24000     0         125       3000      -

Rack1R4#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
413           192000 3000   24000     0         125      3000      -
401           192000 3000   24000     0         125      3000      -
402           128000 160    1280      0         10       160       -
403           192000 3000   24000     0         125      3000      -
405           192000 3000   24000     0         125      3000      -
```


VOL1 Scenario Reference

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

Bridging and Switching > Layer 2 Etherchannel with LACP
Security > AAA Authentication Lists
Security > AAA Exec Authorization
Bridging and Switching > PPP AAA Authentication
OSPF > Network Point to Multipoint
OSPF > Network Point to Multipoint Non-Broadcast
EIGRP > EIGRP Unicast Updates
EIGRP > EIGRP Summarization
RIPv2 > RIPv2 Offset List.
BGP > Authenticating BGP Peering
BGP > BGP Local AS
BGP> BGP TTL Security
BGP > BGP Local-AS/Replace-AS Dual-AS
BGP > BGP Regular Expressions
BGP > BGP Aggregation
BGP > BGP Aggregation - Summary-Only
BGP > BGP Filtering With Prefix-Lists
BGP > BGP Next-Hop Trigger
IPv6 > IPv6 Link Local Addressing
IPv6 > IPv6 Unique Local Addressing
MPLS VPN > MPLS LDP
MPLS VPN> AToM
Multicast > PIM Sparse Mode
Multicast > PIM NBMA Mode
Multicast > RPF Failure
Multicast > PIM NMBA Mode
IP Services > Proxy ARP
IP Services > DHCP
Security > Zone Based Firewall
System Management > SNMPv2 Server
System Management < SNMPv2 Access Control
Security > Traffic Filtering using Reflexive Access-Lists
System Management > RMON Alarms
System Management > Terminal Line Settings
System Management > System Logging
System Management > Syslog Logging
QoS > MQC Classification and Marking
QoS > MQC Bandwidth Reservations and CBWFQ
IP Routing > Policy Routing
QoS > Legacy Frame Relay Traffic Shaping

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

Deep Dive Answers

- How STP interacts with Layer 2 Ether-Channels?
  - STP treats the port-channel as a single link. This is accomplished by sending the STP BPDUs only across one of the physical links in the bundle. The result is that all links could be used for traffic forwarding.
- What happens if one end of the eherchannel unbinds while other remains bonded?
  - This is a very dangerous condition that may happen sometimes with dynamic port-channel negotiation protocols. The net result is layer 2 loop, as the switch that has ports unbound may send back the packets received from the end that treats the bundle as a single link.
- What is the main idea of EAP protocol used with 802.1X?
  - EAP or Extensible Authentication Protocol allows for the authenticator device (e.g.) to forward/tunnel authentication requests to the authentication server, bypassing any local authentication. The result is flexible and extendible authentication scheme, where switch functionality is kept simple.
- How RADIUS performs authorization if it does not have explicit authorization page?
  - Unlike TACACS+, RADIUS does not support explicitly authorization request. The authenticator device deduces the authorization attributes from the authentication response, returned by the server.
- What is one special benefit of P2MP network type, not available with other OSPF networks?
  - You may flexibly change OSPF costs to reach the peers. This is being a direct consequence of the fact that network is being treated as collection of point-to-point links.
- From routing perspective, what is the drawback of using P2MP network type?
  - The amount of Link-State database grows significantly as does the routing table, since every remote node injects a /32 route for its endpoint. This condition could be alleviated by letting the remote nodes in a hub-and-spoke topology using the point-to-point link type.
- Why would you need to use unicast EIGRP neighbors?
  - In the rare case if the underlying network is not supporting multicasting. Aside from that, the other situation could be hub-and-spoke topology, where spokes need to exchange routing updates directly. The TTL in unicast updates is 2, as opposed to 1 in the multicast updates. This allows for the updates to be exchanged across the hub.

- Why summarization is so important in EIGRP?
  - Summarization bound EIGRP query propagation and directly affects EIGRP scalability. It is even more important than in link-state protocols, as it helps preventing SIA conditions in EIGRP.
- Why is it important to keep maximum hop count for RIP low?
  - Because of counting to infinity effect. Under some condition, routing loops may form in RIP, and the only way to stop them is by reaching the infinite metric in circulating updates. The lower is the infinity (maximum hop count) the faster loop is eliminated.
- What other protocol you know that uses TCP MD5 option for authentication?
  - Any TCP-based protocol may use the authentication option. Within the scope of the exam this is the LDP protocol, used for label distribution.
- How could firewalls react on the presence of TCP MD5 option?
  - Some firewalls perform TCP normalization and may drop packets with unknown options. For example, ASA firewall by default drop packets with TCP Option 19, which is used for MD5 authentication.
- What is the purpose of the **`replace-as`** option to **`local-as .. no-prepend`** command?
  - It allows to completely remove the AS# number used by the local BGP speaker (router bgp # command). In result, the local AS# will be completely hidden and the outside systems will only see the Local-AS number.
- How could you implement the same “non-transit” filtering in scalable manner without using AS_PATH matching?
  - One option would be applying BGP community tagging on the routes received from AS 254 and filtering based on the community value egress to the downstream AS.
- What other aggregation methods could be used with BGP, besides the **`aggregate-address`** command?
  - Normally, the recommended way would be using static local routes pointing to Null0 and advertising them into BGP. This would ensure that packets destined to the non-existent local networks covered by the summary are being dropped.
- What could be a disadvantage of BGP table aggregation?
  - Suboptimal connectivity is the normal result of aggregation. Additionally, aggregation results in slower convergence. However, on the scale of BGP networks, stability and scalability is much more important than convergence and this is why summarization is actively employed.

- Why it could be important to filter summary from being advertised to internal peers?
    - Since BGP does not generate a discard route, the summary prefix, may attract traffic to non-existent local networks which in turn may result in routing loops. Therefore, it is normally not recommended exposing the summary to internal BGP peers.
- What problems you may see using L2 VPN solutions?
    - The major problem, just like with any tunneling solution, is the MTU issue. TCP applications may refuse to work across the tunnel under some conditions, unless you enable traffic fragmentation (which is a bad idea) or set up TCP MSS adjustment somewhere (which is hard to do with L2 VPNs, as it needs to be applied at layer 3/4). Alternatively, you may enable PathMTU discovery with the tunnels, but this solution may not always work due to ICMP filtering in the networks.
- Why P2P L2 VPNs are so effective for hardware implementation?
    - Since there are just two endpoints, forwarding packet is obviously easy, as it only requires applying the tunnel encapsulation headers (e.g. AToM labels)
- Can you use L2 VPN P2P tunnels to create multipoint connectivity?
    - P2P tunnels could be groomed to an Ethernet switch, if they are used to connect Ethernet sites. If the P2P links are connected to a switch in hub-and-spoke or ring manner, it is possible to achieve multipoint connectivity, by means of external Ethernet switching.
- What could be the benefits of using Static RP?
    - There is no configuration overhead and you can be sure about configuration consistency. Additionally, using static RP adds more security, provided that you disabled static RP preemption by AutoRP.
- Why should you use the command **`ip igmp join-group`** with caution?
    - Because it enables process-switching for received multicast packets and may put high load on the router's CPU.
- How PIM interprets the NBMA segments by default?
    - From the standpoint of PIM protocol, an NBMA segment is a cloud capable of multicasting. PIM is agnostic of the actual multicast propagation topology. Some Layer 2 technology supports native multicast, like Ethernet or multicast extension for Frame-Relay. However, the latter hasn't been widely implemented and therefore multicasting over complex Frame-Relay mesh clouds is inefficient.

- What could be an alternative to using PIM NBMA mode?
    - Since PIM NBMA mode treats the segments as collections of point-to-point links, an alternative would be using point-to-point subinterfaces as opposed to multipoint interfaces.
- What is the possible drawback of PIM NBMA mode in large mesh topologies?
    - The only drawback could be in the situations where layer 2 technology supports optimum multicast replication, e.g. in some VPLS extensions. In this situation, PIM NBMA mode will rely on edge replication, without utilizing the core multicast.
- What could be the use of Proxy ARP?
    - For example transparently gluing two discrepant IP subnets. The hosts may have the /24 subnet masks while the routers may use /25 for example. The routers will intercept the ARP requests and respond with their own MAC addresses, transparently forwarding packets.
- Why proxy ARP could be dangerous?
    - A malicious host may respond to ARP requests and change the traffic flow to intercept the other uses sessions. Using DHCP snooping with Dynamic ARP inspection is recommended to enforce ARP security.
- How could an attacker use IP source routing?
    - Source route allows for taking arbitrary path in the network (as much as physical constraints allow) and therefore bypassing security devices or presenting a packet as coming from different interface. In general, network security dictates that an end-user is not allowed to affect the routing paths.
- What good use you could think of for IP source routing?
    - IP source route could be used for effective traffic engineering, but creating source-routed tunnels in the network core. The drawback is the unconstrained IP header growth and forwarding complexity, which is hard to implement in hardware.
- What main benefits of ZFW you may name over CBAC?
    - Modular command syntax allows for more readable configurations and better code reuse, e.g. you may reuse the same classes and policies for different zones. Also, the policy no longer applies to interface (physical entities) but rather to logical security zones.

- How is zone-based policing different from MQC policing?
  - Zone based Firewall traffic policing applies to inter-zone traffic, not the traffic leaving an interface. Therefore, it allows for some bandwidth aggregation and more flexible policing.
- Can you report syslog messages via SNMP?
  - Yes, by enabling syslog history buffer for SNMP you can re-translate syslog messages as SNMP traps.
- Why is it so important to use access-lists with SNMPv2c?
  - The only security option supported with SNMPv2c is clear-text community based authentication. Therefore, restricting access based on management station IP addresses becomes important.
- Name the SNMP ports and their use.
  - SNMP ports are UDP 160 and UDP 161 for SNMP management requests and traps respectively. Even though SNMP was designed to support TCP as well, UDP was selected for its light weight and simplicity.
- How would you explain the difference in the purpose of SNMP and Syslog?
  - SNMP is a management tool, which goal is monitoring systems variables and threshold. Syslog is event tracking tool, which allows for various application to report their events. SNMP monitoring is mostly synchronous, driven by operator initiated polling, while Syslog is purely asynchronous, not depending on any human intervention.
- What other classification options could you use to match SMTP traffic?
  - Using NBAR or port number matching would work as well. Matching an access-list is universal classification criterion, though it does not allow for too much flexibility, like stateful or deep packet inspection.
- What other additional CBWFQ option you may use for SMTP traffic to improve performance characteristics under congestion?
  - Enabling RED (Random Early Detection) should help in improving bandwidth utilization for bursty and heavy TCP flows.
- What are the limitations for using PBR for traffic engineering?
  - PBR is not scalable as it has to be configured on every node where change of routing decision is required. Additionally, it may require multi-field classification criteria in the core, e.g. matching access-lists, which puts extra burden on the core routers.
- What other features besides routing does PBR support?
  - PBR could be used to drop packets by setting the output interface to Null0. This was a popular use for PBR before MQC introduced the “drop” function.

- How could you avoid matching traffic with access-lists in the core for classification with PBR?
    - It is not entirely possibly to do that, as using access-lists is the main classification criterion for PBR. However, it is possible to match just the DSCP or IP precedence value in the packet header using the same access-lists thus simplifying the classification. At the same time, the edge routers may perform classification and apply DSCP marking to the packets entering the core.
- If the SP provider gives you information about CIR and PIR rates, how would these translate to your legacy FRTS parameters?
    - With FRTS, the PIR (Peak Rate) value that the SP provides translates to the **`frame-relay cir`** while CIR (committed information rate) translates to **`frame-relay mincir`** setting under the **`map-class frame-relay`** command.

Lab 3 Solutions

Task 1.1 Solutions

🕮 Before You Begin

This scenario uses Intergrated Routed Bridging technology, which is explained in the following article:

http://www.cisco.com/en/US/tech/tk331/tk660/technologies_tech_note09186a0080094471.shtml

Some of the information in the document could be outdated but still helpful for understanding or IRB.

R6:

```
bridge irb
!
interface FastEthernet0/0.16
 bridge-group 1
!
interface FastEthernet0/0.36
 bridge-group 1
!
interface BVI1
 ip address 136.1.136.6 255.255.255.0
!
bridge 1 protocol ieee
bridge 1 route ip
```

Task 1.1 Breakdown

By default, Cisco routers will route IP and bridge all other protocols on all interfaces. Additionally, a protocol can be either routed or bridged, but not both. By using either the concurrent routing and bridging (CRB) or integrated routing and bridging (IRB) features, this limitation can be overcome.

With CRB, a protocol can be routed on one interface while being bridged on another interface. When CRB is used, traffic in the routed domain cannot be passed on to the bridge domain. With IRB, a protocol can be both routed and bridged on the same interface. Therefore traffic from the routed domain *can* be passed on to the bridge domain.

These features are useful when you want to extend the broadcast domain for one protocol, while maintaining it for another. For example, IPX can be bridged between two LAN segments, while IP is routed on those interfaces (CRB). Additionally, a bridge virtual interface (BVI) can be configured with an IPX address so that other segments running IPX routing can communicate with the IPX bridged network (IRB). CRB is considered a legacy feature since IRB inherits all functionality of CRB, with the addition of the BVI.

In order to configure the IOS router as a bridge, you should first define a type of bridging configured in the firewall. This could be either CRB (Concurrent Routing and Bridging) or IRB (Integrated Routing and Bridging). Both use the concept of a bridge group, which is essentially a virtual bridge inside the router, with its own MAC address table.

- CRB mode is enabled using the command **`bridge crb`**. Allows the router to bridge frames between interfaces configured as members of a single bridge group. Interfaces not in any bridge group with IP addresses assigned may accept and forward IP packets. Essentially, the router acts as a bridge for one set of interfaces and L3 router for the rest of the interfaces.

- IRB mode is enabled using the command **`bridge irb`**. allows configuring a special BVI (Bridge Virtual Interface) for every bridge group. This interface represents the router as L3 device within the respective bridge group. All devices in the group aware of this BVI's MAC address may send IP packets to the router and communicate with other L2 domains. This mode allows the router to bridge between the interfaces and route packets for the members of the bridge-group that are aware of this service. You may compare BVI with an SVI (Switch Virtual Interface, e.g. VLAN 31) in a Layer 3 switch. In addition to the **`bridge irb`** command you need to add the command **`bridge <number> route ip`** where <number> is the bridge-group number. Otherwise, the BVI interface will not route the IP packets.

- In order to configure an interface/subinterface as member of a bridge group, use the command **bridge-group <number>** in interface configuration mode. By default, the new group starts with Spanning-Tree protocol disabled. You may need the command **bridge <n> protocol IEEE** to enable the IEEE-compatible spanning tree for the bridge-group. Notice that for 802.1Q subinterfaces the firewall will run the PVST+ variant of STP.

The first step in bridging is to create a transparent bridge group. This is accomplished by issuing the global configuration command **bridge [num] protocol ieee**. The *ieee* option specifies that IEEE spanning-tree will be enabled for the bridge group. To apply the bridge-group, use the interface command **bridge-group [num]**, where *num* is the bridge group previously created.

Since **ip routing** is enabled by default, the above configuration will only enable transparent bridging for non-IP protocols. To enable the integrated routing and bridging process, use the global configuration command **bridge irb**. Next, choose which protocols you want to route and bridge for the bridge group. This is accomplished by issuing the **bridge [num] route [protocol]**. In the above case, IP is both routed and bridged for bridge group 1. Lastly, the BVI is created by issuing the **interface bvi [num]**, where *num* is the bridge group number. All traffic that passes from the bridge domain to the routed domain and vice versa must pass through the BVI. This is the interface where logical configuration is placed, such as an IP address.

Task 1.1 Verification

Verify the IRB configuration on R6:

```
Rack1R6#show interface irb | begin FastEthernet0/0
FastEthernet0/0

 Not bridging this sub-interface.

FastEthernet0/0.16

 Routed protocols on FastEthernet0/0.16:
  ip

 Bridged protocols on FastEthernet0/0.16:
  appletalk  clns       decnet     ip

 <output omitted>

FastEthernet0/0.36

 Routed protocols on FastEthernet0/0.36:
  ip

 Bridged protocols on FastEthernet0/0.36:
  appletalk  clns       decnet     ip

<output omitted>
```

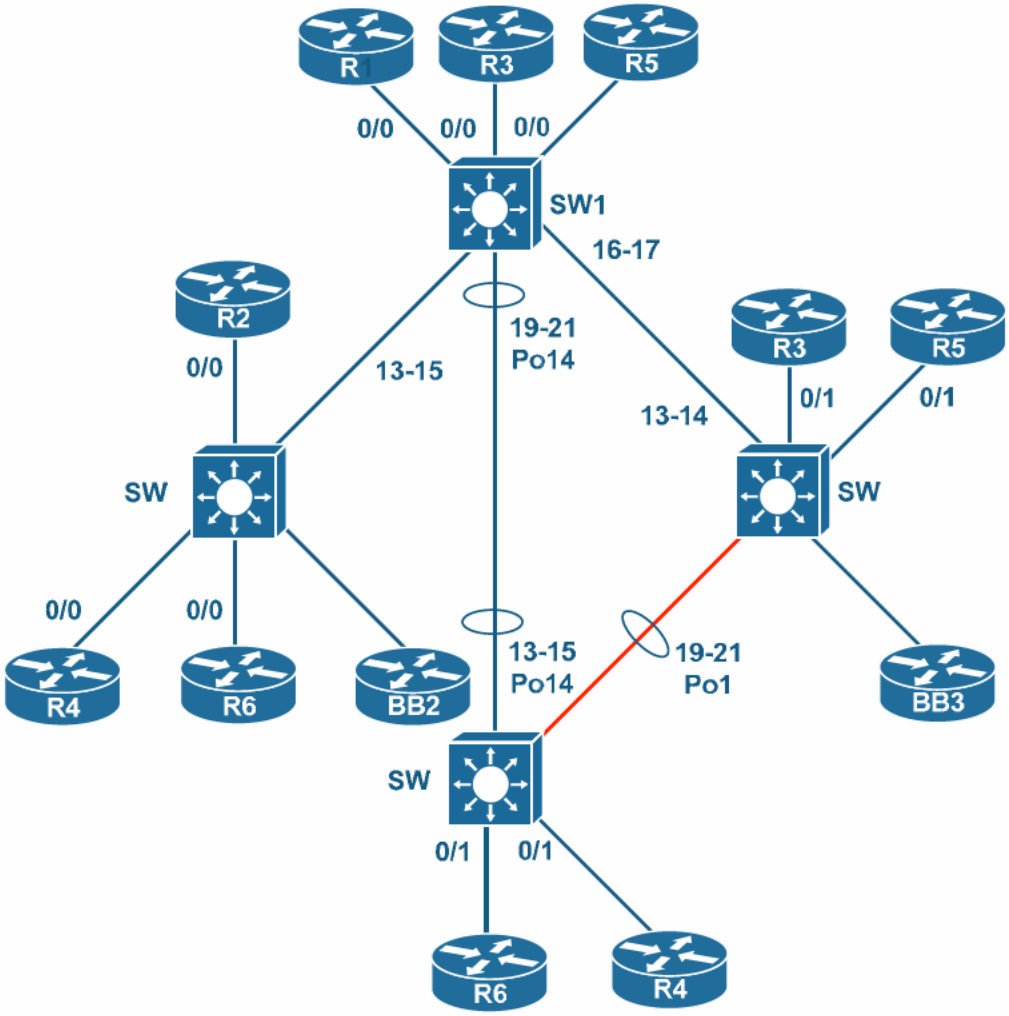


```
Rack1R6#ping 136.1.136.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 136.1.136.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1R6#ping 136.1.136.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 136.1.136.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1R3#ping 136.1.136.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 136.1.136.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms
```


Task 1.2 Solutions

🕮 Before You Begin

This scenario requires your understanding of existing Layer 2 topology. This topology could be discovered using the commands **`show interfaces status`**, **`show interfaces trunk`** and **`show etherchannel summary`** in the switches. The following the the Layer 2 diagram built for this lab (red line is Layer 3 port-channel, which has no STP enabled).



The solution modifes STP priorities and the root bridge in this lab. If you need more fundamental practice with this technologies, check VOL1's scenario named **`Bridging and Switching > STP Load Balancing with Port Priority`** and **`Bridging and Switching > STP Root Bridge Election`**.


```
SW1:
spanning-tree vlan 4,44,52,63 root primary
!
interface FastEthernet0/14
 spanning-tree vlan 4,44,52,63 port-priority 32
!
interface FastEthernet0/15
 spanning-tree vlan 4,44,52,63 port-priority 16

SW2:
spanning-tree uplinkfast
```

Task 1.2 Breakdown

To influence which port is elected the root port, the two user configurable values to change are port **cost** and port **priority**. Changing port cost will affect both the local bridge and all downstream bridges. Changing port priority will only affect the directly connected downstream bridge. Keep in mind that port priority is only taken into account if there is a tie in both cost and bridge-ID (a tie in bridge-ID implies that a bridge has multiple connections to the same upstream bridge).

For this task, port-priority is changed on the root bridge (SW1) in order to influence how the downstream bridge (SW2) elects its root port.

Spanning-tree uplinkfast provides fast reconvergence in the event of a direct failure of the root port. During the initial root port election, a bridge running uplinkfast notes which ports can be used as alternate paths to the root bridge. When the root port fails, the alternate port immediately comes out of blocking state and transitions to forwarding. Also, to ensure convergence of the upstream CAM table, all known MAC addresses are flooded out the new root port as dummy multicast frames. This process typically takes three to five seconds, and reduces convergence time considerably. Uplinkfast is only supported when running PVST+. To configure uplinkfast, use the global configuration command **spanning-tree uplinkfast**. This feature is incorporated in the newer RSTP (Rapding Spanning Tree Protocol) natively.

🕮 Further Learning

To get better understanding of STP/RSP convergence mechanics you may want to read the following post on the INE's blog:

Notice that it requires you being familiar with STP and RSTP concepts, which are covered in the following Cisco's publication and Wikipedia article:

"Understanding RSTP"
http://www.cisco.com/en/US/tech/tk389/tk621/technologies_white_paper09186a0080094cfa.shtml

and

"Spanning Tree Protocol"
http://en.wikipedia.org/wiki/Spanning_tree_protocol

🕮 Deep Dive

- How can you make STP ignore the cost of a certain link, or segment connecting multiple switches?

Task 1.2 Verification

Check any spanning-tree instance for active VLANs, e.g. VLAN44 and confirm that SW2 prefers port number 9 over 7 and 8 for the root port.

```
Rack1SW2#show spanning-tree vlan 44

VLAN0044
  Spanning tree enabled protocol ieee
  Root ID     Priority    24592
              Address     0019.55e6.6580
              Cost        19 ← cost to root
              Port        17 (FastEthernet0/15) ← root port
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32812  (priority 32768 sys-id-ext 44)
              Address     0016.9d31.8380
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/13           Altn BLK 19  ← tie  128.15   P2p
Fa0/14           Altn BLK 19  ← in   128.16   P2p
Fa0/15           Root FWD 19  ← cost 128.17   P2p
                   ↑
                 root port
```

```
Rack1SW2#show spanning-tree vlan 44 detail

Port 15 (FastEthernet0/13) of VLAN0044 is blocking
 <output deleted>
   Designated port id is 128.15, designated path cost 0
<output deleted>          ↑
                upstream port priority

Port 16 (FastEthernet0/14) of VLAN0044 is blocking
<output deleted>
   Designated port id is 32.16, designated path cost 0
<output deleted>          ↑
                upstream port priority

Port 17 (FastEthernet0/15) of VLAN0044 is forwarding
<output deleted>
   Designated port id is 16.17, designated path cost 0
<output deleted>          ↑
                upstream port priority
                    lowest wins
```


Verify that UplinkFast is enabled:

```
Rack1SW2#show spanning-tree uplinkfast
UplinkFast is enabled

Station update rate set to 150 packets/sec.

UplinkFast statistics
-----------------------
Number of transitions via uplinkFast (all VLANs)            : 0
Number of proxy multicast addresses transmitted (all VLANs) : 0

Name                 Interface List
-------------------- ------------------------------------
VLAN0001             Fa0/13(fwd), Fa0/14, Fa0/15
<output omitted>
VLAN0063             Fa0/15(fwd), Fa0/13, Fa0/14
```

Verify that SW1 is the root-bridge for respective VLANs: the root path cost should be zero:

```
Rack1SW1#show spanning-tree root

                                        Root    Hello Max Fwd
Vlan                 Root ID            Cost    Time  Age Dly  Root
Port
---------------- ------------------- --------- ----- --- ---  --------
VLAN0001         32769 0018.7392.0980        19    2   20  15  Fa0/16
VLAN0003         32771 0018.7392.0980        19    2   20  15  Fa0/16
VLAN0004         24580 001f.6ce6.8700         0    2   20  15
VLAN0007         32775 0018.7392.0980        19    2   20  15  Fa0/16
VLAN0016         32768 001a.2f78.4678      3038    2   20  15  Fa0/13
VLAN0029         32797 0018.7392.0980        19    2   20  15  Fa0/16
VLAN0036         32768 001a.2f78.4678      3038    2   20  15  Fa0/13
VLAN0044         24620 001f.6ce6.8700         0    2   20  15
VLAN0052         24628 001f.6ce6.8700         0    2   20  15
VLAN0057         32825 0018.7392.0980        19    2   20  15  Fa0/16
VLAN0063         24639 001f.6ce6.8700         0    2   20  15
```


Task 2.1 Solution

🕮 Before You Begin

This scenario uses OSPF network type modification, OSPF MD5 authentication, proper network type selection, OSPF virtual links and cost manipulation.. For introductory information on each of these concepts you may check the following VOL1 labs:

OSPF > Network Loopback
OSPF > Network Point-to-Multipoint
OSPF > MD5 Authentication
OSPF > Path Selection with Cost
OSPF > Repairing Discontiguous OSPF Areas with Virtual Links

R1:

```
router ospf 1
 network 150.1.1.1 0.0.0.0 area 0
!
! R4 and R5 Loopbacks should appear as /32 for MPLS VPN peering
! So we only change the network type for R1 and R2 and not R4/R5

interface Loopback0
 ip ospf network point-to-point
!
interface Serial0/0
 ip ospf message-digest-key 1 md5 CISCO
!
router ospf 1
 area 0 authentication message-digest
```


R2:

```
!
! P2MP network type allows R2 and R5 to use R5 as next-hop
! for reaching each other
!
interface Serial0/0
 ip ospf network point-to-multipoint
!
router ospf 1
 network 150.1.2.2 0.0.0.0 area 0
!
! R4 and R5 Loopbacks should appear as /32 for MPLS VPN peering
!
interface Loopback0
 ip ospf network point-to-point
!
interface Serial0/0
 ip ospf message-digest-key 1 md5 CISCO
!
router ospf 1
 area 0 authentication message-digest
```

R4:

```
interface Serial0/0/0
 ip ospf network point-to-multipoint
!
router ospf 1
 network 150.1.4.4 0.0.0.0 area 0
!
interface Serial0/1/0
 ip ospf cost 65534
!
router ospf 1
 area 45 virtual-link 150.1.5.5
 network 136.1.45.4 0.0.0.0 area 45
!
interface Serial0/0/0
 ip ospf message-digest-key 1 md5 CISCO
!
router ospf 1
 area 0 authentication message-digest
 area 45 virtual-link 150.1.5.5 message-digest-key 1 md5 CISCO
```


R5:

```
interface Serial0/0/0.245 multipoint
 ip ospf network point-to-multipoint
!
router ospf 1
 network 150.1.5.5 0.0.0.0 area 0
!
interface Serial0/1/0
 ip ospf cost 65534
!
router ospf 1
 area 45 virtual-link 150.1.4.4
 network 136.1.45.5 0.0.0.0 area 45
!
interface Serial0/0/0.15 point-to-point
 ip ospf message-digest-key 1 md5 CISCO
!
interface Serial0/0/0.245 multipoint
 ip ospf message-digest-key 1 md5 CISCO
!
router ospf 1
 area 0 authentication message-digest
 area 45 virtual-link 150.1.4.4 message-digest-key 1 md5 CISCO
```


Task 2.1 Breakdown

The scenarios calls for OSPF network type point-to-multipoint by requiring R1 and R4 to use R5 as the next-hop to reach each other. Next, the /24 prefix length requirement for loopback prefixes translates into changing OSPF network type for the Loopback interfaces. Authentication is configured routinely using MD5 message-digest keys. OSPF supports both clear text and MD5 authentication. Both of these authentication types can be applied to an OSPF area as a whole, or on an individual interface basis. When area authentication is enabled, all *adjacencies* in the area must be authenticated with the defined authentication type. In the above case, MD5 authentication is enabled in area 0. This implies that all area 0 adjacencies must authenticate using MD5, unless otherwise overridden.

For the backup scenario, the first thing to do is set up the backup link with higher OSPF cost. Be careful and make sure the total backup path cost between R4and R5 is less than 65535, otherwise the virtual-link will not work (see later). When R4 loses its connection to the Frame Relay cloud, OSPF areas 4 and 44 lose their connection to area 0. Additionally, since R4's Loopback 0 interface is advertised into OSPF area 0, area 0 becomes discontiguous when the Frame Relay connection of R4 is down. Frame Relay RFC 2328 dictates that OSPF area 0 must be contiguous throughout the OSPF domain. In addition to this requirement, all other areas must be connected to area 0. For situations where physical connectivity cannot be obtained, a *virtual-link* can be used as a logical connection to area 0.

In this particular case, we manually configured OSPF cost on the interface. Alternatively, we could have configured the interface for a lower bandwidth value, to make it less preferred.

Caution

A virtual-link *is* an interface in area 0. Therefore, all attributes of area 0 are inherited by routers attached to the virtual-link. This includes area 0 authentication and stipulations on area summarization. Remember that a router that terminates a virtual-link is an area 0 router (ABR).

Note

Interface level authentication overrides area authentication. Therefore adjacencies within an area may be configured for clear-text authentication while a specific interface in the area is configured for MD5 authentication or NULL (no) authentication.

Deep Dive

- How could virtual links be used for traffic engineering?
- What is authenticated by OSPF authentication?
- Can OSPF ensure LSA authentication?

Task 2.1 Verification

Verify the basic OSPF configuration and network types:

```
Rack1R5#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.4.4         0   FULL/  -        -           136.1.45.4      OSPF_VL0
150.1.4.4         0   FULL/  -        00:01:55    136.1.245.4     Serial0/0/0.245
150.1.2.2         0   FULL/  -        00:01:36    136.1.245.2     Serial0/0/0.245
150.1.1.1         0   FULL/  -        00:00:33    136.1.15.1      Serial0/0/0.15
150.1.4.4         0   FULL/  -        00:00:38    136.1.45.4      Serial0/1/0
7.7.7.7           1   FULL/DR        00:00:37    136.1.57.7      FastEthernet0/1

Rack1R5#show ip ospf interface serial 0/0/0.245
Serial0/0/0.245 is up, line protocol is up
  Internet Address 136.1.245.5/24, Area 0
  Process ID 1, Router ID 150.1.5.5, Network Type POINT_TO_MULTIPOINT,
Cost: 64
  Transmit Delay is 1 sec, State POINT_TO_MULTIPOINT
  Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit
5
    oob-resync timeout 120
    Hello due in 00:00:18
  Supports Link-local Signaling (LLS)
  Cisco NSF helper support enabled
  IETF NSF helper support enabled
  Index 2/2, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 3
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 2, Adjacent neighbor count is 2
    Adjacent with neighbor 150.1.4.4
    Adjacent with neighbor 150.1.2.2
  Suppress hello for 0 neighbor(s)
  Message digest authentication enabled
    Youngest key id is 1
Rack1R5#
```

Verify that R2 could reach R4 via R5, by the virtue of /32 route:

```
Rack1R2#show ip route ospf
     136.1.0.0/16 is variably subnetted, 6 subnets, 2 masks
O       136.1.245.4/32 [110/128] via 136.1.245.5, 00:04:56, Serial0/0
O       136.1.245.5/32 [110/64] via 136.1.245.5, 00:04:56, Serial0/0

Rack1R2#ping 136.1.245.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 136.1.245.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 88/90/96 ms
```



```
Rack1R2#traceroute 136.1.245.4

Type escape sequence to abort.
Tracing the route to 136.1.245.4

  1 136.1.245.5 28 msec 28 msec 32 msec
  2 136.1.245.4 73 msec *  89 msec
```

Verify Loopback advertisements. R1 and R2 should use /24 mask while R4 and R5 should be using /32 mask. Notice that all prefixes are accessible via the Frame-Relay path.

```
Rack1R5#show ip route ospf
     136.1.0.0/16 is variably subnetted, 9 subnets, 2 masks
O       136.1.245.2/32 [110/64] via 136.1.245.2, 00:28:06, Serial0/0/0.245
O       136.1.245.4/32 [110/64] via 136.1.245.4, 00:28:06, Serial0/0/0.245
O IA    136.1.4.0/24 [110/65] via 136.1.245.4, 00:28:06, Serial0/0/0.245
O IA    136.1.44.0/24 [110/65] via 136.1.245.4, 00:28:06, Serial0/0/0.245
     150.1.0.0/16 is variably subnetted, 5 subnets, 2 masks
O IA    150.1.8.8/32 [110/66] via 136.1.245.4, 00:28:06, Serial0/0/0.245
O       150.1.4.4/32 [110/65] via 136.1.245.4, 00:28:06, Serial0/0/0.245
O       150.1.1.0/24 [110/65] via 136.1.15.1, 00:27:03, Serial0/0/0.15
O       150.1.2.0/24 [110/65] via 136.1.245.2, 00:28:06, Serial0/0/0.245
Rack1R5#

Rack1R4#show ip route ospf
     136.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
O       136.1.245.2/32 [110/128] via 136.1.245.5, 00:28:45, Serial0/0/0
O       136.1.245.5/32 [110/64] via 136.1.245.5, 00:28:45, Serial0/0/0
O       136.1.15.0/24 [110/128] via 136.1.245.5, 00:28:45, Serial0/0/0
     150.1.0.0/16 is variably subnetted, 5 subnets, 2 masks
O       150.1.8.8/32 [110/2] via 136.1.44.8, 00:32:36, FastEthernet0/1
O       150.1.5.5/32 [110/65] via 136.1.245.5, 00:28:45, Serial0/0/0
O       150.1.1.0/24 [110/129] via 136.1.245.5, 00:27:15, Serial0/0/0
O       150.1.2.0/24 [110/129] via 136.1.245.5, 00:28:35, Serial0/0/0
Rack1R4#
```


Verify the virtual link between R4 and R5. Notice the virtual link cost which is a result of setting the Serial link cost to a high value.

```
Rack1R5#show ip ospf virtual-links
Virtual Link OSPF_VL0 to router 150.1.4.4 is up
  Run as demand circuit
  DoNotAge LSA allowed.
  Transit area 45, via interface Serial0/1/0, Cost of using 65534
  Transmit Delay is 1 sec, State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:02
    Adjacency State FULL (Hello suppressed)
    Index 4/5, retransmission queue length 0, number of retransmission
0
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 0, maximum is 0
    Last retransmission scan time is 0 msec, maximum is 0 msec
  Message digest authentication enabled
    Youngest key id is 1
```


Verify area 0 adjacencies authentication on R5. Since the authentication is configured and all neighbors are adjacent the configuration is correct.

```
Rack1R5#show ip ospf
 Routing Process "ospf 1" with ID 150.1.5.5
 Start time: 00:00:39.020, Time elapsed: 00:50:06.920
 Supports only single TOS(TOS0) routes
 Supports opaque LSA
 Supports Link-local Signaling (LLS)
 Supports area transit capability
 It is an area border router
 Router is not originating router-LSAs with maximum metric
 Initial SPF schedule delay 5000 msecs
 Minimum hold time between two consecutive SPFs 10000 msecs
 Maximum wait time between two consecutive SPFs 10000 msecs
 Incremental-SPF disabled
 Minimum LSA interval 5 secs
 Minimum LSA arrival 1000 msecs
 LSA group pacing timer 240 secs
 Interface flood pacing timer 33 msecs
 Retransmission pacing timer 66 msecs
 Number of external LSA 0. Checksum Sum 0x000000
 Number of opaque AS LSA 0. Checksum Sum 0x000000
 Number of DCbitless external and opaque AS LSA 0
 Number of DoNotAge external and opaque AS LSA 0
 Number of areas in this router is 2. 2 normal 0 stub 0 nssa
 Number of areas transit capable is 1
 External flood list length 0
 IETF NSF helper support enabled
 Cisco NSF helper support enabled
    Area BACKBONE(0)
        Number of interfaces in this area is 4 (1 loopback)
        Area has message digest authentication
        SPF algorithm last executed 00:43:06.284 ago
        SPF algorithm executed 17 times
        Area ranges are
        Number of LSA 9. Checksum Sum 0x069547
        Number of opaque link LSA 0. Checksum Sum 0x000000
        Number of DCbitless LSA 0
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0
    Area 45
        Number of interfaces in this area is 1
        This area has transit capability: Virtual Link Endpoint
        Area has no authentication
        SPF algorithm last executed 00:43:54.440 ago
        SPF algorithm executed 3 times
        Area ranges are
        Number of LSA 24. Checksum Sum 0x0A6989
        Number of opaque link LSA 0. Checksum Sum 0x000000
        Number of DCbitless LSA 0
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0
```


Task 2.2 Solution

🕮 Before You Begin

The first part of this scenario involves tuning OSPF auto-cost by manipulating the reference bandwidth. Additionally, the LSA pacing timers are tuned. You may read more about these features in VOL1 lab **OSPF > OSPF Global Timers** and reading the blog post "Tuning OSPF Performance" on the INE blog:

http://blog.ine.com/2009/12/31/tuning-ospf-performance/

R1, R2, R4, R5, SW1, and SW2:

```
router ospf 1
 auto-cost reference-bandwidth 20000
```

R5:

```
router ospf 1
 timers lsa arrival 2000
 timers pacing lsa-group 40
```

Note

The following part of the solution is concerned with redistribution only.

Redistribution topology has multiple points of mutual redistribution. This is a complex scenario, which requires good understanding of redistribution process. The breakdown for this scenario provides enough information to understand the solution, but we recommend reading the following post on the INE blog to get better understanding of redistribution: "Understanding Redistribution Part I, II and III":

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

http://blog.ine.com/2008/02/19/understanding-redistribution-part-ii/

Additionally, it is highly recommended completing Lab 2 prior to this lab, to get familiar with the routing loop analysis process.

R1:

```
!
! Mutually redistribute OSPF/EIGRP on R1
!
router eigrp 100
 redistribute ospf 1 metric 10000 1000 100 1 1500
!
! OSPF external AD is set above EIGRP's to make sure EIGRP routers
! don't prefer OSPF to reach to R6-redistributed prefixes
!
router ospf 1
 redistribute eigrp 100 subnets route-map EIGRP_TO_OSPF
 distance ospf external 171
!
! Tag 1 is assigned to BB1's prefixes redistributed by R6
!
route-map EIGRP_TO_OSPF permit 10
 match tag 1
 set metric 1
!
! Don't forget the trailing permit or you'll block other prefixes
!
route-map EIGRP_TO_OSPF permit 1000
```


R2:

```
!
! Mutually redistribute OSPF and EIGRP on R2 and set metric of 1
! to BB3 prefixes, redistributed by R6 with the tag of 3
!
router eigrp 100
 redistribute ospf 1 metric 10000 1000 100 1 1500
!
! OSPF external AD is set above EIGRP's to make sure EIGRP routers
! don't prefer OSPF to reach to R6-redistributed prefixes
!
router ospf 1
 redistribute eigrp 100 subnets route-map EIGRP_TO_OSPF
 distance ospf external 171
!
route-map EIGRP_TO_OSPF permit 10
 match tag 3
 set metric 1
!
route-map EIGRP_TO_OSPF permit 1000
```

R5:

```
!
! Mutually redistribute between OSPF and RIP on R5
!

router ospf 1
 redistribute rip subnets
!
! Change RIP AD to make RIP routes preferred over BGP
!
router rip
 redistribute ospf 1 metric 1
 distance 19
```


R6:

```
router eigrp 100
 redistribute rip metric 10000 1000 100 1 1500 route-map RIP_TO_EIGRP
!
router rip
  redistribute eigrp 100 metric 1

!
! Match BB1's prefixes
!
ip prefix-list RIP_FROM_BB1 seq 5 permit 212.18.0.0/22 ge 24 le 24

!
! Match BB3's prefixes
!
ip prefix-list RIP_FROM_BB3 seq 5 permit 30.0.0.0/14 ge 16 le 16
ip prefix-list RIP_FROM_BB3 seq 10 permit 31.0.0.0/14 ge 16 le 16

!
! Tag BB1 prefixes with tag1 and BB3 prefixes with tag 3
!
route-map RIP_TO_EIGRP permit 10
 match ip address prefix-list RIP_FROM_BB1
 set tag 1
!
route-map RIP_TO_EIGRP permit 20
 match ip address prefix-list RIP_FROM_BB3
 set tag 3
!
route-map RIP_TO_EIGRP permit 30
```


Task 2.2 Breakdown

OSPF cost calculation is based on the following formula:

$$COST = \frac{Reference\_Bandwitdh}{Interface\_Bandwidth}$$

Reference_Bandwidth defaults to 100Mbps and *Interface_Bandwidth* is the configured bandwidth statement of an interface. If *Interface_Bandwidth* is greater than *Reference_Bandwidth,* the resulting cost value is 1. Since the reference bandwidth defaults to 100Mbps, this implies that all interfaces with a bandwidth above 100Mbps (OC3, GigE, etc) will have a cost of 1. In networks with high speed links, this calculation may result in suboptimal forwarding. To adjust how OSPF calculates its cost, change the reference bandwidth value by using the routing process subcommand auto-cost reference-bandwidth [*Reference_Mbps*].

The redistribution part has some complications. Firstly, we need to determine the minimum required number of points of redistribution. Obviously, mutual redistribution is needed at R5 and R6, as they connected the peripheral RIP clouds. Next, mutual redistribution needs to be set in R1 and R2, as the task requires R5 to be able to select between R1 and R2 when routing to BB1/BB3's prefixes. This redistribution topology contains a loop. Here are the problems that you may see

1) RIP routes leaked from R6 into the EIGRP domain will be redistributed into OSPF with the AD of 110. As a result, R1 and R2 will prefer reaching the RIP routes via OSPF which will result in a loop.
2) The above could be fixed by setting OSPF's external AD to 171, higher than EIGRP's external AD. But this will result in another problem.
3) If OSPF's external AD is set higher than EIGRP's then RIP routes redistributed into OSPF on R5 will loop back into OSPF by means of mutual redistribution in R1/R2 and preempt RIP routes in R5. The solution is to set RIP's AD on R5 lower then OSPF's AD (which is 110 by default). This is done by the virtue of the task requirement to keep the RIP prefixes in the routing table, no the eBGP ones, which translates into RIP's AD of 19.

In addition to dealing with redistribution loops, the redistribution configuration utilizes routing tags to distinguish the *origin* of the specified prefixes (e.g. BB1 or BB3 prefixes). To set a routing tag, simply issue the **`set tag [tag]`** command under a route-map used for redistribution.

The above task states that R5 should use R1 to get to the routes learned from BB1, while using R2 to get to the routes learned form BB3. In order to accomplish this, routes from BB1 and BB3 are set with separate tag values when redistributed into EIGRP on R6. As EIGRP is redistributed into OSPF on both R1 and R2, these tag values are *matched*, and an appropriate OSPF cost value is assigned.

🕮 Further Learning

Redistribution scenarios may be overly complicated, and sometimes it does not worth spending too much of precious exam time on them. If you are facing redistribution loops it could be easier to ensure full-reachability and some of the additional requirements (e.g. path engineering like in this task) and don't provide full redundancy by means of additional redistribution.

🕮 Deep Dive

- How could you verify if your topology is stable after redistribution?
- Why using optimum (shortest) paths should be an important result of redistribution?

Task 2.2 Verification

Verify that the costs have been recalculated:

```
Rack1R4#show interfaces FastEthernet0/0
FastEthernet0/0 is up, line protocol is up
  Hardware is Gt96k FE, address is 0019.5622.8dd6 (bia 0019.5622.8dd6)
  Internet address is 136.1.4.4/24
  MTU 1500 bytes, BW 100000 Kbit/sec, DLY 100 usec,
<snip>

Rack1R4#show ip ospf interface fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  Internet Address 136.1.4.4/24, Area 4
  Process ID 1, Router ID 150.1.4.4, Network Type BROADCAST, Cost: 200
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 150.1.4.4, Interface address 136.1.4.4
  No backup designated router on this network
<snip>
```

And for serial interfaces as well:

```
Rack1R4#show interfaces S0/0/0
Serial0/0/0 is up, line protocol is up
  Hardware is QUICC Serial
  Internet address is 136.1.245.4/24
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,

Rack1R4#show ip ospf interface S0/0/0
Serial0/0/0 is up, line protocol is up
  Internet Address 136.1.245.4/24, Area 0
  Process ID 1, Router ID 150.1.4.4, Network Type POINT_TO_MULTIPOINT,
Cost: 12953
```

Verify OSPF group pacing times:

```
Rack1R5#show ip ospf | include arrival|pacing
 Minimum LSA arrival 2000 msecs
 LSA group pacing timer 40 secs
 Interface flood pacing timer 33 msecs
 Retransmission pacing timer 66 msecs
<snip>
```


Verify BB1's prefixes on R5 (chould be reachable via R1):

```
Rack1R5#show ip route | inc 212.18
O E2 212.18.1.0/24 [110/1] via 136.1.15.1, 00:01:29, Serial0/0/0.15
O E2 212.18.0.0/24 [110/1] via 136.1.15.1, 00:01:29, Serial0/0/0.15
O E2 212.18.3.0/24 [110/1] via 136.1.15.1, 00:01:29, Serial0/0/0.15
O E2 212.18.2.0/24 [110/1] via 136.1.15.1, 00:01:29, Serial0/0/0.15
```

Verify BB3's prefixes on R5 (should be reachable via R2):

```
Rack1R5#show ip route | inc ( 31| 30).*via
O E2    31.3.0.0 [110/1] via 136.1.245.2, 00:03:18, Serial0/0/0.245
O E2    31.2.0.0 [110/1] via 136.1.245.2, 00:03:18, Serial0/0/0.245
O E2    31.1.0.0 [110/1] via 136.1.245.2, 00:03:18, Serial0/0/0.245
O E2    31.0.0.0 [110/1] via 136.1.245.2, 00:03:18, Serial0/0/0.245
O E2    30.2.0.0 [110/1] via 136.1.245.2, 00:03:18, Serial0/0/0.245
O E2    30.3.0.0 [110/1] via 136.1.245.2, 00:03:18, Serial0/0/0.245
O E2    30.0.0.0 [110/1] via 136.1.245.2, 00:03:18, Serial0/0/0.245
O E2    30.1.0.0 [110/1] via 136.1.245.2, 00:03:18, Serial0/0/0.245
```

Verify RIP routes on R5:

```
Rack1R5#show ip route rip
R    222.22.2.0/24 [19/7] via 192.10.1.254, 00:00:27, FastEthernet0/0
R    220.20.3.0/24 [19/7] via 192.10.1.254, 00:00:27, FastEthernet0/0
R    205.90.31.0/24 [19/7] via 192.10.1.254, 00:00:27, FastEthernet0/0
```


Verify connectivity between the internal routers using the ping script:

```
tclsh
foreach i {
136.1.136.1
136.1.15.1
150.1.1.1
136.1.245.2
150.1.2.2
136.1.23.2
136.1.136.3
150.1.3.3
136.1.23.3
136.1.245.4
136.1.4.4
150.1.4.4
136.1.45.4
136.1.44.4
136.1.245.5
136.1.15.5
150.1.5.5
136.1.45.5
136.1.57.5
192.10.1.5
136.1.136.6
54.1.3.6
150.1.6.6
204.12.1.6
150.1.7.7
136.1.57.7
136.1.7.7
} { puts "exec [ping $i]" }
```


Note that VLAN3 and VLAN29 are not advertised by an IGP protocol. This can create possible BGP next hop issues with the BGP peering session. Additionally, SW1 and SW2 are not involved in IGP routing at the moment.

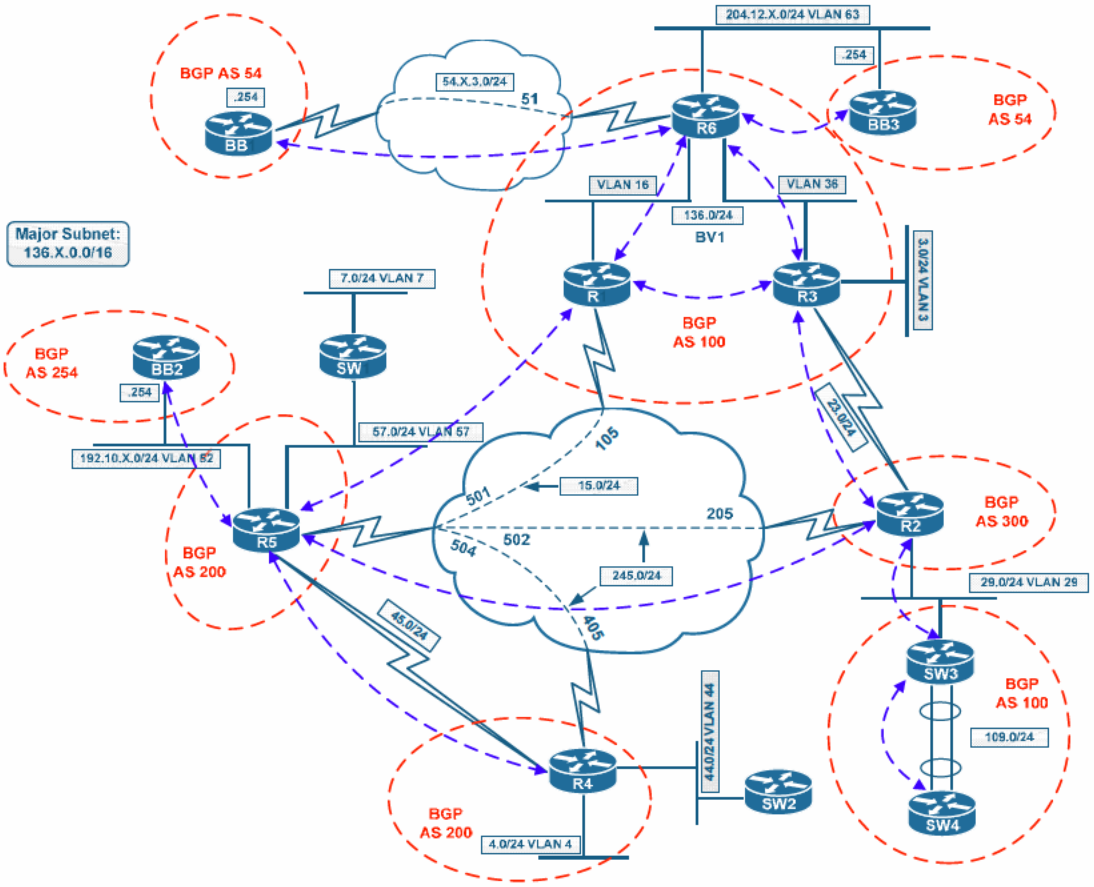
Finally verify connectivity to the backbone IGP networks:

```
foreach i {
212.18.1.1
212.18.0.1
212.18.3.1
212.18.2.1
31.3.0.1
31.2.0.1
31.1.0.1
31.0.0.1
30.2.0.1
30.3.0.1
30.0.0.1
30.1.0.1
192.10.1.254
54.1.3.254
204.12.1.254
} { puts "exec [ping $i]" }
```


Task 2.3 Solutions

🕮 Before You Begin

This scenario involves some BGP path engineering, which requires you knowing the exiting BGP peering sessions. Perform BGP peering session discovery (and troubleshooting if required) using the command **`show ip bgp summary`**. The diagram below reflects the BGP peerings.



The solution futher utilized AS_PATH prepending to affect inbound paths to AS. You may find introduction information for this feature by completing VOL1's scenario **BGP > BGP Bestpath Selection – AP-Path Prepending**. For prefix filtering the solution utilizes tagging with **no-export** community, which is explained in VOL1's scenario **BGP > BGP Communities – No Export**.

R1:

```
!
! R1 preferms AS_PATH manipulation to affect incoming path
! for R3'prefix
!
router bgp 100
 neighbor 136.1.15.5 route-map TO_R5 out
!
ip prefix-list VLAN3 seq 5 permit 136.1.3.0/24
!
route-map TO_R5 permit 10
 match ip address prefix-list VLAN3
 set as-path prepend 100 100
!
route-map TO_R5 permit 1000
```

R3:

```
!
! Advertise VLAN 3 into BGP
!
router bgp 100
 network 136.1.3.0 mask 255.255.255.0
```

R6:

```
!
! R6 exmploys setting the NO_EXPORT community on the prefixes
! received from BB1/BB3 so the prefixes are not exported from AS 100.
!
router bgp 100
 neighbor 54.1.3.254 route-map NO_EXPORT in
 neighbor 136.1.136.1 send-community
 neighbor 136.1.136.3 send-community
 neighbor 204.12.1.254 route-map NO_EXPORT in
!
route-map NO_EXPORT permit 10
 set community no-export
```


Task 2.3 Breakdown

In order to prevent your AS from being used as transit, prefixes which are learned from a provider should not be advertised out to another provider. In the above case, R6 is learning prefixes from AS 54 through two entry points. In order to prevent other ASs (such as AS 300) from using AS 100 as transit to reach these prefixes, AS 100 must prevent them from being advertised. However the task in question states that this "configuration should be done only on R6." Since R6 is not peering with AS 300 or AS 200, this poses a problem. In order to affect whether or not other peers in AS 100 advertise these prefixes, R6 is setting the community to *no-export*.

Pitfall

BGP community attributes are not included in advertisements by default. In order to enable the sending of the community attribute along with a prefix, use the BGP routing process subcommand **`neighbor [neighbor] send-community`**. Unless this command is included, the community value will be stripped from an advertisement, regardless of whether it is going to an iBGP or EBGP neighbor.

Next, AS 100 is trying to affect the inbound traffic flow from its BGP peers. Recall which attribute should be set to affect BGP path selection:

Attribute	Direction Applied	Traffic Flow Affected
Weight	Inbound	Outbound
Local-Preference	Inbound	Outbound
AS-Path	Outbound	Inbound
MED	Outbound	Inbound

To affect inbound traffic flow, you must either prepend the AS-path attribute or set the multi-exit discriminator (MED) value as the prefix is advertised outside the AS. However, since MED is only compared by default on prefixes learned from the same AS, AS-path prepending must be used in this case.

Since AS 100 wants traffic to come in from AS 300, it is prepending its own AS number out twice when sending updates to AS 200. Therefore, when AS 200 compares the AS-path length on the prefixes learned from AS 100 and AS 300, it will prefer the path through AS 300.

Task 2.3 Verification

Verify that the communities are being sent:

```
Rack1R6#show ip bgp neighbors 136.1.136.3 | include Comm
  Community attribute sent to this neighbor
```

Check that R1 and R3 are actually receiving prefixes from AS 54

```
Rack1R3#show ip bgp regexp _54$
<output omitted>

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   204.12.1.254             0    100      0 54 i
*>i28.119.17.0/24   204.12.1.254             0    100      0 54 i
*>i114.0.0.0        204.12.1.254             0    100      0 54 i
<output omitted>
```

Check that AS 54 prefixes are not being exported to eBGP peers:

```
Rack1R3#show ip bgp 114.0.0.0
BGP routing table entry for 114.0.0.0/8, version 37
Paths: (1 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
  Not advertised to any peer
  54
    204.12.1.254 (metric 537600) from 136.1.136.6 (150.1.6.6)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Community: no-export
```


Look into R5's BGP table and RIB contents for the VLAN3 subnet:

```
Rack1R5#show ip bgp 136.1.3.0
BGP routing table entry for 136.1.3.0/24, version 26
Paths: (2 available, best #1, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
     1           2
  300 100
    136.1.245.2 from 136.1.245.2 (150.1.2.2)
      Origin IGP, localpref 100, valid, external, best
  100 100 100
    136.1.15.1 from 136.1.15.1 (150.1.1.1)
      Origin IGP, localpref 100, valid, external

Rack1R5#show ip route 136.1.3.0
Routing entry for 136.1.3.0/24
  Known via "bgp 200", distance 20, metric 0
  Tag 300, type external
  Last update from 136.1.245.2 00:01:26 ago
  Routing Descriptor Blocks:
  * 136.1.245.2, from 136.1.245.2, 00:01:26 ago
      Route metric is 0, traffic share count is 1
      AS Hops 2
      Route tag 300
```


Task 2.4 Solutions

🕮 Before You Begin

The solution utilizes BGP weight modification to match the output presented in the scenario. More information about this BGP attribute and BGP prefix filtering/matching could be found in the following VOL1 labs:

BGP > BGP Bestpath Selection – Weight
BGP > BGP Filtering with Prefix-Lists

R2:

```
!
! Advertise VLAN 29 prefix on R2
!
router bgp 300
 network 136.1.29.0 mask 255.255.255.0
```

R5:

```
!
! Match the prefix as it learned from R1/R2:
!
router bgp 200
 neighbor 136.1.245.2 route-map VLAN29_PREFIX in
 neighbor 136.1.15.1 route-map VLAN29_PREFIX in
!
ip prefix-list VLAN29 seq 5 permit 136.1.29.0/24
!
! Set Weight to match the show command output
!
route-map VLAN29_PREFIX permit 10
 match ip address prefix-list VLAN29
 set weight 100
!
route-map VLAN29_PREFIX permit 1000
```


Note

The scenario further uses conditional advertisement in R2. This technology is explained in details in VOL1 lab **BGP > BGP Conditional Advertisement**

R2:

```
!
! Conditionally advertise 29.0/24 prefix only if the Serial interface
! is not in the BGP table. Conditional advertisement applies
! per-neighbor.
!
router bgp 300
 network 136.1.23.0 mask 255.255.255.0
 neighbor 136.1.245.5 advertise-map ADVERTISE non-exist-map NON_EXIST
!
ip prefix-list SERIAL seq 5 permit 136.1.23.0/24
!
ip prefix-list VLAN29 seq 5 permit 136.1.29.0/24
!
route-map NON_EXIST permit 10
 match ip address prefix-list SERIAL
!
route-map ADVERTISE permit 10
 match ip address prefix-list VLAN29
```

Note

Enable R3 and SW3 to accept the updates with AS 100 in the path – that is the updates from each other. This feature is explained in VOL1 scenario **BGP > AllowAS In**

R3:

```
router bgp 100
 neighbor 136.1.23.2 allowas-in
```

SW3:

```
router bgp 100
 network 136.1.109.0 mask 255.255.255.0
 neighbor 136.1.29.2 allowas-in
```


Task 2.4 Breakdown

The order of the BGP best-path selection algorithm dictates that you have absolute control over how traffic leaves your autonomous system. This is due to the fact that the attributes used to affect outbound traffic (weight and local-preference) are higher in the decision process than those used to affect inbound traffic (AS-path and MED). By setting either the weight or local-preference on a prefix, you can affect how traffic leaves your AS to get *to that* prefix. Under certain circumstances, this behavior may be undesirable. Therefore, BGP conditional advertisement offers an alternative way to affect how traffic enters your AS. By *not* advertising a prefix to a specific neighbor, it is forced to route through a peer that does have a route to it. This feature is typically used when you have multiple connections of varying speeds to one or more providers. By controlling which prefixes get advertised to which neighbors, traffic can be forced to route in the appropriate link. In the case of a link failure, conditional advertisement will begin advertising the prefixes in question to the configured neighbor.

BGP conditional advertisement consists of two parts, the prefix or prefixes to watch, and the prefix or prefixes to advertise. It is applied by issuing the BGP process subcommand **neighbor [neighbor] advertise-map [advertise-map] non-exist-map [non-exist-map]**, where **advertise-map** is a route-map that matches the prefix that will be conditionally advertised, and **non-exist-map** is a route-map that matches the prefix to be watched. Once the prefix in the **non-exist-map** leaves the *BGP table*, the prefix in the **advertise-map** is advertised to the configured neighbor. Note that both of these prefixes must exist in the BGP table before configuring conditional advertisement.

In the task, AS 300 wants all traffic for the prefix 136.1.29.0/24 to come in the Serial link 136.1.23.0/24, unless it is down. To accomplish this, the 136.1.29.0/24 prefix should only be advertised from R2 to R5 if the Serial link 136.1.23.0/24 is down. 136.1.29.0/24 will therefore be matched in the advertise-map, while 136.1.23.0/24 is matched in the non-exist-map.

Last thing in this scenario is the same AS number preventing AS from learning BGP prefixes. By default, if a BGP speaker receives an update with its own AS in the AS path, the update is dropped. This is used by BGP as a means of loop prevention and is the normal desired behavior. To allow a router to accept a BGP update with its own AS in the AS path, the **allowas-in** option is used.

🕮 Further Learning

Another option to solve this problem, although not permitted by this particular task, would be to configure R2 to use the AS override feature. When AS override option is used, if an update is to be sent to a BGP peer that already contains the remote BGP peer's AS in the AS path, the local BGP AS is replaced with the peer's AS. This will ensure the remote BGP peer will accept the BGP update, since its AS is not in the AS path anymore.

🕮 Deep Dive

- Is there an alternative solution to using BGP conditional route advertisement?

Task 2.4 Verification

Verify that VLAN29 has a weight of 100 in the BGP table and no other prefixes (e.g. 205.90.31.0) are affected:

```
Rack1R5#show ip bgp
BGP table version is 11, 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
*> 136.1.3.0/24     136.1.245.2                          0 300 100 i
*                    136.1.15.1                           0 100 100 100
i
*> 136.1.23.0/24    136.1.245.2              0           0 300 i
*                    136.1.15.1                           0 100 300 i
*> 136.1.29.0/24    136.1.15.1                         100 100 300 i
<output omitted>
```

Verify conditional advertisement on R2.

```
Rack1R2#show ip bgp neighbors 136.1.245.5 | include Condition
  Condition-map NON_EXIST, Advertise-map ADVERTISE, status: Withdraw
                                                      ↑
                                          Prefix Not Advertised

Rack1R2#show ip bgp neighbors 136.1.245.5 advertised-routes
BGP table version is 102, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 136.1.3.0/24     136.1.23.3               0             0 100 i
*> 136.1.23.0/24    0.0.0.0                  0         32768 i
*> 136.1.109.0/24   136.1.29.9               0             0 100 i

Total number of prefixes 3
```



```
Rack1R2(config)#interface Serial0/1
Rack1R2(config-if)#shutdown  ← Serial link fails

Rack1R2#show ip bgp 136.1.23.0
% Network not in table← Prefix no longer exists
Rack1R2#show ip bgp neighbors 136.1.245.5 | include Condition
  Condition-map NON_EXIST, Advertise-map ADVERTISE, status: Advertise
                                                          ↑
                                                Prefix Now Advertised

Rack1R2#show ip bgp neighbors 136.1.245.5 advertised-routes
BGP table version is 6, local router ID is 2.2.2.2
Status codes: s suppressed, d damped, h history, * valid, > best,
              i - internal, r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 136.1.29.0/24     0.0.0.0                  0         32768 i
        ↑
Prefix Now Advertised
```

Now confirm that R3 and SW3 can see prefixes originated from within each other's AS:

```
Rack1R3#show ip bgp 136.1.109.0
BGP routing table entry for 136.1.109.0/24, version 40
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  300 100
    136.1.23.2 from 136.1.23.2 (150.1.2.2)
      Origin IGP, localpref 100, valid, external, best

Rack1SW3#show ip bgp
BGP table version is 19, local router ID is 150.1.9.9
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   136.1.29.2                             0 300 100 54 i
*> 28.119.17.0/24   136.1.29.2                             0 300 100 54 i
<output omitted>
*> 136.1.109.0/24   0.0.0.0                  0         32768 i
*> 205.90.31.0      136.1.29.2                             0 300 200 254 ?
*> 220.20.3.0       136.1.29.2                             0 300 200 254 ?
*> 222.22.2.0       136.1.29.2                             0 300 200 254 ?
```


Task 3.1 Solution

🕮 Before You Begin

For introductory information on Site-Local Addresses and their replacement – Unique Local Addresses, as well as IPv6 Tunneling you may check the following VOL1 labs

IPv6 > IPv6 Link Local Addressing
IPv6 > Unique Local Addressing
IPv6 > EUI-64 Addressing
IPv6 > IPv6 Tunneling

R2:
```
interface FastEthernet0/0
 ipv6 address 2001:CC1E:1:202::/64 eui-64
!
interface Tunnel0
 ipv6 address FEC0::2/64
 tunnel source 150.1.2.2
 tunnel destination 150.1.4.4
```

R4:
```
interface FastEthernet0/0
 ipv6 address 2001:CC1E:1:404::/64 eui-64
!
interface Tunnel0
 ipv6 address FEC0::4/64
 tunnel source 150.1.4.4
 tunnel destination 150.1.2.2
```

Task 3.1 Breakdown

The above example illustrates how to tunnel IPv6 datagrams over the IPv4 network using GRE encapsulation. This configuration is similar to that of IPv6IP tunneling. One advantage with GRE is support for tunneling of multiple non-IP protocol stacks simultaneously (IPX, CLNS, etc).

Task 3.1 Verification

Verify IPv6 addressing (note the EUI-64 host part):

```
Rack1R2#show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::212:43FF:FEA6:6C0
    2001:CC1E:1:202:212:43FF:FEA6:6C0
<snip>
Tunnel0                    [up/up]
    FE80::212:43FF:FEA6:6C0
    FEC0::2
Rack1R2#

Rack1R4#show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::219:56FF:FE22:8DD6
    2001:CC1E:1:404:219:56FF:FE22:8DD6
<snip>
Tunnel0                    [up/up]
    FE80::219:56FF:FE22:8DD6
    FEC0::4
Rack1R4#
```

Verify tunnel status:

```
Rack1R4#show interfaces tunnel 0
Tunnel0 is up, line protocol is up
  Hardware is Tunnel
  MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 150.1.4.4, destination 150.1.2.2
  Tunnel protocol/transport GRE/IP
```

Verify L3 connectivity:

```
Rack1R4#ping fec0::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FEC0::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/71/72 ms
```


Task 3.2 Solution

🕮 Before You Begin

OSPFv3 is similar to OSPFv2 in many respects, so a good working knowledge of OSPFv2 is helpful in understanding OSPFv3 configuraiton. For the specific configuration examples on OSPFv3 check VOL1 scenario **IPv6 > OSPFv3**

R2:
```
ipv6 unicast-routing
!
ipv6 router ospf 1
!
interface Tunnel 0
 ipv6 ospf 1 area 0
 ipv6 ospf hello-interval 1
 ipv6 ospf dead-interval 3
!
interface FastEthernet 0/0
 ipv6 ospf 1 area 0
```

R4:
```
ipv6 unicast-routing
!
ipv6 router ospf 1
 default-information originate always
!
interface Tunnel 0
 ipv6 ospf 1 area 0
 ipv6 ospf hello-interval 1
 ipv6 ospf dead-interval 3
!
interface FastEthernet 0/0
 ipv6 ospf 1 area 0
```

Task 3.2 Breakdown

OSPFv3 and OSPFv2 are different protocols, though they exhibit many similarities. Consequently, OSPFv3 configuration commands are very similar to their OSPFv2 counterparts, though they use different syntax starting with the keyword **ipv6**. During your pracrice you should memory all these commands and be able to quickly recover typical OSPFv3 configuration steps: enabling routing process, applying OSPFv3 areas, tuning timer/network types and configuring redistribution if required, including default route origination.

Task 3.2 Verification

Verify OSPFv3 neighbors:

```
Rack1R2#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID
Interface
150.1.4.4         1   FULL/  -        00:00:02    15
Tunnel0
```

Verify OSPFv3 routes:

```
Rack1R2#show ipv6 route ospf
IPv6 Routing Table - 8 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
OE2  ::/0 [110/1], tag 1
     via FE80::21B:D4FF:FE47:37B0, Tunnel0
O   2001:CC1E:1:404::/64 [110/11112]
     via FE80::21B:D4FF:FE47:37B0, Tunnel0
Rack1R2#

Rack1R4#show ipv6 route ospf
IPv6 Routing Table - Default - 6 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:CC1E:1:202::/64 [110/1001]
     via FE80::211:92FF:FE17:D160, Tunnel0
```


Task 4.1Solution

🕮Before You Begin

For introduction to MPLS technology and LDP protocol you may practice VOL1's scenario **MPLS VPN > MPLS LDP**

R4:

```
!
! VPNv4 peering interfrace prefix MUST be /32
!
interface Loopback0
 ip address 150.1.4.4 255.255.255.255
 exit
!
! Enable TDP (Cisco's protocol) for label exchange
! Force router-id for LDP
!
mpls ip
mpls label protocol tdp
mpls ldp router-id loopback 0 force

!
! Force LDP transport connection to stick to the physical interfaces
!
interface Serial 0/1/0
 mpls ip
 mpls ldp discovery transport-address interface
 mpls label protocol tdp
!
interface Serial 0/0/0
 mpls ip
 mpls ldp discovery transport-address interface
 mpls label protocol tdp
```



```
R5:
!
! VPNv4 peering interfrace prefix MUST be /32
!
interface Loopback0
 ip address 150.1.5.5 255.255.255.255
 exit
!
mpls ip
mpls ldp router-id loopback 0 force
!
! Force LDP transport connection to stick to the physical interfaces
!
interface Serial 0/1/0
 mpls ip
 mpls ldp discovery transport-address interface
 mpls label protocol tdp
!
interface Serial 0/0/0.245
 mpls ip
 mpls ldp discovery transport-address interface
 mpls label protocol tdp
```

Task 4.1 Breakdown

The scenario wording instructs to use the Tag Distribution Protocol (TDP) that used to be Cisco's default label-exchange protocol. The second requirement pertains to the TCP session formation for label exchange. By default, the loopbacks with highest IP addresses are used for this pupose, but there is an option to source transport connection off the physical interfaces. Lastly, there is a caveat with the Loopback interface prefixes – the netmasks are not /32 as it should be for proper end-to-end MPLS LSP establishment.

🕮 Deep Dive

- What are your options to deploying BGP VPNs if the underlying network does not support MPLS?
- Why is it important to force LDP router ID?

Task 4.1 Verification

Verify LDP (TPD, to be accurate) neighbors. Notice that connection uses physical interface addresses:

```
Rack1R5#show mpls ldp neighbor
    Peer TDP Ident: 150.1.4.4:0; Local TDP Ident 150.1.5.5:0
        TCP connection: 136.1.245.4.711 - 136.1.245.5.40771
        State: Oper; PIEs sent/rcvd: 0/8; Downstream
        Up time: 00:03:53
        TDP discovery sources:
          Serial0/0/0.245, Src IP addr: 136.1.245.4
        Addresses bound to peer TDP Ident:
          136.1.4.4       136.1.44.4      136.1.245.4     136.1.45.4
          150.1.4.4

Rack1R5#show mpls forwarding-table
Local  Outgoing      Prefix            Bytes Label   Outgoing    Next
Hop
Label  Label or VC   or Tunnel Id      Switched      interface
16     Pop Label     136.1.4.0/24      0             Se0/0/0.245
136.1.245.4
17     Pop Label     136.1.44.0/24     0             Se0/0/0.245
136.1.245.4
18     Pop Label     136.1.45.4/32     0             Se0/1/0
point2point
19     Pop Label     136.1.245.4/32    0             Se0/0/0.245
136.1.245.4
20     Pop Label     150.1.4.4/32      0             Se0/0/0.245
136.1.245.4
21     Pop Label     136.1.245.2/32    0             Se0/0/0.245
136.1.245.2
22     No Label      150.1.2.0/24      0             Se0/0/0.245
136.1.245.2
23     No Label      136.1.23.3/32     0             Se0/0/0.245
136.1.245.2
24     No Label      150.1.3.0/24      0             Se0/0/0.245
136.1.245.2
25     No Label      136.1.136.0/24    0             Se0/0/0.15
point2point
       No Label      136.1.136.0/24    0             Se0/0/0.245
136.1.245.2
26     No Label      150.1.1.0/24      0             Se0/0/0.15
point2point
```



```
Rack1R4#show mpls forwarding-table
Local  Outgoing      Prefix            Bytes Label   Outgoing    Next
Hop
Label  Label or VC   or Tunnel Id      Switched      interface
16     Pop Label     136.1.45.5/32     0             Se0/1/0
point2point
17     Pop Label     150.1.5.5/32      0             Se0/0/0
136.1.245.5
18     Pop Label     192.10.1.0/24     0             Se0/0/0
136.1.245.5
19     Pop Label     136.1.245.5/32    0             Se0/0/0
136.1.245.5
20     21            136.1.245.2/32    0             Se0/0/0
136.1.245.5
21     22            150.1.2.0/24      0             Se0/0/0
136.1.245.5
22     23            136.1.23.3/32     0             Se0/0/0
136.1.245.5
23     24            150.1.3.0/24      0             Se0/0/0
136.1.245.5
24     25            136.1.136.0/24    0             Se0/0/0
136.1.245.5
25     No Label      136.1.23.0/24     0             Se0/0/0
136.1.245.5
26     Pop Label     136.1.15.0/24     0             Se0/0/0
136.1.245.5
27     26            150.1.1.0/24      0             Se0/0/0
136.1.245.5
```


Task 4.2 Solution

🕮 Before You Begin

This scenario involves creating an MPLS VPN and facilitating VPN route exchange. Only conneted routes are being redistributed. For introduction into MPLS VPNs you may practice VOL1's scenarios

MPLS VPN > VRF Lite
MPLS VPN > MP-BGP VPNv4

R4:

```
!
! VRF import/export RTs are different
!
ip vrf VPN_AB
 rd 100:47
 route-target export 100:47
 route-target import 100:74
!
! Apply VRF to the interface
!
interface FastEthernet 0/1
 ip vrf forwarding VPN_AB
 ip address 136.1.44.4 255.255.255.0

!
! Configure the BGP process for MP-BGP exchange
!
router bgp 200
 address-family ipv4 unicast VRF VPN_AB
  redistribute connected
 !
 address-family vpnv4 unicast
  neighbor 150.1.5.5 activate
  neighbor 150.1.5.5 send-community both
```


R5:

```
!
! VRF import/export RTs are different
!
ip vrf VPN_AB
 rd 100:47
 route-target export 100:74
 route-target import 100:47
!
! Apply VRF to the interface
!
interface FastEthernet 0/1
 ip vrf forwarding VPN_AB
 ip address 136.1.57.5 255.255.255.0
!
router bgp 200
 address-family ipv4 unicast VRF VPN_AB
  redistribute connected
 !
 address-family vpnv4 unicast
  neighbor 150.1.4.4 activate
  neighbor 150.1.4.4 send-community both
```


Task 4.2 Breakdown

The goal of this scenario is setting a basic MPLS VPN for two sites. There is no IGP used for PE-CE routing, only the connected routes being redistributed into BGP. The only trick is the requirement of using different route-targets at every stie, which forces to use different extended community values for export/import in R4/R5.

🕮 Deep Dive

- How does regular traceroute utility works in MPLS VPN environment?

Task 4.2 Verification

Check VRF routing table:

```
Rack1R5#show ip route vrf VPN_AB

Routing Table: VPN_AB
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     136.1.0.0/24 is subnetted, 2 subnets
B       136.1.44.0 [200/0] via 150.1.4.4, 03:41:04
C       136.1.57.0 is directly connected, FastEthernet0/1
```

Verify intra-VPN connectivity:

```
Rack1R5#ping vrf VPN_AB 136.1.44.4 source fastEthernet 0/1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 136.1.44.4, timeout is 2 seconds:
Packet sent with a source address of 136.1.57.5
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/60 ms
```


Verify labeling (notice there is no transport label (null) as R4 and R5 are directly connected):

```
Rack1R5#show ip cef vrf VPN_AB 136.1.44.4
136.1.44.0/24
  nexthop 136.1.245.4 Serial0/0/0.245 label 29

Rack1R5#show bgp vpnv4 unicast all 136.1.44.0
BGP routing table entry for 100:47:136.1.44.0/24, version 9
Paths: (1 available, best #1, table VPN_AB)
Flag: 0x820
  Not advertised to any peer
  Local
    150.1.4.4 (metric 12954) from 150.1.4.4 (150.1.4.4)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Extended Community: RT:100:47
      mpls labels in/out nolabel/29
```


Task 4.3 Solution

🕮 Before You Begin

OSPF has somewhat complex extensions designed to ensure loopless redistribution in MP-BGP environment. For introductory information on these features you may want to check VOL1's scenario **MPLS VPN > PE-CE Routing with OSPF**

R4:

```
!
! To make sure the routers translate updates to type-3 LSA properly,
! OSPF domain IDs should match; by default domain-id is baased
! on the ospf process number.
!
router ospf 44 vrf VPN_AB
 network 0.0.0.0 255.255.255.255 area 44
 redistribute bgp 200 subnets
 domain-id 47.47.47.47
!
router bgp 200
 address-family ipv4 vrf VPN_AB
  redistribute ospf 44
```

R5:

```
router ospf 77 vrf VPN_AB
 network 0.0.0.0 255.255.255.255 area 77
 !
 ! Reflect SW1's settings for the NSSA area!
 !
 area 77 nssa
 redistribute bgp 200 subnets
 domain-id 47.47.47.47
!
router bgp 200
 address-family ipv4 vrf VPN_AB
  redistribute ospf 77
```


Task 4.3 Breakdown

Setting OSPF for PE-CE routing should be pretty straightforward, but there is a requirement of using different OSPF process IDs in R4 and R5. Following the BGP to OSPF redistribution rules, this will translate the BGP prefixes into type-5 LSA and install external routes. To resolve this problem, set the OSPF domain-IDs manually to the same value on both routers.

🕮 Deep Dive

- When would you need to manually use different OSPF domain-IDs on the PE routers?

Task 4.3 Verification

Check OSPF routers in the CE devices and confirm they show up as inter-area routes.

```
Rack1SW2#show ip route ospf
     136.1.0.0/24 is subnetted, 3 subnets
O IA    136.1.7.0 [110/3] via 136.1.44.4, 00:02:38, Vlan44
O IA    136.1.57.0 [110/2] via 136.1.44.4, 00:02:53, Vlan44
     150.1.0.0/32 is subnetted, 1 subnets
O IA    150.1.7.7 [110/3] via 136.1.44.4, 00:02:38, Vlan44

Rack1SW1#show ip route ospf
     136.1.0.0/24 is subnetted, 3 subnets
O IA    136.1.44.0 [110/2] via 136.1.57.5, 00:05:00, Vlan57
     150.1.0.0/16 is variably subnetted, 2 subnets, 2 masks
O IA    150.1.8.8/32 [110/3] via 136.1.57.5, 00:00:14, Vlan57
```

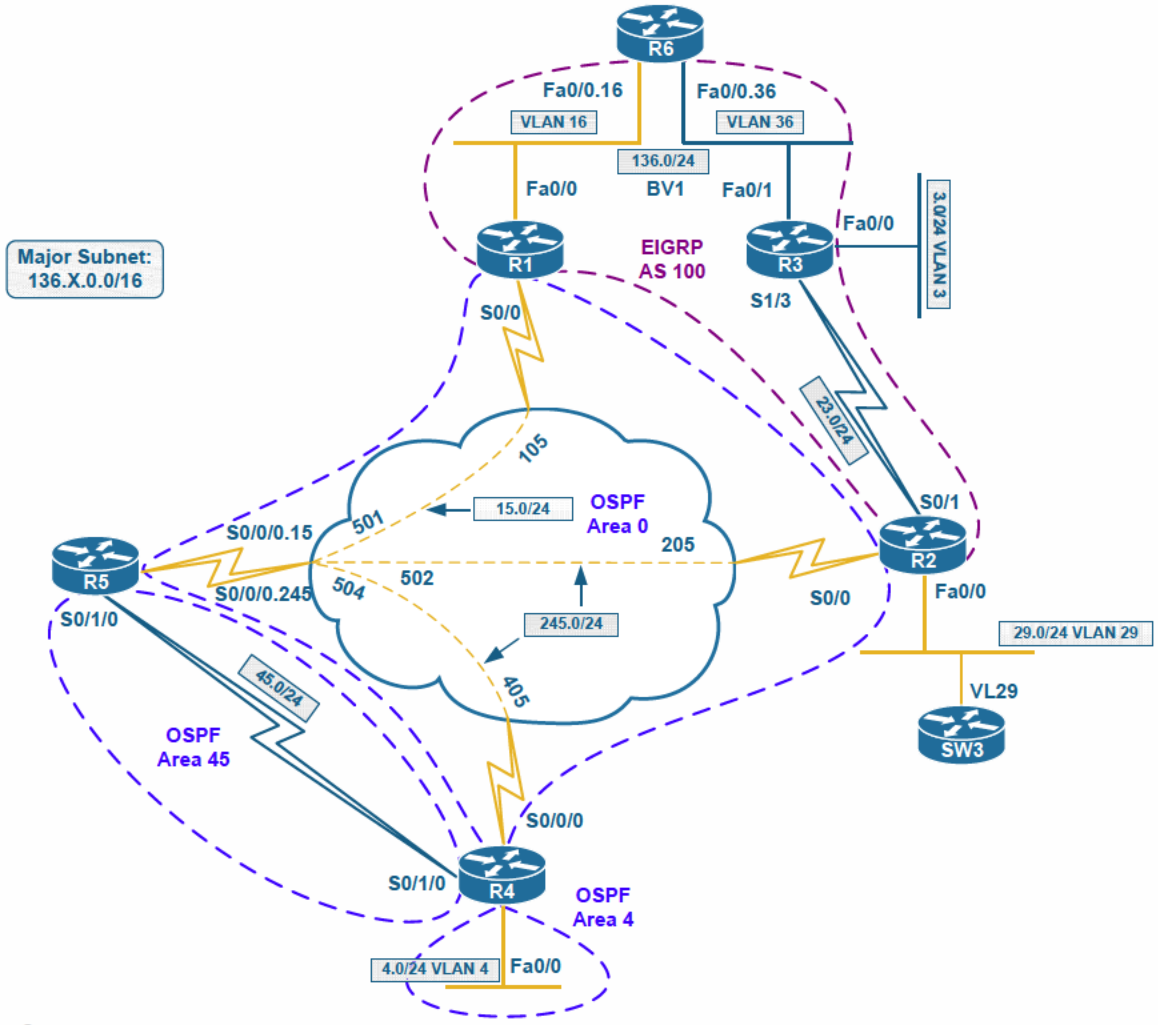
Ensure connectivity between the CE devices

```
Rack1SW1#ping 150.1.8.8 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.8.8, timeout is 2 seconds:
Packet sent with a source address of 150.1.7.7
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 58/62/67 ms

Rack1SW1#traceroute 150.1.8.8
Type escape sequence to abort.
Tracing the route to 150.1.8.8

  1 136.1.57.5 0 msec 9 msec 0 msec
  2 136.1.44.4 67 msec 67 msec 59 msec
  3 136.1.44.8 42 msec *  33 msec
```


Task 5.1 Solution

🕮 Before You Begin

Start by making a quick sketch of the existing multicast topology. Use the commands **`show ip pim neighbors`** and **`show ip pim interface`** to accomplish this. The multicast topology diagram is presented below. Note that all interfaces use sparse-dense mode.



R2:

```
interface FastEthernet0/0
 ip igmp static-group 228.22.22.22
```

Task 5.1 Breakdown

The **ip igmp static-group** command is different than the **ip igmp join-group** command in that the static-group command does not cause the devices to process the multicast packets themselves. Instead the device will just forward the multicast packets out the interface.

With the **ip igmp static-group** command, a device will not respond to ICMP echo requests sent to the multicast group as with the **ip igmp join-group** command. If a particular multicast group needs to be sent out an interface, the static-group command would be preferred over the join-group command. The static-group command causes the group to be fast switched. The join-group command will cause the device to process switch the group. Neither command is needed to enable hosts on a segment to receive multicast traffic, as long as the host supports IGMP.

🕮 Deep Deive

- When using dense-mode flooding, what would you need to configure in order to allow the spoke nodes on NBMA segment to exchange multicast traffic?

Task 5.1 Verification

Verify that the 228.22.22.22 group is statically configured:

```
Rack1R2#show ip igmp membership
Flags: A  - aggregate, T - tracked
       L  - Local, S - static, V - virtual, R - Reported through v3
       I - v3lite, U - Urd, M - SSM (S,G) channel
       1,2,3 - The version of IGMP the group is in
<output omitted>

 Channel/Group         Reporter        Uptime   Exp.  Flags  Interface
 *,224.0.1.39          136.1.245.5     00:05:37 02:44 2A     Se0/0
 *,224.0.1.40          136.1.29.2       00:06:31 02:34 2LA    Fa0/0
 *,228.22.22.22        0.0.0.0         00:00:08 stop  2SA    Fa0/0
```


Verify multicast routing (note that 228.22.22.22 has no RP, and is flooded in dense mode):

```
Rack1R5#ping 228.22.22.22

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 228.22.22.22, timeout is 2 seconds:
.

Rack1R2#show ip mroute
IP Multicast Routing Table
<output omitted>

(*, 228.22.22.22), 02:50:04/stopped, RP 0.0.0.0, flags: DC
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 02:48:50/00:00:00
    Serial0/0, Forward/Sparse-Dense, 02:50:04/00:00:00

(136.1.245.5, 228.22.22.22), 00:00:09/00:02:53, flags: T
  Incoming interface: Serial0/0, RPF nbr 136.1.245.5
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 00:00:09/00:00:0
```

Task 5.2 Solution

🕮 Before You Begin

For introduction to IGMP filtering features, you may refer to VOL1's scenario **Multicast > IGMP FIltering**

R4:

```
interface FastEthernet0/0
 ip igmp access-group 50
!
access-list 50 permit 225.25.25.25
access-list 50 permit 226.26.26.26
```


Task 5.2 Breakdown

To limit which multicast groups clients on a segment can join, use the **`ip igmp access-group`** interface command. In this task, clients will only be allowed to join multicast groups 225.25.25.25 and 226.26.26.26.

🕮 Before You Begin

- How would you filter a particular multicast source when using IGMPv3?

Task 5.2 Verification

Verify that the filtering is configured:

```
Rack1R4#show ip igmp interface Fa0/0 | inc access
  Inbound IGMP access group is 50

Rack1R4#show ip access-lists 50
Standard IP access list 50
    10 permit 225.25.25.25
    20 permit 226.26.26.26
```


Task 5.3 Solution

```
R1:
interface FastEthernet0/0
 ip multicast ttl-threshold 12
```

Task 5.3 Breakdown

Whenever a multicast packet is sent by a server, the TTL of the packet is set. The TTL can be set anywhere from 0 to 255. A multicast packet sent with a TTL of 1, or technically 0, will not allow a multicast router to forward the packet. This is the mechanism that OSPF uses to ensure that its packets are never sent across a router.

By using TTL, a multicast implementation can limit where the multicast traffic can be sent in the network or stop the multicast traffic from being sent out of a network. This is commonly referred to as *multicast scoping*. Multicast scoping can be difficult to implement reliably in a medium or large sized network. Issues like alternate paths in the network or tunnels used to carry multicast traffic can make scoping nearly impossible to properly implement.

The **ip multicast ttl-threshold** interface command stops the forwarding of multicast packets that do not have a TTL value greater than what is defined by the command.

🕮 Deep Dive

- What could be an alternative to using multicast TTL scoping?

Task 5.3 Verification

Check the interface-level IGMP settings and look to the threshold.

```
Rack1R1#show ip multicast interface fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  Internet address is 136.1.136.1/24
  Multicast routing: enabled
  Multicast switching: fast
  Multicast packets in/out: 6160/345
  Multicast TTL threshold: 12
  Multicast Tagswitching: disabled
```


Task 6.1 Solution

🕮 Before You Begin

This task is similar to the one presented in Lab1, and intend to solidify your knowledge of Reflexive Access-Lists. For introduction to RACLs, you may practice VOL1's scenario **`Security > Traffic Filtering with Reflexive Access Lists`**

R6:

```
interface Serial0/0/0
 ip access-group IN_ACL in
 ip access-group OUT_ACL out
!
ip access-list extended IN_ACL
 permit icmp any any time-exceeded
 permit icmp any any port-unreachable
 permit udp any any eq rip
 permit tcp any any eq bgp
 permit tcp any eq bgp any
 evaluate MY_REFLECT
!

ip access-list extended OUT_ACL
 permit tcp any any reflect MY_REFLECT
 permit udp any any reflect MY_REFLECT
 permit icmp any any reflect MY_REFLECT
```


Task 6.1 Breakdown

A reflexive access-list is used to meet the requirements of this section. Remember that by default, packets generated by the router itself will not be reflected. This is why BGP is explicitly permitted inbound. Instead of using a reflexive access list, CBAC could also be used to meet the section requirements, but the simplest solution has been selected.

We explicitly permitted ICMP time-exceeded and ICMP port-unreachable messages to allow the `traceroute` command to function normally. The time-exceeded messages are needed when a router in the path discards a packet due to the TTL expiring. The port unreachable messages are needed by UDP based traceroute (i.e. Cisco IOS). If ICMP echo based traceroute was used, we would not need to permit the ICMP port-unreachables as the reflective access-list will detect the ICMP echo request and dynamically allow the ICMP echo reply.

Configuring an access list on the interface blocking ICMP inbound will prevent successful pings sourced from R6. The section states to only allow ICMP traffic inbound if it originated from behind R6, but does not state anything about ICMP traffic originating from R6. When in doubt, make sure to get clarification from the proctor. This will affect the output of ping tests run from R6, so you may want to make a note to yourself that the failures are expected behavior.

🕮 Deep Dive

- How do reflective access-lists define sessions?
- What is the main scaling limitation of reflexive access-lists?

Task 6.1 Verification

Verify that the BGP peering is not disrupted and RIP updates are still being received:

```
Rack1R6#show ip bgp summary | include ^54|Nei
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
54.1.3.254      4    54     364     374       40    0    0 05:36:26        10

Rack1R6#show ip route rip | inc 54.1.3.254
R    212.18.1.0/24 [120/1] via 54.1.3.254, 00:00:23, Serial0/0/0
R    212.18.0.0/24 [120/1] via 54.1.3.254, 00:00:23, Serial0/0/0
R    212.18.3.0/24 [120/1] via 54.1.3.254, 00:00:23, Serial0/0/0
R    212.18.2.0/24 [120/1] via 54.1.3.254, 00:00:23, Serial0/0/0
```

Verify that reflective ACL works. Initiate TCP traffic from R3 to BB1 and check reflective ACL:

```
Rack1R3#telnet 54.1.3.254
Trying 54.1.3.254 ... Open

BB1>

Rack1R6#show ip access-lists  MY_REFLECT
Reflexive IP access list MY_REFLECT
     permit tcp host 54.1.3.254 eq telnet host 136.1.136.3 eq 16410 (34
matches) (time left 287)
```


Verify that traceroute is working with our filtering:

```
Rack1R3#traceroute 212.18.2.1

Type escape sequence to abort.
Tracing the route to 212.18.2.1

  1 136.1.136.6 4 msec 0 msec 0 msec
  2 54.1.3.254 36 msec *  36 msec
```

Task 6.2 Solutions

🕮 Before You Begin

To get better understanding of TCP intercept we recommend practicing VOL1's scenario **Security > TCP Intercept** and **Security > TCP Intercept Watch Mode**

R4:

```
ip tcp intercept list 125
ip tcp intercept watch-timeout 15
ip tcp intercept mode watch
!
access-list 125 permit tcp any host 136.1.4.100
```

Task 6.2 Breakdown

TCP intercept is used to help prevent a TCP SYN flood DoS attack. In a SYN flood DoS attack, a source (or in most cases many sources), sends a flood of TCP SYN packets usually containing bogus (i.e. fake) source IP addresses.

The SYN packet is the first part of the TCP 3-way handshake. When a server receives the TCP SYN (synchronization) packet from a client, the server replies with a 'SYN ACK' (synchronization acknowledgement). The server will then wait for the client to complete the handshake process. For the process to be completed, the client will send an ACK in response to the server's 'SYN, ACK'. At this point the TCP session will be established. If the ACK is not received from the client, the session will timeout and in turn will be torn down. Once the session is torn down, the server's resources will be released.

A TCP SYN flood DoS attack uses this 3-way handshake process to cause the server to allocate resources for sessions that will never become established. The source or sources of the attack in most cases send thousands of TCP SYN packets per second to the server using bogus source IP addresses. The server, which does not know that the source IP addresses are bogus, receives the SYN packets, and replies with the 'SYN ACK'. The server then begins to allocate resources for the anticipated TCP session. After 10's of thousands of these SYN packets have been received by the server within a few seconds, the server will run out of resources to allocate for additional TCP sessions. The server now has thousands of half open TCP sessions that will eventually timeout after failing to receive the ACK from the client. Since most, if not all, of the server's resources are tied up replying to the SYN packets generated by attackers, legitimate users will not be able to establish a TCP session with the server.

TCP intercept can be used to enable the router to intercept the TCP SYN packets. The router will proxy for the server, and send the SYN ACK to the client. If the router receives an ACK from the client, the router knows that the session is valid and connects the session with the server. In theory, TCP intercept is a good solution in that the attack is offloaded from the server. In reality, it is not a good long term solution, as the burden of the attack is now taken on by the router. In a real network, it is normally easier to just install more (or faster) servers to deal with the burden of a DoS attack.

There are two modes of TCP intercept. The first is intercept mode and the second is watch mode. With intercept mode, the router will actively intercept the TCP sessions. In watch mode, the router will not intercept the TCP sessions but will monitor the TCP sessions. If a session does not reach the established state within 30 seconds (default time), the router will send a RST to the server so the server can release the resources allocated for that particular session. This is the intercept mode used by this section. In this section, a TCP RST packet will be sent to the server for TCP sessions that do not become established within 15 seconds. An access-list is additionally used to restrict which hosts are being 'watched'.

🕮 Deep Dive

- What is the benefit of Watch mode over Intercept mode in the TCP Intercept feature?
- What is the only TCP intercept mode supported by CBAC inspection rules?
- How effective could be a router-based TCP intercept feature?

Task 6.2 Verification

Verify that TCP Intercept is working by initialting a “fake” TPC session, e.g. a session to non-existent destination.

```
Rack1R5#telnet 136.1.4.100
Trying 136.1.4.100 ...

Rack1R4#show tcp intercept statistics
Watching new connections using access-list 125
1 incomplete, 0 established connections (total 1)
0 connection requests per minute

Rack1R4#show tcp intercept connections
Incomplete:
Client            Server              State    Create   Timeout  Mode
136.1.245.5:59676 136.1.4.100:23      SYNSENT  00:00:11 00:00:03 W

Established:
Client            Server              State    Create   Timeout  Mode
```

Task 6.3 Solution

🕮 Before You Begin

The scenario uses Catalyst DHCP Snooping feature, which is explaned in depth in VOL1 scenarios

Security > DHCP Snooping
Security > DHCP Snooping and Information Option

R6:

```
!
! Exclude addreses to enforce DHCP range
!
ip dhcp excluded-address 136.1.136.1 136.1.136.99
ip dhcp excluded-address 136.1.136.201 136.1.136.254
!
! The next command is required to make IOS DHCP server accept empty
! giaddr in the DHCP messages. The giaddr field is inserted by DHCP
! snooping switches.
!
ip dhcp relay information trust-all
!
ip dhcp pool NET136
 network 136.1.136.0 /24
 default-router 136.1.136.254
 update arp
```


SW1:

```
!
! Enable DHCP snooping in SW1 and SW3
!
ip dhcp snooping vlan 16,36
ip dhcp snooping
!
interface FastEthernet 0/1
 ip dhcp snooping limit rate 10
!
interface FastEthernet 0/13
 ip dhcp snooping trust
!
interface FastEthernet 0/14
 ip dhcp snooping trust
!
interface FastEthernet 0/15
 ip dhcp snooping trust
!
interface FastEthernet 0/16
 ip dhcp snooping trust
!
interface FastEthernet 0/17
 ip dhcp snooping trust
!
interface FastEthernet 0/18
 ip dhcp snooping trust
```

SW3:

```
ip dhcp snooping vlan 16,36
ip dhcp snooping
!
interface FastEthernet 0/1
 ip dhcp snooping limit rate 10
!
interface range FastEthernet 0/13
 ip dhcp snooping trust
!
interface range FastEthernet 0/14
 ip dhcp snooping trust
```


Task 6.3 Breakdown

The scenario calls for the use of DHCP snooping along with DHCP-based ARP update feature in R6. The first feature allows for enforcement of DHCP security, by preventing DHCP spoofing attacks. The second feature is useful in wireless environment, where MAC-spoofing attacks are common. When configuring DHCP snooping, you need to make sure that the downstream switches (SW1 and SW3) are configured to trust DHCP messages coming from their uplink trunks, otherwise R6's responses will be simply ignored.

DHCP snooping feature is known to insert empty giaddr field in the transit DHCP messages, which makes many DHCP servers to reject them. There are two solutions to this problem – either configuring the DHCP server to trust packets with zero giaddr field (like we did in the task) or disable information option (Option 82) insertion in the switches.

In order for R6 to populate its ARP table based on DHCP database contents the **`update arp`** commands needs to be enabled under the DHCP pool configuration. When this feature is active, secure ARP table entries are created by the DHCP server and can only be removed from the ARP cache with command **`clear ip dhcp binding`** This provides additional security against ARP poisoning attacks.

🕮 Deep Dive

- What use of Option 82 you could think of?
- What possible DHCP attack you can think of?
- What other security features work together with DHCP Snooping?

Task 6.3 Verification

Temporarily change R1's IP address to auto-assigned via DHCP. Don't forget to change it back when you're done.

R1:

```
interface FastEthernet 0/0
 ip address dhcp
```

Rack1SW1#show ip dhcp snooping binding

```
MacAddress          IpAddress        Lease(sec)  Type           VLAN
Interface
------------------  ---------------  ----------  -------------  ----  -
00:0D:BD:F6:2F:80   136.1.136.100    86329       dhcp-snooping   16
FastEthernet0/1
Total number of bindings: 1
```

Rack1R6#show ip dhcp binding

```
Bindings from all pools not associated with VRF:
IP address          Client-ID/              Lease expiration
Type
                    Hardware address/
                    User name
136.1.136.100       0063.6973.636f.2d30.    Dec 07 2009 01:59 PM
Automatic
                    3030.642e.6264.6636.
                    2e32.6638.302d.4661.
                    302f.30
```



```
Rack1SW1#show ip dhcp snooping
Switch DHCP snooping is enabled
DHCP snooping is configured on following VLANs:
16,36
Insertion of option 82 is enabled
   circuit-id format: vlan-mod-port
    remote-id format: MAC
Option 82 on untrusted port is not allowed
Verification of hwaddr field is enabled
Interface                    Trusted     Rate limit (pps)
------------------------     -------     ----------------
FastEthernet0/1              no          10
FastEthernet0/13             yes         unlimited
FastEthernet0/14             yes         unlimited
FastEthernet0/15             yes         unlimited

Rack1SW3#show ip dhcp snooping
Switch DHCP snooping is enabled
DHCP snooping is configured on following VLANs:
16,36
DHCP snooping is operational on following VLANs:
16,36
DHCP snooping is configured on the following L3 Interfaces:

Insertion of option 82 is enabled
   circuit-id format: vlan-mod-port
    remote-id format: MAC
Option 82 on untrusted port is not allowed
Verification of hwaddr field is enabled
Verification of giaddr field is enabled
DHCP snooping trust/rate is configured on the following Interfaces:

Interface                    Trusted     Rate limit (pps)
------------------------     -------     ----------------
FastEthernet0/1              no          10
FastEthernet0/13             yes         unlimited
FastEthernet0/14             yes         unlimited
```


Task 7.1 Solution

🕮 Before You Begin

The scenario involves configuring IOS HTTP server. For introduction to this technology, check VOL1 lab **System Management > HTTP Server and Client**.

```
R4:
!
! Store the hashed password using MD5 hash
! Make sure the user has privilege level 15
! to be able to access the Web GUI
!
username WEB secret CISCO
username WEB privilege 15
!
! Enable HTTP server and tune server options, including access class.
!
ip http server
ip http port 8080
ip http access-class 75
ip http authentication local
!
access-list 75 permit 136.1.2.0 0.0.0.255
```

Task 7.1 Breakdown

Although not widely used, Cisco IOS supports management and configuration through a web browser. In this section, the router has been configured to listen to HTTP requests on TCP port 8080. An access-list has been additionally defined to permit devices from the 136.1.2.0/24 subnet to access the router via HTTP. This is similar to applying an access-class inbound under the VTY lines. Notice that the user configured for HTTP authentication has privilege level of 15. This is required by IOS to granth access to HTTP management.

🕮 Deep Dive

- What if the route was using AAA for authentication, are there any special settings required for HTTP server?

Task 7.1 Verification

Verify the HTTP server configuration:

```
Rack1R4#show ip http server status
HTTP server status: Enabled
HTTP server port: 8080
HTTP server authentication method: local
HTTP server access class: 75
<output omitted>
```

Check to see if password for user WEB is encrypted with md5 hash:

```
Rack1R4#show running-config | inc username WEB
username WEB secret 5 $1$L09G$fX0brRRcfgoQygNfWTc0Q1
```

Task 7.2 Solution

🕮 Before You Begin

For more information on Cisco IOS TFTP server refer to VOL1 scenario named **System Management > TFTP Server and Client**.

```
R3:
tftp-server flash:c2600-iuo-mz.122-13.bin alias cisco2-C2600
```

Task 7.2 Breakdown

The key to this section is the *alias* portion of the **tftp-server** command. When a router starts to boot up, it will look in its global configuration for any boot commands. If there are not any boot commands specified, the router will fall over to using the first image in flash. If an image is not found in flash, the router will then try to boot a default image via TFTP. The default IOS image name for the 2600 used in this lab is 'cisco2-C2600'. The image name is hardware dependent in that a 3800 will attempt to boot a different default IOS image.

To determine which IOS image a router will attempt to boot, reload the router and then send *control-break* to get into ROMMON mode. Once in ROMMON mode, type **confreg**, assuming you are using a 2600 series or higher router. You will then be able to see the default IOS image.

🕮 Deep Dive

- Can TFTP work across a packet firewall?


```
Rack1R1#reload

Proceed with reload? [confirm]

%SYS-5-RELOAD: Reload requested by console.
System Bootstrap, Version 11.3(2)XA4, RELEASE SOFTWARE (fc1)
Copyright (c) 1999 by cisco Systems, Inc.
TAC:Home:SW:IOS:Specials for info
C2600 platform with 65536 Kbytes of main memory

<BREAK SEQUENCE SENT>

PC = 0xfff0a530, Vector = 0x500, SP = 0x83fff8b0

monitor: command "boot" aborted due to user interrupt
rommon 1 > confreg

    Configuration Summary
enabled are:
load rom after netboot fails
console baud: 9600
boot: image specified by the boot system commands
      or default to: cisco2-C2600

do you wish to change the configuration? y/n  [n]:
```

Task 7.3 Solutions

🕮 Before You Begin

For more information on auto-install over Frame-Relay look into VOL1's scenario named **System Management > Auto-Install over Frame-Relay**.

R2:
```
interface FastEthernet0/0
 ip directed-broadcast
```

R5:
```
interface Serial0/0/0.555 point-to-point
 ip address 136.1.5.1 255.255.255.252
 ip helper-address 136.1.29.255
 frame-relay interface-dlci 555 protocol ip 136.1.5.2
```

Task 7.3 Breakdown

The solution for this section is auto-install over Frame Relay. A router is a BOOTP server by default, unless you disable it using the command no **ip bootp-server**. In this case R5 will give the new router the IP address of 136.X.5.2 via BOOTP.

One issue with this section is that the IP address of the TFTP was not given. The only information given is that the TFTP server is located in VLAN 29. The solution is to configure the **ip helper-address** command to point to the directed broadcast for the subnet. To allow for successful transmission, ensure that the last hop interface supports directed-broadcast, which is disabled by default.

🕮 Deep Dive

- How does BOOTP server learn the IP address to allocate for remote client?
- How would you disable BOOTP server in IOS router?

Task 7.3 Verification

Verify that the helper-address is configured:

```
Rack1R5#show ip helper-address
Interface           Helper-Address  VPN VRG Name             VRG State
Serial0/0/0.555       136.1.29.255     0   None                  None
```

Verify that DLCI is mapped, for BOOTP to work:

```
Rack1R5#show frame-relay map  | beg 0/0\.555
Serial0/0/0.555 (down): point-to-point dlci, dlci 555(0x22B,0x88B0),
broadcast
 status deleted
```

PVC with DLCI 555 is not yet provisioned.

Task 7.4 Solution

🕮 Before You Begin

This scenario employs IOS exec privilege level manipulation. If you need more information about this topic, you may want to check VOL1 scenario named **Security > AAA Local Command Authorization**.

R2:

```
enable secret level 1 CISCO
!
privilege exec level 0 traceroute
privilege exec level 0 ping
!
line vty 0 4
 privilege level 0
```


Task 7.4 Breakdown

Privilege levels are used to restrict user access to certain commands. There are 16 privilege levels available on the router (0-15). The privilege levels that (by default) have commands assigned to them are 0, 1, and 15. Privilege level 1 is commonly referred to as user mode, and privilege level 15 is commonly referred to as enable mode.

When first logging into a router, the default privilege level assigned to all lines (VTY, console, etc) is privilege level 1.

> **Note**
>
> When in privilege level 0 or 1, the router's prompt will be '>'. Any level above level 1 will have a prompt of '#'.

To change the default privilege level for a line, the **`privilege level`** line command is used. If the privilege level is set to 15 for a particular line the user will automatically be placed into enable mode (privilege level 15).

```
Rack1R1#show run | include (vty)|(privilege)
line vty 0 4
 privilege level 15
Rack1R1#
Rack1R1#telnet 150.1.1.1
Trying 150.1.1.1 ... Open


User Access Verification

Password:
Rack1R1#show privilege
Current privilege level is 15
Rack1R1#
```

Privilege level 0 is the lowest level on the router. There are only a few commands available to a user in privilege level 0.

```
Rack1R1>?
Exec commands:
  <1-99>   Session number to resume
  disable  Turn off privileged commands
  enable   Turn on privileged commands
  exit     Exit from the EXEC
  help     Description of the interactive help system
  logout   Exit from the EXEC
  voice    Voice Commands

Rack1R1>
```


Normally when privilege level 0 is used, additional commands are moved down to privilege level 0 from privilege level 1 or 15. To move a command from one privilege level to another, the **privilege** global configuration command is used. Commands in lower privilege levels are automatically available to users in higher privilege levels.

To switch between privilege levels, use the **enable** command. The default option on the **enable** command is '15'.

```
Rack2R3#enable ?
  <0-15>  Enable level
  <cr>

Rack2R3#enable
```

When switching from a higher privilege level to a lower privilege level, a password is not required. Only when switching from a lower level to a higher level is a password required. To configure a password for particular privilege level, use the **enable secret level** or the **enable password level** commands.

🕮 Deep Dive

- What other options to local command authorization you can think of?
- How could you prevent user from issuing configuration commands but viewing the running configuration using local command authorization?

Task 7.4 Verification

Telnet to R2 and verify the new privilege level 0 commands:

```
Rack1R2>ping 150.1.3.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.3.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/29/32 ms

Rack1R2>traceroute 150.1.6.6

Type escape sequence to abort.
Tracing the route to 150.1.6.6

  1 136.1.23.3 16 msec 16 msec 16 msec
  2 136.1.136.6 16 msec *  16 msec

Rack1R2>?
Exec commands:
  <output omitted>
  ping        Send echo messages
  traceroute  Trace route to destination
  <output omitted>
```


Task 7.5 Solution

```
R5:
!
! Move the commands to level 1
!
privilege exec level 1 debug ip rip
privilege exec level 1 undebug ip rip
privilege exec level 1 terminal monitor
```

Task 7.5 Breakdown

Pitfall

In a real network, when allowing users access to debugging command, ensure that the users are also given access to the **undebug** command.

Task 7.5 Verification

Enter privilege level 1 and verify the debug commands:

```
Rack1R5#enable 1
Rack1R5>
Rack1R5>debug ip ?
  rip  RIP protocol transactions

Rack1R5>debug ip rip
RIP protocol debugging is on

Rack1R5>undebug ip rip
```


Task 7.6 Solution

🕮 Before You Begin

For more information on SNMPv2c and access-control pleasre refer to VOL1 scenarios

System Management > SNMPv2 Server
System Management > SNMPv2c Access Control

```
SW1 and SW2:
!
! The management station is fictitious
!
access-list 50 permit 136.1.2.100
!
snmp-server community CISCORO RO 50
snmp-server community CISCORW RW 50
snmp-server location San Jose, CA US
snmp-server contact CCIE Lab SW1
snmp-server chassis-id 221-787878
snmp-server enable traps vtp
snmp-server host 136.1.2.100 CISCOTRAP vtp
```

Task 7.6 Verification

Verify that SNMP is configured correctly:

```
Rack1SW1#show snmp
Chassis: 221-787878
Contact: CCIE Lab SW1
Location: San Jose, CA US
<output omitted>
SNMP logging: enabled
    Logging to 136.1.2.100.162, 0/10, 0 sent, 0 dropped.
SNMP agent enabled
```


Task 7.7 Solution

🕮 Before You Begin

For comprehensive introduction to GLBP check VOL1's scenario **IP Service > GLBP**.and the post on the INE blog named "GLBP Explained"

http://blog.ine.com/2008/04/24/glbp-explained/

R1

```
!
! R1's weight is 10
! Additionally, load-balancing is set to weighted
! and MD5 hash authentication enabled.
!
interface FastEthernet0/0
 glbp 1 ip 136.1.136.254
 glbp 1 priority 200
 glbp 1 weighting 10
 glbp 1 load-balancing weighted
 glbp 1 authentication md5 key-string CISCO
 glbp 1 preempt
```

R3:

```
!
! R3's weight is 30
! Additionally, load-balancing is set to weighted
! and MD5 hash authentication enabled.
!
interface FastEthernet0/1
 glbp 1 ip 136.1.136.254
 glbp 1 weighting 30
 glbp 1 load-balancing weighted
 glbp 1 authentication md5 key-string CISCO
 glbp 1 preempt
```


R6:

```
!
! R6's GLBP settings apply to the BVI interface
! GLBP priority is set to maximum to ensure R6 is the AVG
! Weighting is set to 60 and linked to a tracked object
!
interface BVI1
 glbp 1 ip 136.1.136.254
 glbp 1 priority 255
 glbp 1 weighting 60
 glbp 1 load-balancing weighted
 glbp 1 authentication md5 key-string CISCO
 glbp 1 weighting track 1 decrement 60
 glbp 1 preempt
!
track 1 ip sla 1 reachability
!
ip sla 1
 icmp-echo 54.1.3.254
 timeout 1000
 frequency 1
!
ip sla schedule 1 start-time now life forever

!
! Since R6 has an ACL applied inbound from BB1 we need to allow ICMP
! replies back in
!
ip access-list extended IN_ACL
 25 permit icmp host 54.1.3.254 host 54.1.3.6 echo-reply
```


Task 7.7 Breakdown

The task calls for the use of GLBP as it requires weighted load-balancing. R1, R3 and R6 are set as members of the same GLBP group with R6 having the highest priority. This ensures R6 is elected as the AVG – Active Virtual Gateway. All routers in the group are set up as for weighted load-balancing with the respective weights of 10, 30 and 60, per the task requirements.

In this GLBP group, R6's weight is the maximum. The solution uses IP SLA object tracking to validate R6's connection to BB1. Per the task requirements, SLA paratmeters are set so that R6 pings BB1 every second. In case if R6 loses connection to BB1, R6's value is decremented to zero, effectively taking R6 out of load-balancing scheme.

🕮 Deep Dive

- What could be the problems with ARP-based load-balancing?
- What happens if an AVF goes down and some hosts are still using its virtual MAC address for ARP entries?

Task 7.7 Verification

Verify GLBP running status:

```
Rack1R6#show glbp
BVI1 - Group 1
  State is Active
    1 state change, last state change 00:21:09
  Virtual IP address is 136.1.136.254
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 1.440 secs
  Redirect time 600 sec, forwarder timeout 14400 sec
  Authentication MD5, key-string
  Preemption enabled, min delay 0 sec
  Active is local
  Standby is 136.1.136.3, priority 100 (expires in 8.256 sec)
  Priority 255 (configured)
  Weighting 60 (configured 60), thresholds: lower 1, upper 60
    Track object 1 state Up decrement 60
  Load balancing: weighted
  Group members:
    000d.285a.e5a1 (136.1.136.3) authenticated
    000d.bdf6.2f80 (136.1.136.1) authenticated
    001b.d470.03d8 (136.1.136.6) local
  There are 3 forwarders (1 active)
  Forwarder 1
    State is Listen
    MAC address is 0007.b400.0101 (learnt)
    Owner ID is 000d.bdf6.2f80
    Redirection enabled, 599.424 sec remaining (maximum 600 sec)
    Time to live: 14399.424 sec (maximum 14400 sec)
    Preemption enabled, min delay 30 sec
    Active is 136.1.136.1 (primary), weighting 10 (expires in 9.600
sec)
  Forwarder 2
    State is Active
      3 state changes, last state change 00:00:36
    MAC address is 0007.b400.0102 (default)
    Owner ID is 001b.d470.03d8
    Redirection enabled
    Preemption enabled, min delay 30 sec
    Active is local, weighting 60
  Forwarder 3
    State is Listen
    MAC address is 0007.b400.0103 (learnt)
    Owner ID is 000d.285a.e5a1
    Redirection enabled, 599.584 sec remaining (maximum 600 sec)
    Time to live: 14399.584 sec (maximum 14400 sec)
    Preemption enabled, min delay 30 sec
    Active is 136.1.136.3 (primary), weighting 30 (expires in 11.232
sec)
```


Check the status of the tracked object:

```
Rack1R6#show track
Track 1
  IP SLA 1 reachability
  Reachability is Up
    2 changes, last change 00:04:10
  Latest operation return code: OK
  Latest RTT (millisecs) 63
  Tracked by:
    GLBP BVI1 1
```

Now shut down R6's conncetion to BB1 and see how that affects the tracked object status:

```
Rack1R6#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R6(config)#int se 0/0/0
Rack1R6(config-if)#shutdown

Rack1R6#show track
Track 1
  IP SLA 1 reachability
  Reachability is Down
    3 changes, last change 00:00:56
  Latest operation return code: Timeout
  Tracked by:
    GLBP BVI1 1
```

Next, look at GLBP statistics. Notice that Virtual Forwarders 2 and 3 now belong to 136.X.136.3, meaning R3 has taken the virtual MAC address of R6. Even though Virtual Forwarder 2's owners MAC address is the address of R6, the active physical gateway is really R3, since R6's weighting has gone below the acceptable level. R3 will keep responding to the additional virtual MAC address for some time, until all the hosts have migrated.


```
Rack1R6#show glbp
BVI1 - Group 1
  State is Active
    1 state change, last state change 00:25:21
  Virtual IP address is 136.1.136.254
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 0.992 secs
  Redirect time 600 sec, forwarder timeout 14400 sec
  Authentication MD5, key-string
  Preemption enabled, min delay 0 sec
  Active is local
  Standby is 136.1.136.3, priority 100 (expires in 7.968 sec)
  Priority 255 (configured)
  Weighting 0, low (configured 60), thresholds: lower 1, upper 60
    Track object 1 state Down decrement 60
  Forwarder 2
    State is Listen
      4 state changes, last state change 00:01:20
    MAC address is 0007.b400.0102 (default)
    Owner ID is 001b.d470.03d8
    Redirection enabled
    Preemption enabled, min delay 30 sec
    Active is 136.1.136.3 (secondary), weighting 30 (expires in 9.280
sec)
  Forwarder 3
    State is Listen
    MAC address is 0007.b400.0103 (learnt)
    Owner ID is 000d.285a.e5a1
    Redirection enabled, 597.632 sec remaining (maximum 600 sec)
    Time to live: 14397.632 sec (maximum 14400 sec)
    Preemption enabled, min delay 30 sec
    Active is 136.1.136.3 (primary), weighting 30 (expires in 8.896
sec)
```


Task 8.1 Solution

🕮 Before You Begin

The scenario uses legacy Frame-Relay Traffic Shaping with the feature known as FECN reflection. To find more information about these technologies, check out VOL1's scenarios

QoS > Legacy Frame-Relay Traffic Shaping
QoS > Legacy Adaptive FRTS

R1 and R2

```
interface Serial0/0
 frame-relay class FRTS
 frame-relay traffic-shaping
!
! FECN reflection enabled under map-class
!
map-class frame-relay FRTS
 frame-relay cir 256000
 frame-relay bc 32000
 frame-relay mincir 192000
 frame-relay adaptive-shaping becn
 frame-relay fecn-adapt
```

R4 and R5:

```
interface Serial0/0/0
 frame-relay class FRTS
 frame-relay traffic-shaping
!
map-class frame-relay FRTS
 frame-relay cir 256000
 frame-relay bc 32000
 frame-relay mincir 192000
 frame-relay adaptive-shaping becn
 frame-relay fecn-adapt
```


Task 8.1 Breakdown

The CIR and MinCIR values are obvious from the task requirements, since the wording explicitly says to limit the values to 259K/192K. Since the task does not say anything about the Tc value, it could be selected as the default 125ms. Additionally, shaping should be configured as FECN-adaptive.

Forward Explicit Congestion Notification (FECN) is used by the Frame Relay switch to notify a router that the remote router is causing congestion in the network. Backward Explicit Congestion Notification (BECN) is use to notify a router that it is the source of the congestion. OSI and DECnet Phase V are the only protocols that will automatically map the FECN bit to their own congestion experienced bit. This allows the devices to decrease their window size and in turn will theoretically decrease network utilization. This method of slowing down by using windowing is similar to TCP decreasing the window size when a packet is lost.

It is important to note that the BECN and FECN bits are set in normal data frames, and are not explicit frames generated by the Frame Relay switch. This makes it theoretically possible that a router will never receive a frame with the BECN bit set if the remote router never sends data to it. A realistic example would be where a DLCI is used exclusively to send multicast traffic. In this case, the vast majority of frames will be in one direction, source toward the receivers. If congestion occurs the Frame Relay switch will start marking frames. The majority of the frames will of course be marked in the opposite direction of the congestion as most traffic will flow from the source toward the receivers. The **`frame-relay fecn-adapt`** map-class command is useful in this type of situation as the receiving router can generate a frame with the BECN bit set upon receipt of a frame with the FECN bit set. This will allow the source of the congestion to throttle its sending rate down if the **`frame-relay adaptive-shaping becn`** map-class command is configured on the router causing the congestion.

🕮 Deep Dive

- What other congestion notification FRTS may respond to?

Task 8.1 Verification

Check the Per-VC shaping settings on any of the routers, e.g. R5:

```
Rack1R5#show frame-relay pvc 502

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

DLCI = 502, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0/0.245
<output omitted>
  Shaping adapts to BECN
  pvc create time 05:11:19, last time pvc status changed 03:33:07
  cir 256000    bc 32000     be 0        byte limit 4000   interval 125
  mincir 192000    byte increment 4000  Adaptive Shaping BECN
  pkts 5         bytes 442       pkts delayed 0         bytes delayed 0
<output omitted>
```

Task 8.2 Solution

🕮 Before You Begin

The scenario combines a few technologies together, including MQC-based classification and CBWFQ settings as well as classification by means of time-based access-lists. For more information, check the following VOL1 scenarios:

QoS > MQC Classification and Marking
QoS > MQC Bandwidth Reservations and CBWFQ
Security > Traffic Filtering with Time-Based Access-Lists

R4:

```
class-map match-all HTTP_CLASS
 match access-group 150
!
policy-map HTTP_POLICY
  class HTTP_CLASS
   police cir 256000
!
interface FastEthernet0/1
 service-policy output HTTP_POLICY
!
access-list 150 permit tcp any eq www any time-range HTTP_TIMERANGE
!
time-range HTTP_TIMERANGE
 periodic weekdays 8:00 to 17:00
```


Task 8.2 Breakdown

In this task, a common mistake made is to configure the access-list incorrectly. Make sure that you closely read the wording of tasks in regards to the direction of the traffic flow when configuring access-lists. This task mentioned that HTTP responses send out R4's interface Fa0/1 be limited. This means that the below access-list would be incorrect, as it would match HTTP packets destined to a web server in VLAN 44.

```
access-list 150 permit tcp any any eq www time-range HTTP_TIMERANGE
```

The access-list needs to match the HTTP server's responses. This is why the access-list is configured as shown below.

```
access-list 150 permit tcp any eq www any time-range HTTP_TIMERANGE
```

The **time-range** option of the extended access-list allows for selective filtering based on the clock of the local router. Time based access-lists are created by first defining the **time-range** in global configuration mode. The keywords **absolute** and **periodic** determine whether the event will occur at one specific (absolute) time, or will recur at a certain (periodic) interval.

Once the local time is within the specified time-range, the access-list entry or entries which reference the time-range are active. When the time range is inactive, it is as if the access-list entries do not exist.

There are various methods that can be used to limit traffic, including policing, shaping, and even legacy CAR using the rate-limit command on the interface. If there are no section restrictions, then any method that limits the traffic could be used.

Task 8.2 Verification

Change the local clock and see how the time-based access-lists change their status:

```
Rack1R4#clock set 00:00:00 1 Jan 2004
Rack1R4#show clock
00:00:01.322 UTC Thu Jan 1 2004
Rack1R4#show access-lists
Extended IP access list 150
    10 permit tcp any eq www any time-range HTTP_TIMERANGE (inactive)
                                                          ↗
              time-range is inactive on a Thursday at 12am

Rack1R4#clock set 10:00:00 1 Jan 2004
Rack1R4#show clock
10:00:03.069 UTC Thu Jan 1 2004
R4#show access-lists
Extended IP access list 150
    10 permit tcp any eq www any time-range HTTP_TIMERANGE (active)
                                                          ↗
                time-range is active on a Thursday at 10am

Rack1R4#clock set 10:00:00 3 Jan 2004
Rack1R4#show clock
10:00:02.480 UTC Sat Jan 3 2004
Rack1R4#show access-lists
Extended IP access list 150
    10 permit tcp any eq www any time-range HTTP_TIMERANGE (inactive)
                                                          ↗
          time-range is inactive at the same time on a Saturday
```


Task 8.3 Solution

🕮 Before You Begin

For detailed overview of RSVP signaling procedures and use of RSVP for QoS check out VOL1 scenarios named:

QoS > RSVP and WFQ
QoS > RSVP an CBWFQ
QoS > RSVP and Per-VC WFQ

R4 and R5:

```
map-class frame-relay FRTS
 frame-relay fair-queue
```

R4:

```
!
! Notice the use of Per-VC WFQ as FRTS has been configured previously
!
interface Serial0/0/0
 ip rsvp bandwidth 128 64
 ip rsvp resource-provider wfq pvc
```

R5:

```
interface Serial0/0/0
ip rsvp bandwidth 128 64
!
interface Serial0/0/0.245 multipoint
 ip rsvp bandwidth 128 64
 ip rsvp resource-provider wfq pvc
```


Task 8.3 Breakdown

Resource Reservation Protocol (RSVP) is used to dynamically request specific QoS from the network for a particular data flow. A data flow is defined as sequence of packets that have the same QoS requirements and have the same source and destination. Note that the destination could possibly be more than one host in the case of IP multicast. RSVP requests will normally result in resources being reserved by each router along the path between the source and destination.

There are three possible requests that R4 can make for its VoIP traffic. The first is best effort. Best effort does just what is says, supplies a best effort QoS policy for particular data flow. Best effort is commonly used with general application traffic. The second possibility is controlled load. Controlled load is used for rate sensitive traffic. Rate sensitive traffic will be guaranteed bandwidth (rate) through RSVP. The third possibility is guaranteed delay. Guaranteed delay is used to help ensure a minimum amount of jitter. Delay is normally one of the more important QoS requirements in relation to VoIP.

Although this is a basic configuration in regards to RSVP, it is important to note that when using subinterfaces, the **`ip rsvp bandwidth`** command will need to be applied to the physical interface along with the subinterface. If more than one subinterface is using RSVP, the physical interface's **`ip rsvp bandwidth command`** will be the sum of all the subinterface's **`ip rsvp bandwidth`** commands. Also, in case where FRTS is used, you need to tell RSVP to use the per-VC WFQ mechanics for scheduling, as FRTS disables the per-interface WFQ. This is achieved using the command **`ip rsvp resource-provider wfq pvc.`**

🕮 Deep Dive

- What reservation styles RSVP supports?
- What good use could be of RSVP in modern networks?
- Is there a situation where RSVP scales well?

Task 8.3 Verification

Verify per-VC queueing (note the WFQ and the reserved conversations):

```
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 =
Serial0/0.245
<output omitted>

  Queueing strategy: weighted fair
  Current fair queue configuration:
   Discard     Dynamic      Reserved
   threshold   queue count  queue count
    64          16           4
  Output queue size 0/max total 600/drops 0
```

Check RSVP resources:

```
Rack1R5#show ip rsvp interface S0/0/0
interface    allocated  i/f max  flow max sub max
Se0/0        0          128K     64K      0

Rack1R5#show ip rsvp interface S0/0/0.245
interface    allocated  i/f max  flow max sub max
S0/0/0.245    0          128K     64K      0
```

To finish with RSVP verification, simulate RSVP sender and reservation hosts:

R5:

```
ip rsvp reservation-host 150.1.5.5 150.1.4.4 UDP 5000 4000 FF RATE 32 4
```

R4:

```
ip rsvp sender-host 150.1.5.5 150.1.4.4 UDP 5000 4000 32 4
```

Verify that the RSVP reservation is installed:

```
Rack1R4#show ip rsvp reservation
To          From        Pro DPort Sport Next Hop      I/F       Fi Serv BPS
150.1.5.5   150.1.4.4   UDP 5000  4000  136.1.245.5   S0/0/0     FF RATE
32K

Rack1R4#show ip rsvp installed
RSVP: Serial0/0/0
BPS     To          From          Protoc DPort  Sport  Weight Conversation
32K     150.1.5.5 150.1.4.4       UDP    5000   4000   6      25

Rack1R4#show ip rsvp interface
interface    allocated  i/f max  flow max sub max
S0/0/0         32K        128K     64K      0
```


VOL1 Scenario Reference

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

Bridging and Switching > STP Load Balancing with Port Priority
Bridging and Switching > STP Root Bridge Election.
OSPF > Network Loopback
OSPF > Network Point-to-Multipoint
OSPF > MD5 Authentication
OSPF > Path Selection with Cost
OSPF > Repairing Discontiguous OSPF Areas with Virtual Links
OSPF > OSPF Global Timers
BGP > BGP Bestpath Selection – AP-Path Prepending.
BGP > BGP Communities - No Export
BGP > BGP Bestpath Selection – Weight
BGP > BGP Filtering with Prefix-Lists
BGP > BGP Conditional Advertisement
BGP > AllowAS In
IPv6 > IPv6 Link Local Addressing
IPv6 > Unique Local Addressing
IPv6 > EUI-64 Addressing
IPv6 > IPv6 Tunneling
IPv6 > OSPFv3
MPLS VPN > MPLS LDP
MPLS VPN > VRF Lite
MPLS VPN > MP-BGP VPNv4
MPLS VPN > PE-CE Routing with OSPF
Multicast > IGMP FIltering
Security > Traffic Filtering with Reflexive Access Lists
Security > TCP Intercept
Security > TCP Intercept Watch Mode
Security > DHCP Snooping
Security > DHCP Snooping and Information Option

System Management > HTTP Server and Client
System Management > TFTP Server and Client
System Management > Auto-Install over Frame-Relay
Security > AAA Local Command Authorization
System Management > SNMPv2 Server
System Management > SNMPv2c Access Control
IP Service > GLBP
QoS > Legacy Frame-Relay Traffic Shaping
QoS > Legacy Adaptive FRTS
QoS > MQC Classification and Marking
QoS > MQC Bandwidth Reservations and CBWFQ
Security > Traffic Filtering with Time-Based Access-Lists
QoS > RSVP and WFQ
QoS > RSVP an CBWFQ
QoS > RSVP and Per-VC WFQ

Deep Dive Answers

- How can you make STP ignore the cost of a certain link, or segment connecting multiple switches?
    - You may use Layer 2 protocol tunneling to “bypass” certain links and switches, effectively treating them as one link.
- How could virtual links be used for traffic engineering?
    - OSPF will not use non-backbone area as transit to other area routes unless there is a virtual link across it. Therefore, by deploying virtual links you may affect the routing paths that would have been otherwise inaccessible.
- What is authenticated by OSPF authentication?
    - OSPF authentication is a field in OSPF packet. Therefore, any packet exchanged over a link should have it in order to be accepted by a peer.
- Can OSPF ensure LSA authentication?
    - Strictly speaking OSPF does not authenticate LSAs, it only authenticate adjacencies. Therefore, unless all links in the area are authenticated you cannot ensure that all LSAs received by OSPF process are legit.
- How could you verify if your topology is stable after redistribution?
    - The easiest way is by using the **`debug ip routing`** command to see if there are oscillating routing updates in the network.
- Why using optimum (shortest) paths should be an important result of redistribution?
    - Suboptimal paths, i.e. having to transit another routing domain to reach another, while that domain is directly connected, normally result in routing loops as some routing domains may route back across the redistribution path.
- Is there an alternative solution to using BGP conditional route advertisement?
    - An alternate solution would be using reliable static routes with route tags. The route is only active if the tracked object is up or down (inverse boolean tracking). Two routes are created with different tag values, tracking the same object state: one tracks the object to be up other tracks it being down. Therefore, the route is advertised with different tag depending on the tracked prefix state. The routes sent to neighbors are filtered based on tags: e.g. neighbor 1 only receives the prefix if it’s tagged with 100 and neighbor 2 receives it only if it’s tagged with 200. Effectively, the advertisement is conditioned based on the tracked prefix state.
- What are your options to deploying BGP VPNs if the underlying network does not support MPLS?

  - Since the core of MPLS VPNs is MP-BGP, any tunneling technology would work. You may use GRE or mGRE tunnels or even L2TPv3 tunneling for this purpose.

- Why is it important to force LDP router ID?
  - As soon as router has a loopback with higher IP address it resets the LDP peering session. This will disrupt connectivity, so forcing the LDP router-ID is an important task.
- How does regular traceroute utility works in MPLS VPN environment?
  - When you trace the route from within MPLS VPN, the target and source IP addresses in the UDP probe belong to MPLS VPN. If the UDP packet is dropped in core, the SP has no way to figure out the return route to the VPN source address. Therefore, when a UDP packet is dropped due to TTL expiration, the original label stack found in this packet is applied to ICMP unreachable message. The ICMP unreachable message travels across the MPLS SP core until it reaches the destination site, where it is routed back to the site of origin. Therefore, the traceroute from within the MPLS VPN cannot detect LSP breaks in the SP core. You may find more information about MPLS traceroute in the blog post "MPLS Ping and Traceroute" http://blog.ine.com/2008/11/24/mpls-ping-and-traceroute/
- When would you need to manually use different OSPF domain-IDs on the PE routers?
  - Only in the situations when you have VRF-lite function enabled in the CE routers which prevent OSPF routes with the down-bit set from being considered for routing. If the CE's do not support the OSPF VRF-lite capability you may use different OSPF domain-ID in the PE routers to convert the prefixes to external routes that don't have the down bit set.
- When using dense-mode flooding, what would you need to configure in order to allow the spoke nodes on NBMA segment to exchange multicast traffic?
  - Dense mode does not support PIM NBMA functioning and therefore you cannot bypass the split-horizon rules on the NBMA interfaces in hub-and-spoke topology. The only solution is using a tunnel interface from a spoke to the hub.
- How would you filter a particular multicast source when using IGMPv3?
  - By using extended access-lists with IGMP access-group you may filter both the source and the group for the joined multicast flows.
- What could be an alternative to using multicast TTL scoping?
  - The other procedure is known as administrative scoping, implemented using the command **`ip multicast-boundary`**. This command explicitly specifies the groups that are allowed

across the boundary, as opposed to limiting the propagation radius (TTL) of any given group.

- How do reflective access-lists define sessions?
  - A session is defined as an IP flow based on source/destination IP address pair, source/destination port number pair (TCP/UDP) and ICMP message type for some messages (e.g. echo is reflected as echo-reply). The sessions are timed out based on the global timer specified by the command **`ip reflexive-list timeout`**.
- What is the main scaling limitation of reflexive access-lists?
  - The amount of reflected entries to store. Since all entries are timed out based on the global timer and there is no rate-limiting, a spoofing attack may overload the router with the work required to time out the old entries.
- What is the benefit of Watch mode over Intercept mode in the TCP Intercept feature?
  - It consumes less resources as sessions don't need to be proxies initially, but exposes to protected server to the potential attacker.
- What is the only TCP intercept mode supported by CBAC inspection rules?
  - CBAC only implements the watch mode for TCP intercept, as it put much less load on the router resources.
- How effective could be a router-based TCP intercept feature?
  - Router resources are normally much more limited that those of a typical modern server appliance. It makes sense to use host-based protection against DoS attacks (e.g. SYN cookies) or use special appliances deployed in data centers that allow for wire-speed inspection and packet handling.
- What use of Option 82 you could think of?
  - Normally Option 82 allows for IP address allocation based on the customer's physical point of attachment. A typical use is for two MTUs (buildings) connected to the different ports of the same switch in the same VLAN. Based on the port numbers, you may allocate different portions of the same IP subnets to the customers.
- What possible DHCP attack you can think of?
  - DHCP attacks normally concern with impersonating the DHCP server and providing spoofed default gateway and DNS servers, thus allowing for DNS spoofing and complete packet interception for victim hosts.
- What other security features work together with DHCP Snooping?
  - IP Source Guard and Dynamic ARP inspection will provide protection against IP address spoofing and ARP poisoning attacks.

Together, all three features make the IP networking in Ethernet much secure than by default.

- What if the router was using AAA for authentication, are there any special settings required for HTTP server?
  - Yes, since the HTTP admin needs exec privilege level of 15, you need to make sure the command **`aaa authorization exec default`** local is enabled, otherwise IOS will not consider the local database for privilege authorization.
- Can TFTP work across a packet firewall?
  - No, because the UDP port for data transfer is opened dynamically. A stateful firewall with TFTP protocol inspection is required. The same applies for rsh/rcp utilities.
- How does BOOTP server learn the IP address to allocate for remote client?
  - Based on the **`frame-relay map`** command, there are no BOOTP address pools in IOS.
- How would you disable BOOTP server in IOS router?
  - By using the command **`no ip bootp server`**. This will keep DHCP working but reject the BOOTP messages.
- What other options to local command authorization you can think of?
  - Using RBAC (Role Based Access Control) is a good alternative as it allows for non-hierarchical command nesting, i.e. organizing commands in sets (roles) and assigning them to the users. The main functionality remains the same though, just the command grouping is more flexible.
- How could you prevent user from issuing configuration commands but viewing the running configuration using local command authorization?
  - By giving them access to the respective configuration mode commands but disallowing access to the **`configure terminal`** commands and other similar commands.
- What could be the problems with ARP-based load-balancing?
  - ARP cache issues. Some hosts may not properly migrate to the new virtual MAC addresses due to different ARP timeouts, or the amount of hosts could be less than required to properly load-balance between the AVFs. Using GLBP for load-balancing could be a complicated task due to the underlying ARP technology. Additionally, there could be problems with MAC-address table expiration in the switches and unicast flooding due to the asymmetric routing promoted by GLBP.
- What happens if an AVF goes down and some hosts are still using its virtual MAC address for ARP entries?

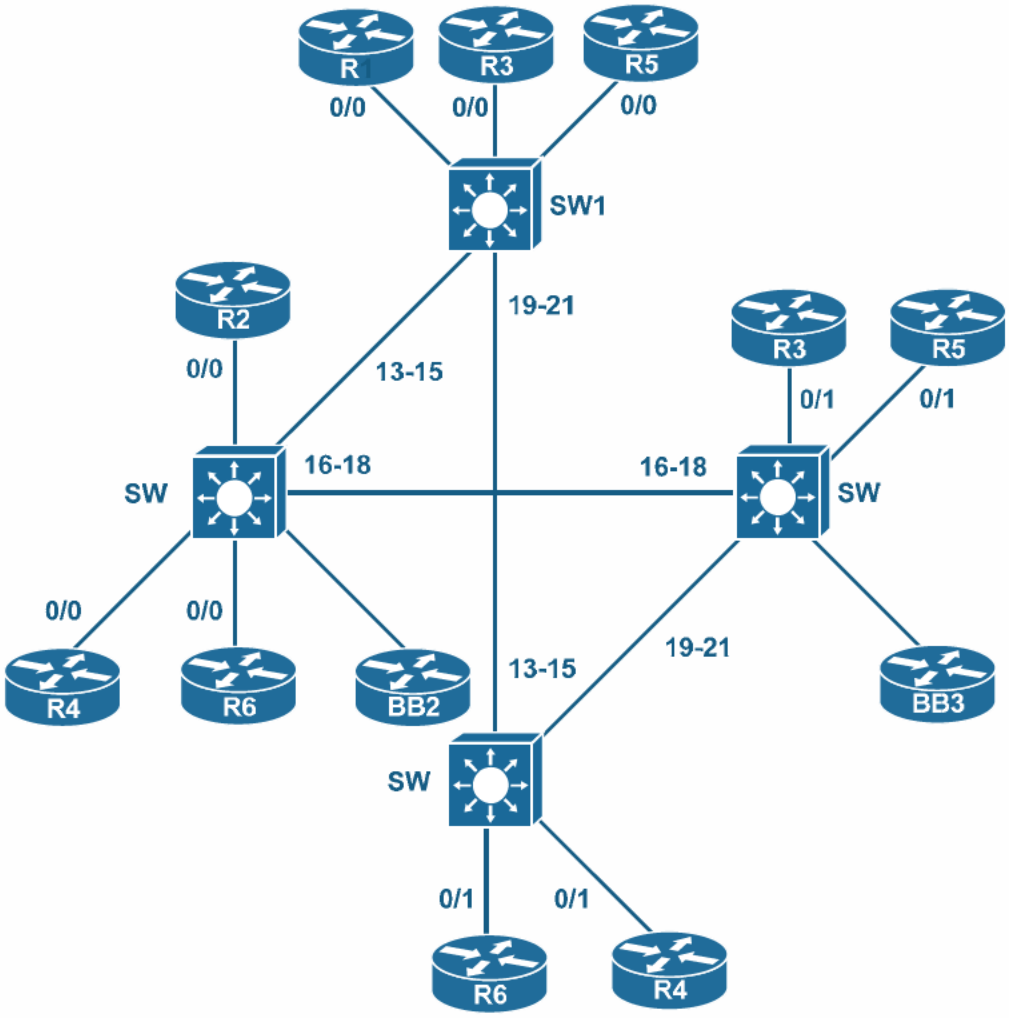
  - GLBP protocol implements MAC address interception, where the failed AVF's virtual MAC is automatically takes by a backup AVF, which keeps servicing two MAC addresses for some time, while migrating the hosts off the failed virtual MAC.

- What other congestion notification FRTS may respond to?
  - Interface congestion (i.e. non-empty TX-ring) could be used to signal the throttling of PVC sending rates. Unlike BECN signaling this technique related only on a local condition, not on the global network state.
- What reservation styles RSVP supports?
  - Fixed Filter (FF) when a receiver specifies all senders explicitly and asks to install a separate reservation for every one of them. Shared Explicit (SE) – when the receiver requests explicit list of senders but asks the network to install one shared reservation for them all. Finally, Wilcard Filter (WF) does not specify any senders, only the flow destination/port and allow all senders to share the same requested reservation. The flow in RSVP is treated as unidirectional with only the destination IP address (e.g. multicast group for videoconference) and destination port set in combination with protocol. Notice that multiple senders only make sense in case where the session destination is multicast.
- What good use could be of RSVP in modern networks?
  - RSVP is normally used for end-to-end admission control to ensure the network has enough resources to accommodate a new session, e.g. a VoIP call. It does not normally specify the QoS policy, but only performs the admission control. Another example of the use is for signaling MPLS TE tunnels end-to-end in a network.
- Is there a situation where RSVP scales well?
  - Yes, normally with multicast flows RSVP can keep the amount of state/reservation in the network to minimum thanks to the state merging in case of shared reservations.

Lab 4 Solutions

Preliminary Tasks

Creating a Layer 2 diagram is recommended for this scenario, since layer 2 traffic manipulation is required in some tasks.



Task 1.1 Solution

🕮 Further Learning

Before you begin this scenario you should have solid understanding of VTP and VTP Pruning features. You may find introductory information in VOL1 scenarios **Bridging and Switching > VTP** and **Bridging and Switching > VTP Pruning**, **Bridging and Switching > VTP Pruning Eligible List**.

```
SW1:
!
! Enable pruning in the VTP server and change pruning eligible list
!
vtp pruning
!
interface FastEthernet 0/13
 switchport trunk pruning vlan 2-7,9-1001
interface range FastEthernet 0/14-15
 switchport trunk allowed vlan except 8

SW3:
!
! Configure VTP pruning list
!
interface FastEthernet 0/16
 switchport trunk pruning vlan 2-7,9-1001
interface range FastEthernet 0/17-18
 switchport trunk allowed vlan except 8
```

Task 1.1 Breakdown

Since all 4 switches are VTP servers, pruning could be enabled on any of them. When a trunk is created all VLANs throughout the VTP domain may transit the trunk link. These VLANs are said to be in the *allowed vlan* list. In the same manner, when VTP pruning is enabled, all non-default VLANs can be pruned off of a trunk link (default VLANs such as 1 and 1002-1005 cannot be pruned). These VLANs are said to be *prune eligible*. In certain cases, such as when dealing with switches in transparent mode, it is not desirable to have a switch send pruning information out a specific trunk link. Since pruning can only be enabled or disabled globally, manually editing the prune eligible list is the only way to achieve the desired effect.

To edit the prune eligible list, use the interface level command **switchport trunk pruning vlan [add | remove | none | except] [num]**. To verify what is prune eligible on an interface, issue the **show interface [int] switchport** command. By default VLANs 2-1001 are prune eligible.

Task 1.1 Verification

Check the pruning list before and after you removed VLAN 8 from there:

```
Rack1SW1#show interface fa0/13 switchport | include Pruning
Pruning VLANs Enabled: 2-1001  ← Prune eligible list

Rack1SW1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW1(config)#interface fa0/13
Rack1SW1(config-if)#switchport trunk pruning vlan 2-7,9-1001
                                                  ↑
                          Delete vlan 8 from the prune eligible list
Rack1SW1(config)#^Z
Rack1SW1#show interface fa0/13 switchport | include Pruning
Pruning VLANs Enabled: 2-7,9-1001 ← VLAN 8 can no longer be pruned

Rack1SW1#show interfaces fastEthernet 0/14 switchport | include
Trunking
Administrative Trunking Encapsulation: dot1q
Operational Trunking Encapsulation: dot1q
Negotiation of Trunking: On
Trunking Native Mode VLAN: 255 (VLAN_BEE)
Trunking VLANs Enabled: 1-7,9-4094
SW1#
```

Verify that VLAN 8 is not allowed on Fa0/14 and Fa0/15; check that VLAN 8 cannot be pruned on Fa0/13:

```
Rack1SW1#show interfaces trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      on           802.1q         trunking      255
Fa0/14      on           802.1q         trunking      255
Fa0/15      on           802.1q         trunking      255
Fa0/19      on           802.1q         trunking      255
Fa0/20      on           802.1q         trunking      255
Fa0/21      on           802.1q         trunking      255

Port        Vlans allowed on trunk
Fa0/13      1-4094
Fa0/14      1-7,9-4094
Fa0/15      1-7,9-4094
Fa0/19      1-4094
Fa0/20      1-4094
Fa0/21      1-4094

Port        Vlans allowed and active in management domain
Fa0/13      1,6-8,12,36,43,45,77,88,255,258
Fa0/14      1,6-7,12,36,43,45,77,88,255,258
Fa0/15      1,6-7,12,36,43,45,77,88,255,258
Fa0/19      1,6-8,12,36,43,45,77,88,255,258
Fa0/20      1,6-8,12,36,43,45,77,88,255,258
```



```
Fa0/21      1,6-8,12,36,43,45,77,88,255,258

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      1,6,8,12,36,43,45,88,255,258
Fa0/14      none
Fa0/15      none
Fa0/19      1
Fa0/20      1
Fa0/21      1

Rack1SW1#show interfaces fastEthernet 0/13 pruning

Port        Vlans pruned for lack of request by neighbor
Fa0/13      7,77

Port        Vlan traffic requested of neighbor
Fa0/13      1,7-8,12,45,77
```

🕮 Deep Dive

- Why VTP might be impractical in modern switching environment?
- What is the danger of using VTP?

Task 1.2 Solutions

Before You Begin

The scenario uses layer 2 traffic engineering based on STP cost manipulation. For more information on this feature we recommend you completing the following VOL1 scenarios: **Bridging and Switching > STP Load Balancing with Port Cost**.

```
SW1:
!
! Make SW1 the root bridge
!
spanning-tree vlan 258 root primary

SW2:
!
! Change
!
interface Fa0/13
 spanning-tree vlan 258 cost 100
!
interface Fa0/14
 spanning-tree vlan 258 cost 100
!
interface Fa0/15
 spanning-tree vlan 258 cost 100

SW3:
spanning-tree vlan 258 root secondary
```

Task 1.2 Breakdown

As previously discussed, the two user defined variables that can be used to affect the spanning-tree root port selection are port-cost and port-priority. The above task specifies to "use the fewest number of commands to accomplish this task and do not alter SW1's port-priority." Since SW1 is the root of the spanning-tree, the appropriate value to change is the spanning tree cost for VLAN 258 on SW2 .

Note

To affect how the local switch elects its root port change the spanning-tree port-cost. Cost is cumulative throughout the STP domain.

To affect how a downstream switch elects its root port, change the spanning-tree port-priority. Port-priority is only locally significant between two directly connected bridges.

Task 1.2 Verification

```
Rack1SW2#show spanning-tree vlan 258

VLAN0258
  Spanning tree enabled protocol ieee
  Root ID     Priority    24834
              Address     000a.f4f3.e780
              Cost        10
              Port        18 (FastEthernet0/16) ← Root port
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

<output deleted>

Interface         Role Sts Cost      Prio.Nbr Type
----------------  ---- --- --------- -------- --------------------
Fa0/2             Desg FWD 19        128.4    P2p
Fa0/13            Altn BLK 100       128.15   P2p
Fa0/14            Altn BLK 100       128.16   P2p
Fa0/15            Altn BLK 100       128.17   P2p
Fa0/16            Root FWD 19        128.18   P2p
Fa0/17            Altn BLK 19        128.19   P2p
Fa0/18            Altn BLK 19        128.20   P2p
```

Verify that SW1 is the root bridge for VLAN 258; root bridge has all ports in Desg FWD role/state:

```
Rack1SW1#show spanning-tree vlan 258

VLAN0258
  Spanning tree enabled protocol ieee
  Root ID     Priority    24834
              Address     001f.6ce6.8700
              This bridge is the root
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    24834  (priority 24576 sys-id-ext 258)
              Address     001f.6ce6.8700
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 15

Interface         Role Sts Cost      Prio.Nbr Type
----------------  ---- --- --------- -------- --------------------------
Fa0/13            Desg FWD 19        128.15   P2p
Fa0/14            Desg FWD 19        128.16   P2p
Fa0/15            Desg FWD 19        128.17   P2p
Fa0/19            Desg FWD 19        128.21   P2p
Fa0/20            Desg FWD 19        128.22   P2p
Fa0/21            Desg FWD 19        128.23   P2p
```


Verify that SW3 will become root bridge in case SW1 fails:

```
Rack1SW3#show spanning-tree vlan 258

VLAN0258
  Spanning tree enabled protocol ieee
  Root ID     Priority    24834
              Address     001f.6ce6.8700
              Cost        38
              Port        21 (FastEthernet0/19)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    28930  (priority 28672 sys-id-ext 258)
              Address     0018.7392.0980
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------
Fa0/5               Desg FWD 19        128.7    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
Fa0/19              Root FWD 19        128.21   P2p
Fa0/20              Altn BLK 19        128.22   P2p
Fa0/21              Altn BLK 19        128.23   P2p
```

🕮 Deep Dive

- Will changing cost affect any downstream bridges?
- If you are not allowed to change the STP link cost, what other parameter you may tune to achieve the same effect?

Task 1.3 Solutions

🕮 Before You Begin

Unidirectional Link Detection is the feature normally used on optical links to prevent loss of single strand from causing Layer 2 Loops. You may find more information completing VOL1 scenario **Bridging and Switching > Unidirectional Link Detection**. Additionally, you may read the post named "UDLD Modes of Operation" on the INE blog:

http://blog.ine.com/2008/07/05/udld-modes-of-operation/

STP Loopguard is a complimentary STP feature that may detect unidirectional link condition based on STP solely. It is coveredin VOL1 scenario **Bridging and Switching > STP Loop Guard**.

SW1:

```
!
! Enable UDP and Loopguard on SW1, as this is the downstream bridge.
!
interface FastEthernet0/15
 udld port aggressive
 spanning-tree guard loop
```

SW2:

```
interface FastEthernet0/15
 udld port aggressive
```


Task 1.3 Breakdown

Unidirectional Link Detection (UDLD) is used to detect when the send channel of a cable is down, but not the receive channel, and vice versa. This situation typically can occur in a fiber optic cable when there is a break in one side of the cable run. When UDLD detects this situation, the interface is brought down and a log message is generated. This feature is useful to prevent against spanning-tree loops and traffic black holes due to unidirectional links. To enable UDLD on fiber optic interfaces, issue the *global* configuration command **udld enable** or **udld aggressive**. To enable UDLD on a copper interface, issue the *interface* command **udld port aggressive**.

In certain cases, a spanning-tree loop can occur when the send channel of a designated port is damaged. This will cause the bridge on the other side of the link to stop receiving STP BPDUs. When this occurs, the non-designated port assumes that it should become the designated port, and can eventually result in a loop in the topology. In order to prevent this case, spanning-tree loopguard will transition the non-designated port to loop-inconsistent state, and will not pass user traffic when this problem occurs. To enable loopguard, use the interface level command **spanning-tree guard loop.**

Although UDLD aggressive will put the port into "err-disabled" state upon unidirectional link detection, the task specifically asks for port Fa0/15 not to be elected as designated port (SW1 is the root bridge for VLAN 258 so the task refers to all other VLANs). This means we need to configure STP loop guard feature in addition to the above UDLD settings.

Pitfall

The global command **udld enable** only applies to fiber interfaces. Ensure to use the *interface* command **udld port aggressive** for copper interfaces.

Task 1.3 Verification

Verify the UDLD configuration:

```
Rack1SW2#show udld fa0/15

Interface Fa0/15
---
Port enable administrative configuration setting: Enabled / in
aggressive mode
Port enable operational state: Enabled / in aggressive mode
Current bidirectional state: Bidirectional
Current operational state: Advertisement - Single neighbor detected
Message interval: 7
Time out interval: 5
<output omitted>
```

Verify loop guard status on SW1 Fa0/15:

```
Rack1SW1#show spanning-tree interface fa0/15 detail
<output omitted>
   Link type is point-to-point by default
   Loop guard is enabled on the port
   BPDU: sent 1234, received 21
<output omitted>
```

🕮 Deep Dive

- Why twisted-pair connections are not suspectible to the unidirectional link problem?

Task 1.4 Solutions

🕮 Before You Begin

Modifying port STP priority is an option to change STP blocked link selection. There is a practical example in VOL1 lab **`Bridging and Switching > STP Load Balancing with Port Priority`** featuring detailed breakdown for this feature use.

SW4:
```
interface FastEthernet0/21
 spanning-tree vlan 258 port-priority 16
```

Task 1.4 Breakdown

As mentioned earlier, the two common methods to affect the spanning-tree path to the root are cost and port-priority. The key to remembering which one to use where is to understand the direction these two options affect in regards to spanning tree. When going away from the root of the tree, use port-priority. When going towards the root of the tree, use cost.

Task 1.4 Verification

Check the spanning-tree status for VLAN 258:

```
Rack1SW3#show spanning-tree vlan 258 | in Fa0/21
Fa0/21           Root FWD 19        128.21   P2p
```

Task 1.5 Solutions

🕮 Before You Begin

For more information about the Storm-Control feature, refer to VOL1 scenario **`Bridging and Switching > Storm Control`**.

```
SW1:
interface FastEthernet0/1
 storm-control unicast level pps 250
```

Task 1.5 Breakdown

Storm control limits the amount of unicast, multicast, or broadcast traffic that is received on a layer 2 switchport. When the threshold of unicast or broadcast traffic is exceeded, traffic in excess of the threshold is dropped. When the multicast threshold is exceeded, all unicast, multicast, or broadcast traffic is dropped until the level falls below the threshold. To configure storm-control, issue the **`storm-control [unicast | broadcast | multicast] level [level]`** interface level command.

☠ Pitfall

Do not assume that the task title will directly indicate the solution. In this case, the title of the task is “Rate-Limiting”, but the solution used is storm-control.

🕮 Deep Dive

- Is there a reason to prefer storm-control over policing to rate-limit ingress traffic?
- How does storm-control feature meter the incoming traffic?

Task 1.5 Verification

Check the storm-control status:

```
Rack1SW1#show storm-control unicast
Interface  Filter State   Upper        Lower        Current
---------  -------------  -----------  -----------  ----------
Fa0/1      Forwarding        250 pps      250 pps        0 pps
```


Task 1.6 Solution

🕮 Before You Begin

To better understand MLS QoS mapping tables and their purpose you may want to read the following post on the INE blog:

http://blog.ine.com/2008/10/30/traffic-classification-in-the-35503560-switches/

And complete VOL1 scenario **QoS > Catalyst QoS Port-Based Classification**.

SW2:
```
mls qos map ip-prec-dscp 0 0 0 0 32 40 0 0
```

Task 1.6 Breakdown

The switches use internal DSCP mappings for classification and marking of frames as they traverse the switch. Mappings are used to help determine how the switch will handle the frame. To alter the default mappings between the internal DSCP and the IP precedence, the **mls qos map ip-prec-dscp <8 DSCP values>** command is used.

🕮 Deep Dive

- When would you use the IP precedence to DSCP table?
- How could the DSCP to CoS mapping table be used to set CoS marking on non-IP packets?

Task 1.6 Verification

List the mapping table content:

```
Rack1SW2#show mls qos maps ip-prec-dscp
   IpPrecedence-dscp map:
     ipprec:    0  1  2  3  4  5  6  7
     --------------------------------
       dscp:    0  0  0  0 32 40  0  0
```

Task 1.7 Solution

🕮 Before You Begin

To better understand MLS QoS mapping tables and their purpose you may want to read the following post on the INE blog:

http://blog.ine.com/2008/10/30/traffic-classification-in-the-35503560-switches/

And complete VOL1 scenario **QoS > Catalyst QoS Port-Based Classification**.

```
SW2:
mls qos
interface FastEthernet0/2
 mls qos trust ip-precedence
```

Task 1.7 Breakdown

By configuring the interface to trust IP precedence, the IP precedence to DCSP map created in the previous task will be used to map ingress packet's IP precedence values to the internal DSCP values. If you forget to enable QoS with the command **mls qos**, or your output may look like this:

```
Rack1SW2#show mls qos interface fa0/2
QoS is disabled. When QoS is enabled, following settings will be
applied
trust state: trust ip-precedence
trust mode: trust ip-precedence
trust enabled flag: ena
COS override: dis
default COS: 0
DSCP Mutation Map: Default DSCP Mutation Map
Trust device: none
qos mode: port-based
```


Task 1.7 Verification

Check the IP precedence trust status:

```
Rack1SW2#show mls qos interface fa0/2 | include ip-precedence
trust state: trust ip-precedence
trust mode: trust ip-precedence
```

🕮 Deep Dive

- What is the difference in trusting IP precedence and DSCP values?
- How is IP Precedence QoS model different from DiffServ?

Task 2.1 Solution

🕮 Before You Begin

The scenario makes use of OSPF NSSA areas. More information about OSPF stub area types could be found in the following VOL1 scenarios:

OSPF > Not So Stubby Areas
OSPF > Not So Stubby Areas and Default Routing

R2:

```
!
!R2 and R5 are ABRs, and therefore require "no-summary"
!
router ospf 1
 area 1 nssa no-summary
```

R5:

```
router ospf 1
 area 1 nssa no-summary
```

SW2:

```
router ospf 1
 area 1 nssa
```


Task 2.1 Breakdown

The task requirements clearly point toward the use of OSPF stub area. However, if you inspect SW2's configuration you'll find that route redistribution is configured there. You may opt to remove the redistribution, but normally the initial configurations in the CCIE exams should not be modified, unless required for troubleshooting. Therefore, your only option is using the NSSA area. The use of "no-summary" option on the ABRs ensures that no summary routes are injected in the NSSA area, just the default route to reach all destinations outside the area.

In order to properly compute the shortest path first (SPF) algorithm, routers within a link-state area must have a consistent view of the link state topology. For this reason, link-state protocols such as OSPF and IS-IS do not support the removal of a link state advertisement (LSA) from the link-state database on a per router basis. Instead, this must be done on a per link-state area basis. In OSPF, this is accomplished by the various stub area definitions.

By preventing certain types of LSAs from entering an area, the various stub area types can be used to reduce the amount of forwarding information required to be in both the OSPF database and the IP routing table. Such cases may be advantageous when there is only one exit point out of an area, or only one exit point out of the autonomous system. In such a design, it may be feasible to replace specific forwarding information with default information, hence reducing memory utilization and speeding up the routing table lookup process. There are four OSPF stub area definitions. These are stub, totally stubby, not-so-stubby (NSSA), and not-so-totally-stubby.

To understand why certain LSAs are removed from an area, you must first understand what each LSA type accomplishes. LSA types are defined as follows:

LSA	Name	Description
1	Router LSA	Generated by all routers in an area to describe their directly attached links (intra-area routes). Does not leave the area.
2	Network LSA	Generated by the DR of a broadcast or non-broadcast segment to describe the neighbors connected to that segment. Does not leave the area.
3	Summary LSA	Generated by the area border router (ABR) to describe a route to neighbors outside the area (inter-area route).
4	Summary LSA	Generated by the ABR to describe a route to an autonomous system border router (ASBR) to neighbors outside the area.
5	External LSA	Generated by the ASBR to describe routes redistributed into the area. These routes appear as E1 or E2 in the IP routing table. E2 (default) uses a static cost throughout the OSPF domain, as it only takes the cost into account that is reported at redistribution. E1 uses a cumulative cost of the cost reported into the OSPF domain at redistribution plus the local cost to the ASBR.
6	Multicast LSA	Used in multicast OSPF. Not supported by Cisco.
7	NSSA External LSA	Generated by an ASBR inside a not-so-stubby (NSSA) area to describe routes redistributed into the NSSA area. LSA 7 is translated into LSA 5 as it leaves the NSSA area. These routes appear as N1 or N2 in the IP routing table inside the NSSA area. Much like LSA 5, N2 is a static cost while N1 is a cumulative cost that includes the cost up to the ASBR.

A stub area blocks OSPF external routes (LSAs 4 & 5) from entering the area. The ABR of a stub area automatically generates a default route (LSA 3) into the stub area. A stub area is defined by issuing the **`area [area_id] stub`** routing process subcommand on all devices in the stub area.

A totally stubby area is a stub area that in addition to blocking OSPF external routes blocks OSPF inter-area routes (LSA 3). The ABR of a totally stubby area automatically generates a default route (LSA 3) into the stub area. Redistribution into stub and totally stubby areas is not permitted. A totally stubby area is defined by issuing the **`area [area_id] stub no-summary`** routing process subcommand on all ABRs of the stub area.

The not-so-stubby area (NSSA) overcomes the problem of not being able to redistribute into a stub area. Like a stub area, a not-so-stubby area blocks OSPF external routes (LSAs 4 & 5) from entering the area. However, redistribution is allowed into the NSSA area. These routes are redistributed as NSSA external (LSA 7) and are different than normal LSA 5 external routes. As these LSA 7 prefixes leave the NSSA area, the ABR translates them into LSA 5. In other words, routers outside the NSSA area do not know that these routes were redistributed into an NSSA area, but instead simply see them as LSA 5 external routes. A not-so-stubby area is defined by issuing the **area [area_id] nssa** routing process subcommand on all routers in the stub area.

Another difference between the stub area and the not-so-stubby area is that the ABR of the NSSA does not automatically originate a default route into the area. A default route may be originated into an NSSA by adding the **default-originate** keyword onto the **area [area_id] nssa** statement. This default is type 7 LSA.

The not-so-totally-stubby area combines the concept of the totally stubby area and the not-so-stubby area. The not-so-totally-stubby area blocks both OSPF external (LSA 5) and inter-area (LSA 3 & 4) routes from entering the area. The ABR of the not-so-totally-stubby area automatically generates a default route (LSA 3) into the not-so-totally-stubby area. Redistribution into the not-so-totally-stubby area is permitted. A not-so-totally-stubby area is defined by issuing the **area [area_id] nssa no-summary** routing process subcommand on all ABRs in the stub area.

Note

All routers in a stub or not-so-stub are must agree on the stub or NSSA flag. It is the ABR(s) of the stub area or NSSA area that determine if it is totally stubby or not-so-totally stubby by adding the **no-summary** keyword on to the appropriate stub command.

The stub area types can be summarized as follows:

Stub Type	Keyword	LSAs	Default Injected
Stub	area x stub	1,2,3	YES
Totally Stubby	area x stub no-summary	1,2, default of 3	YES
Not-So-Stub	area x nssa	1,2,3,7	NO
Not-So-Totally-Stubby	area x nssa no-summary	1,2, default of 3, 7	YES

🕮 Deep Dive

- What could be the purpose of OSPF NSSA area in real-word deployments?

Task 2.1 Verification

Verify area configuration and type 7 to type 5 translated prefixes:

```
Rack1R2#show ip ospf | begin Area 1
    Area 1
        Number of interfaces in this area is 1
        It is a NSSA area
        Perform type-7/type-5 LSA translation
<output omitted>

Rack1R2#show ip route ospf | include 150.1.8.0
O N2    150.1.8.0/24 [110/20] via 141.1.0.8, 00:23:07, FastEthernet0/0

Rack1R1#show ip route ospf | include 150.1.8.0
O E2    150.1.8.0/24 [110/20] via 141.1.123.2, 00:27:24, Serial0/0.1
```

Verify that SW2 receives only the default route:

```
Rack1SW2#show ip route ospf
O*IA 0.0.0.0/0 [110/2] via 141.1.0.2, 00:02:09, Vlan258
```


Task 2.2 Solution

🕮 Before You Begin

This task core idea is using a tunnel across a stub area to emulate virtual link functionality. If you need additional information about virtual links you may want to check VOL1 scenarios **OSPF > Repairing Discontiguous Area with Virtual Links**

R2:

```
!
! Tunnel is needed to replace a virtual link
! As the area is stubby and does not support VLs
!
interface Tunnel0
 ip address 141.1.25.2 255.255.255.0
 tunnel source FastEthernet0/0
 tunnel destination 141.1.0.5
!
router ospf 1
 network 141.1.25.2 0.0.0.0 area 0
```

R3:

```
router ospf 1
 redistribute rip subnets
!
router rip
 redistribute ospf 1 metric 1
```

R4:

```
!
! Using OSPF P2P network is easier from config standpoint
!
interface Serial0/0/0
 ip ospf network point-to-point
!
router ospf 1
 router-id 150.1.4.4
 network 141.1.45.4 0.0.0.0 area 2
 network 141.1.54.4 0.0.0.0 area 2
 network 141.1.145.4 0.0.0.0 area 2
 network 150.1.4.4 0.0.0.0 area 2
```



```
R5:
interface Serial0/0/0
 ip ospf network point-to-point
!
interface Tunnel0
 ip address 141.1.25.5 255.255.255.0
 tunnel source FastEthernet0/1
 tunnel destination 141.1.0.2
!
router ospf 1
 router-id 150.1.5.5
 network 141.1.25.5 0.0.0.0 area 0
 network 141.1.45.5 0.0.0.0 area 2
 network 141.1.54.5 0.0.0.0 area 2
 network 141.1.145.5 0.0.0.0 area 2
 network 150.1.5.5 0.0.0.0 area 2
```

Task 2.2 Breakdown

Pitfall

A stub area cannot be used as a transit area for a virtual-link. Therefore, a GRE tunnel configured with OSPF area 0 is created between R2 and R5. This is due to the fact that OSPF area 2 is discontiguous from OSPF area 0. Typically this problem is fixed by creating a virtual-link back to connect area 0 with the discontiguous area. However, since in this case area 1 (the transit area) is stub, this method will not work. Therefore a virtual connection (GRE tunnel) is created between R2 and R5 to run OSPF area 0.

Task 2.2 Verification

Verify that tunnel is up and running in OSPF area 0:

```
Rack1R5#show interfaces tu0
Tunnel0 is up, line protocol is up
  Hardware is Tunnel
  Internet address is 141.1.25.5/24
<output omitted>

Rack1R5#show ip ospf interface tu0
Tunnel0 is up, line protocol is up
  Internet Address 141.1.25.5/24, Area 0
  Process ID 1, Router ID 150.1.5.5, Network Type POINT_TO_POINT, Cost:
11111
  Transmit Delay is 1 sec, State POINT_TO_POINT,
<output omitted>

Rack1R5#ping 141.1.25.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 141.1.25.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/10/36 ms
```

Next verify OSPF neighbors on the tunnel:

```
Rack1R5#show ip ospf neighbor tu0

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.2.2         0   FULL/  -        00:00:38    141.1.25.2      Tunnel0
```

Finally verify that we are seeing R5 and R2 Loopback0 prefixes as inter-area summary prefixes on R1:

```
Rack1R1#show ip route ospf  | include (150.1.4.4|150.1.5.5)
O IA    150.1.5.5/32 [110/11176] via 141.1.123.2, 01:44:25, Serial0/0.1
O IA    150.1.4.4/32 [110/11186] via 141.1.123.2, 01:44:19, Serial0/0.1
```


Verify that RIP prefixes are redistributed into OSPF:

```
Rack1R3#show ip route rip
     141.1.0.0/24 is subnetted, 14 subnets
R       141.1.6.0 [120/1] via 141.1.36.6, 00:00:21, FastEthernet0/1
R       141.1.7.0 [120/1] via 141.1.37.7, 00:00:10, FastEthernet0/0
R       141.1.77.0 [120/1] via 141.1.37.7, 00:00:10, FastEthernet0/0
     150.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
R       150.1.7.0/24 [120/1] via 141.1.37.7, 00:00:10, FastEthernet0/0
R       150.1.6.0/24 [120/1] via 141.1.36.6, 00:00:21, FastEthernet0/1

Rack1R2#show ip route ospf | i 141.1.123.3
O E2    141.1.6.0 [110/20] via 141.1.123.3, 00:03:15, Serial0/0
O E2    141.1.7.0 [110/20] via 141.1.123.3, 00:03:15, Serial0/0
O E2    141.1.36.0 [110/20] via 141.1.123.3, 00:03:15, Serial0/0
O E2    141.1.37.0 [110/20] via 141.1.123.3, 00:03:15, Serial0/0
O E2    141.1.77.0 [110/20] via 141.1.123.3, 00:03:15, Serial0/0
O E2    150.1.7.0/24 [110/20] via 141.1.123.3, 00:03:15, Serial0/0
O E2    150.1.6.0/24 [110/20] via 141.1.123.3, 00:03:15, Serial0/0
O       150.1.3.3/32 [110/65] via 141.1.123.3, 00:03:44, Serial0/0
```

Verify that OSPF prefixes are redistributed into RIP:

```
Rack1R3#show ip route ospf
     141.1.0.0/24 is subnetted, 14 subnets
O IA    141.1.145.0 [110/11893] via 141.1.123.2, 00:04:59, Serial1/0.1
O IA    141.1.255.0 [110/783] via 141.1.123.2, 00:04:59, Serial1/0.1
O IA    141.1.8.0 [110/783] via 141.1.123.2, 00:04:59, Serial1/0.1
O IA    141.1.0.0 [110/782] via 141.1.123.2, 00:04:59, Serial1/0.1
O       141.1.25.0 [110/11892] via 141.1.123.2, 00:04:59, Serial1/0.1
O IA    141.1.45.0 [110/11956] via 141.1.123.2, 00:04:59, Serial1/0.1
O IA    141.1.54.0 [110/11956] via 141.1.123.2, 00:04:59, Serial1/0.1
O IA    141.1.88.0 [110/783] via 141.1.123.2, 00:04:59, Serial1/0.1
O IA 192.10.1.0/24 [110/782] via 141.1.123.1, 00:04:59, Serial1/0.1
     150.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
O IA    150.1.5.5/32 [110/11893] via 141.1.123.2, 00:04:59, Serial1/0.1
O IA    150.1.4.4/32 [110/11894] via 141.1.123.2, 00:03:55, Serial1/0.1
O       150.1.2.2/32 [110/782] via 141.1.123.2, 00:04:59, Serial1/0.1
O       150.1.1.1/32 [110/782] via 141.1.123.1, 00:04:59, Serial1/0.1
O E2    150.1.8.0/24 [110/20] via 141.1.123.2, 00:04:36, Serial1/0.1
```



```
Rack1SW1#show ip route rip
     141.1.0.0/24 is subnetted, 14 subnets
R       141.1.145.0 [120/1] via 141.1.37.3, 00:00:20, FastEthernet0/3
R       141.1.255.0 [120/1] via 141.1.37.3, 00:00:20, FastEthernet0/3
R       141.1.8.0 [120/1] via 141.1.37.3, 00:00:20, FastEthernet0/3
R       141.1.6.0 [120/2] via 141.1.37.3, 00:00:20, FastEthernet0/3
R       141.1.0.0 [120/1] via 141.1.37.3, 00:00:20, FastEthernet0/3
R       141.1.25.0 [120/1] via 141.1.37.3, 00:00:20, FastEthernet0/3
R       141.1.45.0 [120/1] via 141.1.37.3, 00:00:20, FastEthernet0/3
R       141.1.36.0 [120/1] via 141.1.37.3, 00:00:21, FastEthernet0/3
R       141.1.54.0 [120/1] via 141.1.37.3, 00:00:21, FastEthernet0/3
R       141.1.88.0 [120/1] via 141.1.37.3, 00:00:21, FastEthernet0/3
R       141.1.123.0 [120/1] via 141.1.37.3, 00:00:21, FastEthernet0/3
R    192.10.1.0/24 [120/1] via 141.1.37.3, 00:00:21, FastEthernet0/3
     150.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
R       150.1.6.0/24 [120/2] via 141.1.37.3, 00:00:21, FastEthernet0/3
R       150.1.3.0/24 [120/1] via 141.1.37.3, 00:00:22, FastEthernet0/3
R       150.1.5.5/32 [120/1] via 141.1.37.3, 00:00:22, FastEthernet0/3
R       150.1.4.4/32 [120/1] via 141.1.37.3, 00:00:22, FastEthernet0/3
R       150.1.2.2/32 [120/1] via 141.1.37.3, 00:00:22, FastEthernet0/3
R       150.1.1.1/32 [120/1] via 141.1.37.3, 00:00:22, FastEthernet0/3
R       150.1.8.0/24 [120/1] via 141.1.37.3, 00:00:22, FastEthernet0/3
```


Now validate full-reachability using the TCL script below:

```
foreach i { 150.1.1.1
192.10.1.1
141.1.123.1
141.1.25.2
150.1.2.2
141.1.0.2
141.1.123.2
150.1.3.3
141.1.36.3
141.1.37.3
141.1.123.3
141.1.45.4
141.1.46.4
141.1.54.4
141.1.145.4
150.1.4.4
141.1.0.5
141.1.25.5
141.1.45.5
141.1.54.5
141.1.145.5
150.1.5.5
141.1.6.6
141.1.36.6
141.1.46.6
150.1.6.6
150.1.7.7
141.1.7.7
141.1.37.7
141.1.77.7
141.1.255.8
150.1.8.8
141.1.8.8
141.1.0.8
141.1.88.8
141.1.255.9
141.1.255.9
141.1.255.10
} { puts "[ exec ping $i ]" }
```


Notice that the IP addresses used for the VPN addressing on R4 and R6 are excluded from this test.

🕮 Deep Dive

- What other way you could use in place of virtual-link or tunnel to connect a non-zero area to OSPF backbone?

Task 3.1 Solutions

🕮 Before You Begin

If you need more information about OSPFv3 configuration, refer to VOL1 scenarios **IPv6 > OSPFv3** and **IPv6 > OSPFv3 over NBMA**

R1:
```
ipv6 unicast-routing
!
! Configure IP addressing and enable OSPFv3 area
!
interface Serial0/0.1 point-to-point
 ipv6 address 2001:141:1:12::1/64
 ipv6 address FE80::1 link-local
 ipv6 ospf 1 area 0
```

R2:
```
ipv6 unicast-routing
!
interface FastEthernet0/0
 ipv6 address 2001:141:1:25::2/64
 ipv6 ospf 1 area 1
!
interface Serial0/0
 ipv6 address 2001:141:1:12::2/64
 ipv6 address FE80::2 link-local
 ipv6 ospf network point-to-point
 ipv6 ospf 1 area 0
 frame-relay map ipv6 FE80::1 201
 frame-relay map ipv6 2001:141:1:12::1 201 broadcast
```

R5:
```
ipv6 unicast-routing
!
interface FastEthernet0/1
 ipv6 address 2001:141:1:25::5/64
 ipv6 ospf 1 area 1
```

SW2:
```
sdm prefer dual-ipv4-and-ipv6 default
!
ipv6 unicast-routing
!
interface VLAN258
 ipv6 address 2001:141:1:25::8/64
 ipv6 ospf 1 area 1
```


Task 3.1 Breakdown

Note

Output shown in this breakdown is general information regarding OSPFv3, and is not specific to this lab. For output specific to this lab, see the verification section.

The first step in enabling OSPF for IPv6 (OSPFv3) is to enable IPv6 routing globally with the **ipv6 unicast-routing** command, followed by the **ipv6 router ospf 1 [process-id]** command, where *process-id* is a locally significant number similar to OSPFv2 for IPv4. Unlike OSPFv2, OSPFv3 does not use the network statement to enable the process on an interface. Instead, the interface level command **ipv6 ospf [process-id] area [area-id]** command is used. Once OSPFv3 has been configured, issue the **show ipv6 ospf neighbor** command to verify adjacency status.

```
Rack1R1#show ipv6 ospf neighbor

Neighbor ID  Pri  State          Dead Time   Interface ID    Interface
222.22.2.1   1    FULL/DR        00:00:30    12
FastEthernet0/0
```

In the above output, we can see that R1 has formed an adjacency with an OSPFv3 router with the router-id 222.22.2.1. This router has a priority of 1 and has been elected the designated router. Although the above output relates specifically to IPv6 routing, the OSPFv3 router-id still uses the 32-bit dotted decimal format as used in OSPFv2. To get more detailed information about interfaces running OSPFv3 issue the **show ipv6 ospf interface** command.

```
Rack1R1#show ipv6 ospf interface
Ethernet0/0 is up, line protocol is up
  Link Local Address FE80::2D0:58FF:FE6E:B720, Interface ID 3
  Area 0, Process ID 1, Instance ID 0, Router ID 150.1.1.1
  Network Type BROADCAST, Cost: 10
  Transmit Delay is 1 sec, State BDR, Priority 1
  Designated Router (ID) 222.22.2.1, local address
FE80::205:5EFF:FE0F:B8E0
  Backup Designated router (ID) 150.1.1.1, local address
FE80::2D0:58FF:FE6E:B720
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
<snip>
```


From the above output, we can see more detailed information about the OSPF adjacency that has formed on interface FastEthernet0/0, such as the area, process-id, local router-id, network type, interface cost, local state (DR, BDR, or DROTHER), local priority, router-id of the DR and BDR, link-local address of the DR and BDR, and interface timers.

```
Rack1R1#show ipv6 route ospf
IPv6 Routing Table - 7 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
    O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
OE2  2001:51:51:51::/64 [110/20]
     via FE80::205:5EFF:FE0F:B8E0, FastEthernet0/0
```

Note that when the routing table is examined, the next-hop address for OSPFv3 learned routes is the remote link-local address of the advertising router. Since the FastEthernet interface used for this adjacency is a broadcast media, ICMPv6 neighbor discovery will automatically resolve the remote link-local IPv6 address to the remote layer 2 (MAC) address. This can be verified by issuing the **show ipv6 neighbors** command in global configuration mode.

```
Rack1R1#show ipv6 neighbors
IPv6 Address                              Age Link-layer Addr State Interface
FE80::205:5EFF:FE0F:B8E0                   17 0005.5e0f.b8e0  STALE Et0/0
2001:192:10:1::254                         17 0005.5e0f.b8e0  STALE Et0/0
```

Had the interface used for adjacency been a multipoint non-broadcast interface, such as the main interface in Frame Relay or ATM, an explicit layer 3 to layer 2 resolution statement would have been required for the remote link-local IPv6 address.

🕮 Deep Dive

- What important improvement has been made to OSPFv3 as compared to OSPFv2?

Task 3.1 Verification

Check OSPFv3 neighbros:

```
Rack1R2#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
150.1.1.1         1   FULL/  -        00:00:31    10              Serial0/0
150.1.5.5         1   FULL/BDR        00:00:34    4               FastEthernet0/0
150.1.8.8         1   FULL/DR         00:00:35    1222            FastEthernet0/0
```

```
Rack1R1#show ipv6 route ospf
IPv6 Routing Table - 5 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
OI  2001:141:1:25::/64 [110/65]
     via FE80::2, Serial0/0.1
```

```
Rack1SW2#show ipv6 interface brief vlan 258
Vlan258                    [up/up]
    FE80::218:73FF:FE8D:9E44
    2001:141:1:25::8
```

```
Rack1R1#ping 2001:141:1:25::8

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:141:30:25::8, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 60/62/69 ms
```


Task 3.2 Solution

🕮 Before You Begin

For detailed description of OSPFv3 summarization procedure refer to VOL1 scenario **IPv6 > OSPFv3 Summarization**.

```
R2:
!
! Summarize are 1 prefixes
!
ipv6 router ospf 1
 area 1 range 2001:150:1::/60

R5:
!
! Advertise Loopback prefixes into Area 0
!
interface Loopback100
 ipv6 address 2001:150:1:5::5/64
 ipv6 ospf 1 area 1

SW2:
interface Loopback100
 ipv6 address 2001:150:1:8::8/64
 ipv6 ospf 1 area 1
```

Task 3.2 Breakdown

Similar to OSPFv2, OSPFv3 supports both internal and external summarization. Internal summarization occurs as LSAs are moving between areas, and is configured with the interface level command **area [area-id] range [summary]**, where *area-id* is the area you are summarizing from and *summary* is the summary prefix. For external summaries, the process level command **summary-prefix** is used.

Task 3.2 Verification

Verify that the summary is generated and test reachability:

```
Rack1R1#show ipv6 route ospf
<snip>
OI  2001:141:1:25::/64 [110/65]
     via FE80::2, Serial0/0.1
OI  2001:150:1::/60 [110/65]
     via FE80::2, Serial0/0.1

Rack1R1#ping 2001:150:1:8::8

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:150:1:8::8, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/59/60 ms

Rack1R1#ping 2001:150:1:5::5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:150:1:5::5, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 60/60/60 ms
```

Task 4.1 Solution

🕮 Before You Begin

For information about using RIP as PE-CE routing protocol, you may want to consult VOL1 scenario **MPLS VPN > PE-CE Routing with RIP**. Additionally, check the scenario **RIPv2 > RIPv2 Offset list** for information about RIP metric manipulation process.

R4:

```
!
! Enable RIP for VRF VPN_A
!
router rip
 version 2
 no auto-summary
 address-family ipv4 vrf VPN_A
 network 204.12.1.0
 no passive-interface FastEthernet 0/1
```

R6:

```
!
! Enable RIP for VRF VPN_A
!
router rip
 version 2
!
! Apply offset-list to the selected prefixes
! Only the ones with even 3rd octet are matched
!
 offset-list 1 in 9 Serial0/0/0
 no auto-summary
 address-family ipv4 vrf VPN_A
 network 54.0.0.0
 no passive-interface Serial 0/0/0

!
access-list 1 permit 0.0.0.0 255.255.254.255
```


Task 4.1 Verification

Check the VRF routing table for RIP routes:

```
Rack1R4#show ip route vrf VPN_A

Routing Table: VPN_A
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

C    204.12.1.0/24 is directly connected, FastEthernet0/1
     31.0.0.0/16 is subnetted, 4 subnets
R       31.3.0.0 [120/1] via 204.12.1.254, 00:00:18, FastEthernet0/1
R       31.2.0.0 [120/1] via 204.12.1.254, 00:00:18, FastEthernet0/1
R       31.1.0.0 [120/1] via 204.12.1.254, 00:00:18, FastEthernet0/1
R       31.0.0.0 [120/1] via 204.12.1.254, 00:00:18, FastEthernet0/1
     30.0.0.0/16 is subnetted, 4 subnets
R       30.2.0.0 [120/1] via 204.12.1.254, 00:00:18, FastEthernet0/1
R       30.3.0.0 [120/1] via 204.12.1.254, 00:00:18, FastEthernet0/1
R       30.0.0.0 [120/1] via 204.12.1.254, 00:00:18, FastEthernet0/1
R       30.1.0.0 [120/1] via 204.12.1.254, 00:00:18, FastEthernet0/1
```

R6 should have metric 10 for the prefixes with even 3rd octet:

```
Rack1R6#show ip route vrf VPN_A

Routing Table: VPN_A
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     54.0.0.0/24 is subnetted, 1 subnets
C       54.1.1.0 is directly connected, Serial0/0/0
R    212.18.1.0/24 [120/1] via 54.1.1.254, 00:00:17, Serial0/0/0
R    212.18.0.0/24 [120/10] via 54.1.1.254, 00:00:17, Serial0/0/0
R    212.18.3.0/24 [120/1] via 54.1.1.254, 00:00:17, Serial0/0/0
R    212.18.2.0/24 [120/10] via 54.1.1.254, 00:00:17, Serial0/0/0
```



```
Rack1R4#ping vrf VPN_A 204.12.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 204.12.1.254, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
```

🕮 Deep Dive

- Could you use RIPv2 in multihomed VPN scenrios with backdoor links?

Task 4.2 Solutions

🕮 Before You Begin

Check VOL1 scenarios **MPLS VPN > MPLS LDP** and **MPLS VPN > MP-BGP VPNv4** for information about setting up basic MPLS VPN components.

R4:

```
!
! Connect R4 and R6 using a tunnel interface
!
interface Tunnel 46
 ip address 141.1.46.4 255.255.255.0
 tunnel source Loopback0
 tunnel destination 150.1.6.6
 mpls ip

!
! Run MPLS over the Tunnel. Static route for the new Loopback required
!
ip route 150.1.66.66 255.255.255.255 Tunnel 46
!
! New Loopback for MP-BGP peering
!
interface Loopback100
 ip address 150.1.44.44 255.255.255.255

!
! Enable VPNv4 peering using the new Loopbacks
!
router bgp 400
 neighbor 150.1.66.66 remote-as 100
 neighbor 150.1.66.66 update-source Loopback100
 neighbor 150.1.66.66 ebgp-multihop

!
! Redistribute RIP routes into MP-BGP
!
address-family ipv4 vrf VPN_A
 redistribute rip
address-family ipv4
 no neighbor 150.1.66.66 activate
address-family vpnv4 unicast
 neighbor 150.1.66.66 activate
 neighbor 150.1.66.66 send-community both
```



```
router rip
 address-family ipv4 vrf VPN_A
!
! Transparent keyword allows for copying RIP metric from BGP MED
! attribute.
!
  redistribute bgp 400 metric transparent
```

R6:

```
!
! Tunnel required to run MPLS b/w R4 and R6
!
interface Tunnel 46
 ip address 141.1.46.6 255.255.255.0
 tunnel source Loopback0
 tunnel destination 150.1.4.4
 mpls ip

!
! Static route needed for new /32 loopbacks
! since the tunnel uses other loopbacks as source
!
ip route 150.1.44.44 255.255.255.255 Tunnel 46

!
! New Loopback for MP-BGP peering
!
interface Loopback100
 ip address 150.1.66.66 255.255.255.255
!
router bgp 100
 neighbor 150.1.44.44 remote-as 400
 neighbor 150.1.44.44 update-source Loopback100
 neighbor 150.1.44.44 ebgp-multihop

!
! Peer VPNv4 session only on the new loopbacks
!
address-family ipv4 vrf VPN_A
 redistribute rip
address-family ipv4
 no neighbor 150.1.44.44 activate
address-family vpnv4 unicast
 neighbor 150.1.44.44 activate
 neighbor 150.1.44.44 send-community both
!
router rip
!
! Transparent keyword allows for copying RIP metric from BGP MED
! attribute.
!
 address-family ipv4 vrf VPN_A
  redistribute bgp 100 metric transparent
```


Task 4.2 Breakdown

The task requires creating MPLs VPN using R4 and R6 as PE routers. The routers between these two are not MPLS capable, and therefore you need a workaround to run MPLS across. One traditional approach is using GRE tunnels to encapsulate MPLS packets. GRE fits the requirements as it allows multiprotocol encapsulation, unlike say IP in IP tunnels.

In order to establish an end-to-end LSP, /32 prefixes for R4 and R6 need to be known by the PEs. Since we chose to use Loopback0 interfaces as tunnel sources, new /32 prefixes are required. Therefore, we create additional Loopback100 interfaces with /32 prefixes on both R4 and R6. But this is only a part of the solution – the prefixes need to be known on R4 and R6 via IGP. We could run additional routing protocol over the tunnel, e.g. an OSPF process, and advertise the prefixes there. However, the simplest solution is using static routes at every endpoint, which is explicitly allowed by the task.

As soon as static routes are established, MPLS could be enabled on R4 and R6. By default, LDP will pick up the highest Loopback addresses (which are Loopback100 in our case) and use them as LDP router-IDs. The LDP session is then established using these addresses as sources. We don't force the use of Loopback100 interfaces as LDP router-IDs, but this could be a good idea in the exam.

When you get the LDP session up and running, check for MPLS labels for the new Loopbac100 prefixes. If you see the "pop" labels allocated across the tunnel interface, you may now proceed with configuring MP-BGP between the two devices. Create new peers in R4 and R5 using the new Loopback100 interfaces and disable IPv4 exchange on the new session. Notice that R4 and R5 reside in different Autonomous Systems, but that does not matter in this context. Enable VPNv4 address family only for this session, and make sure the session comes up.

The last step is redistributing RIP into MP-BGP and vice-versa. You need to configure this under the BGP and RIP routing processes, using the respective address families. Notice the use of the transparent keyword when redistributing into RIP. This allows for recovering RIP metric from BGP MED attribute, and effectively transporting RIP prefixes with unmodified metric.

Task 4.2 Verification

Check LDP neighbors

```
Rack1R6#show mpls ldp neighbor
    Peer LDP Ident: 150.1.44.44:0; Local LDP Ident 150.1.66.66:0
        TCP connection: 150.1.44.44.646 - 150.1.66.66.55164
        State: Oper; Msgs sent/rcvd: 38/39; Downstream
        Up time: 00:09:44
        LDP discovery sources:
          Tunnel46, Src IP addr: 141.1.46.4
        Addresses bound to peer LDP Ident:
          141.1.145.4     141.1.54.4      141.1.45.4      150.1.4.4
          150.1.44.44     141.1.46.4
```

Check BGP VPNv4 peering

```
Rack1R6#show bgp vpnv4 unicast vrf VPN_A
BGP table version is 34, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 100:1 (default for vrf VPN_A)
*> 30.0.0.0/16      150.1.44.44              1             0 400 ?
*> 30.1.0.0/16      150.1.44.44              1             0 400 ?
*> 30.2.0.0/16      150.1.44.44              1             0 400 ?
*> 30.3.0.0/16      150.1.44.44              1             0 400 ?
*> 31.0.0.0/16      150.1.44.44              1             0 400 ?
*> 31.1.0.0/16      150.1.44.44              1             0 400 ?
*> 31.2.0.0/16      150.1.44.44              1             0 400 ?
*> 31.3.0.0/16      150.1.44.44              1             0 400 ?
*> 54.1.1.0/24      0.0.0.0                  0         32768 ?
*> 204.12.1.0       150.1.44.44              0             0 400 ?
*> 212.18.0.0       54.1.1.254              10         32768 ?
*> 212.18.1.0       54.1.1.254               1         32768 ?
*> 212.18.2.0       54.1.1.254              10         32768 ?
*> 212.18.3.0       54.1.1.254               1         32768 ?
```


Check that the routes learned via VPN connection are being advertised. Notice that split-horizon is disabled by default with RIP on Frame-Relay interfaces, thus R6 re-adverise the routes learned from BB1. Pay attention to the metric values, learned from BGP MED attribute for RIP prefixes.

```
Rack1R6#debug ip rip
RIP protocol debugging is on
RIP: sending v2 update to 224.0.0.9 via Serial0/0/0 (54.1.1.6)
RIP: build update entrie
        30.0.0.0/16 via 0.0.0.0, metric 2, tag 0
        30.1.0.0/16 via 0.0.0.0, metric 2, tag 0
        30.2.0.0/16 via 0.0.0.0, metric 2, tag 0
        30.3.0.0/16 via 0.0.0.0, metric 2, tag 0
        31.0.0.0/16 via 0.0.0.0, metric 2, tag 0
        31.1.0.0/16 via 0.0.0.0, metric 2, tag 0
        31.2.0.0/16 via 0.0.0.0, metric 2, tag 0
        31.3.0.0/16 via 0.0.0.0, metric 2, tag 0
        204.12.1.0/24 via 0.0.0.0, metric 1, tag 0
        212.18.0.0/24 via 54.1.1.254, metric 11, tag 0
        212.18.1.0/24 via 54.1.1.254, metric 2, tag 0
        212.18.2.0/24 via 54.1.1.254, metric 11, tag 0
```


Check the BGP VPNv4 prefixes learned at R4:

```
Rack1R4#show bgp vpnv4 unicast vrf VPN_A
BGP table version is 38, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 100:1 (default for vrf VPN_A)
*> 30.0.0.0/16      204.12.1.254             1         32768 ?
*> 30.1.0.0/16      204.12.1.254             1         32768 ?
*> 30.2.0.0/16      204.12.1.254             1         32768 ?
*> 30.3.0.0/16      204.12.1.254             1         32768 ?
*> 31.0.0.0/16      204.12.1.254             1         32768 ?
*> 31.1.0.0/16      204.12.1.254             1         32768 ?
*> 31.2.0.0/16      204.12.1.254             1         32768 ?
*> 31.3.0.0/16      204.12.1.254             1         32768 ?
*> 54.1.1.0/24      150.1.66.66              0             0 100 ?
*> 204.12.1.0       0.0.0.0                  0         32768 ?
*> 212.18.0.0       150.1.66.66             10             0 100 ?
*> 212.18.1.0       150.1.66.66              1             0 100 ?
*> 212.18.2.0       150.1.66.66             10             0 100 ?
*> 212.18.3.0       150.1.66.66              1             0 100 ?

Rack1R4#debug ip rip
RIP protocol debugging is on
RIP: sending v2 update to 224.0.0.9 via FastEthernet0/1 (204.12.1.4)
RIP: build update entries
        54.1.1.0/24 via 0.0.0.0, metric 1, tag 0
        212.18.0.0/24 via 0.0.0.0, metric 11, tag 0
        212.18.1.0/24 via 0.0.0.0, metric 2, tag 0
        212.18.2.0/24 via 0.0.0.0, metric 11, tag 0
        212.18.3.0/24 via 0.0.0.0, metric 2, tag 0
```

Notice that in both cases metrics were carried as BGP MEDs and preserved across the VPN connection.

🕮 Deep Dive

- What other tunneling technology you can use to transport MPLS over IP cloud?
- Can you completely replace MPLS forwarding with IP-based tunnels?
- What could be the problem of carrying RIP metric in MED attribute?

Task 4.3 Solutions

🕮 Before You Begin

Refer to the following VOL1 labs for more information on the features used in this solution: **BGP > BGP Local-AS** and **MPLS VPN > PE-CE Routing with BGP**.

R4:

```
router bgp 400
 address-family ipv4 vrf VPN_A
  neighbor 204.12.1.254 remote-as 54
!
! Needed to override the last AS with local AS
!
  neighbor 204.12.1.254 as-override
!
! No-prepend is needed to make R6 learn prefixes from AS 54
!
  neighbor 204.12.1.254 local-as 100 no-prepend
```

R6:

```
router bgp 100
 address-family ipv4 vrf VPN_A
  neighbor 54.1.1.254 remote-as 54
  neighbor 54.1.1.254 as-override
```

Task 4.3 Breakdown

Thie scenario makes use for two features: BGP local-AS and BGP AS-Override. The first feature allows for R4, the PE router, to look like it belongs to AS 100, by advertising this AS in the BGP OPEN message and placing it as the last AS in the AS_PATH. Effectlviely, from BB1/BB3 perspective R4 and R6 will look like they belong to AS 100.

However, the other problem is that we want “virtual” AS 100 to provide transit services to AS 54. When BB3 prefixes are received by BB1 the latter will reject them as it will notice AS 54 in the path. The same hold true for BB3 learning BB1 prefixes. By using the “AS-Override” feature, we resolve this problem, instructing R4 and R6 to replace the “previosly last” AS# in the AS_PATH with their own AS# configured for given BGP session.

Effectively, R4 will replace the AS# 54 found in the end of AS_PATH with AS# 100 (the local-AS) and R6 will do the same, using the AS 100 configured for peering with BB1. BB1 and BB3 will see the prefixes as coming from AS 100, not AS 54. An alternative would be configuring BB1 and BB3 to allow accepting its own AS, but this is impossible as we can’t modify the backbone router configurations.

Task 4.3 Verification

Check the BGP prefixes learned by the backbone routers. Notice that in the real exam you may have not the ability to access the backbone routers.

```
BB1>show ip bgp regexp 100
BGP table version is 411, local router ID is 212.18.3.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 30.0.0.0/16      54.1.1.6                             0 100 400 ?
*> 30.1.0.0/16      54.1.1.6                             0 100 400 ?
*> 30.2.0.0/16      54.1.1.6                             0 100 400 ?
*> 30.3.0.0/16      54.1.1.6                             0 100 400 ?
*> 31.0.0.0/16      54.1.1.6                             0 100 400 ?
*> 31.1.0.0/16      54.1.1.6                             0 100 400 ?
*> 31.2.0.0/16      54.1.1.6                             0 100 400 ?
*> 31.3.0.0/16      54.1.1.6                             0 100 400 ?
r> 54.1.1.0/24      54.1.1.6                  0          0 100 ?
*> 204.12.1.0       54.1.1.6                             0 100 400 ?
* i                 172.16.4.3                0    100     0 100 400 ?
r> 212.18.0.0       54.1.1.6                 10          0 100 ?
r> 212.18.1.0       54.1.1.6                  1          0 100 ?
r> 212.18.2.0       54.1.1.6                 10          0 100 ?
r> 212.18.3.0       54.1.1.6                  1          0 100 ?
```



```
BB3>show ip bgp regexp 100
BGP table version is 361, local router ID is 31.3.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
r>i30.0.0.0/16      172.16.4.1               0    100      0 100 400 ?
r                   204.12.1.4               1             0 100 400 ?
r>i30.1.0.0/16      172.16.4.1               0    100      0 100 400 ?
r                   204.12.1.4               1             0 100 400 ?
r>i30.2.0.0/16      172.16.4.1               0    100      0 100 400 ?
r                   204.12.1.4               1             0 100 400 ?
r>i30.3.0.0/16      172.16.4.1               0    100      0 100 400 ?
r                   204.12.1.4               1             0 100 400 ?
r>i31.0.0.0/16      172.16.4.1               0    100      0 100 400 ?
r                   204.12.1.4               1             0 100 400 ?
r>i31.1.0.0/16      172.16.4.1               0    100      0 100 400 ?
r                   204.12.1.4               1             0 100 400 ?
r>i31.2.0.0/16      172.16.4.1               0    100      0 100 400 ?
r                   204.12.1.4               1             0 100 400 ?
r>i31.3.0.0/16      172.16.4.1               0    100      0 100 400 ?
r                   204.12.1.4               1             0 100 400 ?
r>i54.1.1.0/24      172.16.4.1               0    100      0 100 ?
r                   204.12.1.4                             0 100 400
100 ?
r i204.12.1.0       172.16.4.1               0    100      0 100 400 ?
r>                  204.12.1.4               0             0 100 400 ?
r>i212.18.0.0       172.16.4.1              10    100      0 100 ?
r                   204.12.1.4                             0 100 400
100 ?
r>i212.18.1.0       172.16.4.1               1    100      0 100 ?
r                   204.12.1.4                             0 100 400
100 ?
r>i212.18.2.0       172.16.4.1              10    100      0 100 ?
r                   204.12.1.4                             0 100 400
100 ?
r>i212.18.3.0       172.16.4.1               1    100      0 100 ?
r                   204.12.1.4                             0 100 400
100 ?
```


Trace end-to-end connecitvity across the VPN:

```
Rack1R4#traceroute vrf VPN_A 212.18.1.1 source 204.12.1.4

Type escape sequence to abort.
Tracing the route to 212.18.1.1

  1 54.1.1.6 [MPLS: Label 39 Exp 0] 72 msec 76 msec 72 msec
  2 54.1.1.254 64 msec *  60 msec
```

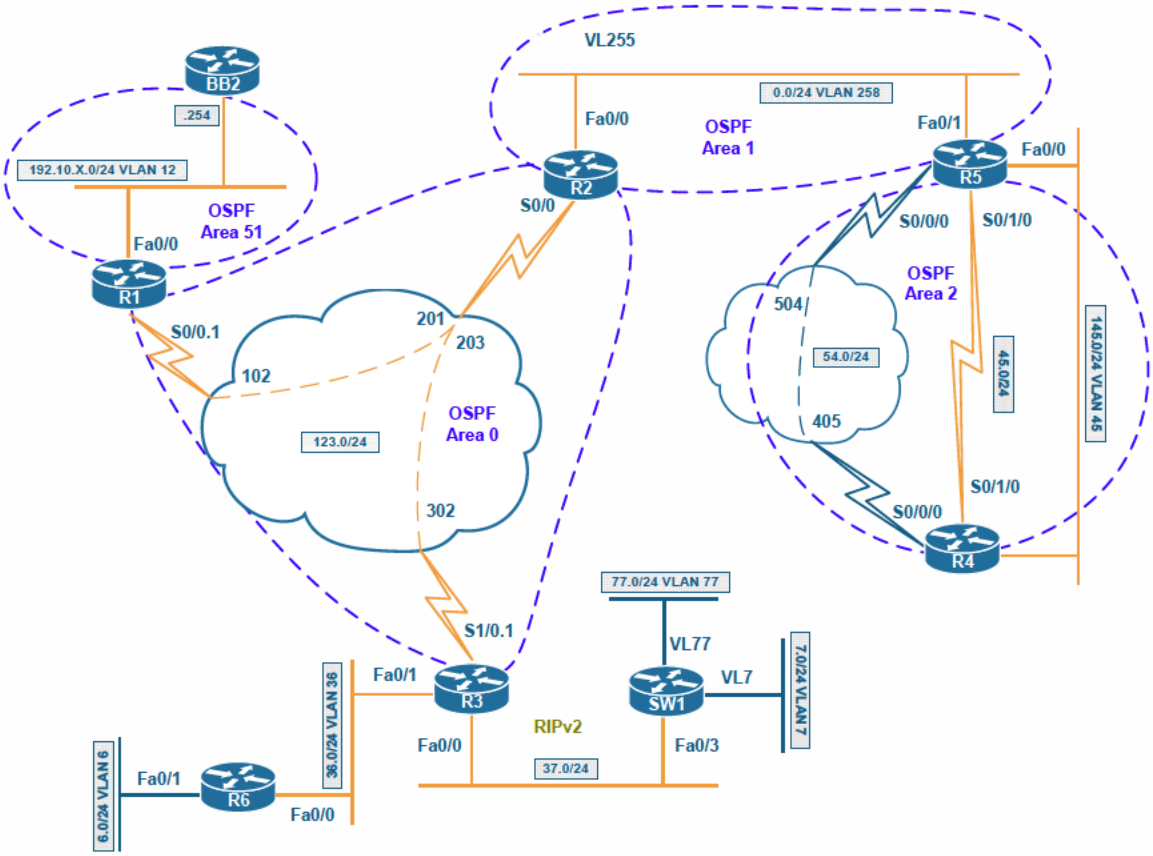
🕮 Deep Dive

- Why as-override option could be dangerous?
- What solution you could use the prevent routing loops with the as-override feature?

Task 5.1 Solutions

🕮 Before You Begin

You may find more information about Auto-RP MA and RP configuration in the following VOL1 scenarios: **`Multicast > AutoRP`** and **`Multicast > AutoRP - Multiple Candidate RPs`** and **`Multicast > AutoRP - Filtering Candidate RPs.`** The multicast topology for this scenario is presented below (you are supposed to discover and document it on your own):



R1, R2, R3, R4, R5, SW1, and SW2:

```
!
! You dont need to enable the listener in MA/RP
! but it wont affect the solution if you do
!
ip pim autorp listener
```

R2, R5, and SW2:

```
interface Loopback0
 ip pim sparse-mode
```

R2:

```
!
! R2 is the RP candidate, announcing a group range
!
ip pim send-rp-announce Loopback0 scope 16 group-list 50
!
access-list 50 permit 225.0.0.0 0.255.255.255
```

R5:

```
!
! R5 is the RP candidate, announcing another group range
!
ip pim send-rp-announce Loopback0 scope 16 group-list 50
!
access-list 50 permit 239.0.0.0 0.255.255.255
```

SW2:

```
!
! SW2 is the MA, announcing information learned from R2/R5:
!
ip pim send-rp-discovery Loopback0 scope 16
```


Task 5.1 Breakdown

Since PIM sparse mode was configured in the network and this section asks for Auto-RP, the **`ip pim autorp listener`** command will need to be used on all multicast devices to enable the 224.0.1.39 and 224.0.1.40 groups to be distributed in dense mode.

🕮 Deep Dive

- Why don't you need PIM RP Listener in the RP and MA?
- How does Auto-RP interact with static RP configuration?
- Can you flood Auto-RP groups without ever using PIM dense mode?

Task 5.1 Verification

Verify the group to RP mappings on each multicast enabled router. Look for the RPs and the Mapping Agent addresses.

```
Rack1R1#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 225.0.0.0/8
  RP 150.1.2.2 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:21:16, expires: 00:02:50
Group(s) 239.0.0.0/8
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:21:02, expires: 00:02:51

Rack1R2#show ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP (Auto-RP)

Group(s) 225.0.0.0/8
  RP 150.1.2.2 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:21:22, expires: 00:02:43
Group(s) 239.0.0.0/8
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:21:08, expires: 00:02:45

Rack1R3#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 225.0.0.0/8
  RP 150.1.2.2 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:21:28, expires: 00:02:35
Group(s) 239.0.0.0/8
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:21:14, expires: 00:02:39
```



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

Group(s) 225.0.0.0/8
  RP 150.1.2.2 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:21:34, expires: 00:02:30
Group(s) 239.0.0.0/8
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:21:20, expires: 00:02:31

Rack1R5#show ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP (Auto-RP)

Group(s) 225.0.0.0/8
  RP 150.1.2.2 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:21:43, expires: 00:02:23
Group(s) 239.0.0.0/8
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:21:28, expires: 00:02:22

Rack1SW1#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 225.0.0.0/8
  RP 150.1.2.2 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:22:03, expires: 00:02:05
Group(s) 239.0.0.0/8
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:21:49, expires: 00:02:03

Rack1SW2#show ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP-mapping agent (Loopback0)

Group(s) 225.0.0.0/8
  RP 150.1.2.2 (?), v2v1
    Info source: 150.1.2.2 (?), elected via Auto-RP
         Uptime: 00:22:09, expires: 00:02:52
Group(s) 239.0.0.0/8
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.5.5 (?), elected via Auto-RP
         Uptime: 00:21:54, expires: 00:02:01
```


Task 5.2 Solutions

🕮 Before You Begin

For more information about the fratures used in this solution refer to VOL1 scenario **Multicast > PIM NBMA-Mode**

R2:

```
!
! Enable NBMA mode to allow sparse traffic from R1 to R3
!
interface Serial0/0
 ip pim nbma-mode
!
! There is a tunnel b/w R2 and R5 and it has to be multicast enabled
! to allow for successful RPF checks
!
interface Tunnel0
 ip pim sparse-mode
```

R3:

```
!
! Join the two groups corresponding to different RPs
!
interface FastEthernet0/0
 ip igmp join-group 225.25.25.25
!
interface FastEthernet0/1
 ip igmp join-group 239.39.39.39
```

R5:

```
!
! There is a tunnel b/w R2 and R5 and it has to be multicast enabled
! to allow for successful RPF checks
!
interface Tunnel0
 ip pim sparse-mode
```


Task 5.2 Breakdown

The problem in this task is that R2 will not forward multicast packets received on its Frame-Relay interface out of the same interface. This will cause the multicast pings from R1 to not reach R3. To overcome this issue, PIM NBMA mode can be configured.

Before the NBMA mode is configured:

```
Rack1R1#ping 225.25.25.25

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 225.25.25.25, timeout is 2 seconds:
.
Rack1R1#ping 239.39.39.39

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 239.39.39.39, timeout is 2 seconds:
.
Rack1R1#
```

After the NBMA mode is configured:

```
Rack1R1#ping 225.25.25.25

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 225.25.25.25, timeout is 2 seconds:

Reply to request 0 from 141.1.13.3, 36 ms

Rack1R1#ping 239.39.39.39

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 239.39.39.39, timeout is 2 seconds:

Reply to request 0 from 141.1.13.3, 36 ms
Rack1R1#
```

The second problem is that R5/R2 may reject multicast traffic from each other, since they both use the tunnel interace to route inter-area traffic. This is because, multicast (PIM) is only enabled on the physical link between R2 and R5. In order to fix this problem enable PIM and multicast forwarding over the tunnel interface.

Task 5.2 Verification

Simulate multicast packets from R1 to group 225.25.25.25:

```
Rack1R1#ping
Protocol [ip]:
Target IP address: 225.25.25.25
Repeat count [1]: 100
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Interface [All]: Serial0/0.1
Time to live [255]:
Source address: 192.10.1.1
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 225.25.25.25, timeout is 2 seconds:
Packet sent with a source address of 192.10.1.1

Reply to request 0 from 141.1.123.3, 92 ms
Reply to request 1 from 141.1.123.3, 92 ms
Reply to request 2 from 141.1.123.3, 92 ms
```

Verify the multicast routing table on R2:

```
Rack1R2#show ip mroute
IP Multicast Routing Table
<snip>

(*, 225.25.25.25), 00:10:49/stopped, RP 150.1.2.2, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0, 141.1.123.3, Forward/Sparse, 00:10:49/00:02:35

<output omitted>

(192.10.1.1, 225.25.25.25), 00:01:43/00:03:22, flags: T
  Incoming interface: Serial0/0, RPF nbr 141.1.123.1
  Outgoing interface list:
    Serial0/0, 141.1.123.3, Forward/Sparse, 00:01:43/00:02:46
```


Finally, simulate multicast packets from R4 to group 239.39.39.39:

```
Rack1R4#ping
Protocol [ip]:
Target IP address: 239.39.39.39
Repeat count [1]: 100
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Interface [All]: FastEthernet0/0
Time to live [255]:
Source address: 204.12.1.4
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 239.39.39.39, timeout is 2 seconds:
Packet sent with a source address of 204.12.1.4

Reply to request 0 from 141.1.123.3, 40 ms
Reply to request 1 from 141.1.123.3, 36 ms
Reply to request 2 from 141.1.123.3, 32 ms
```

Verify the multicast table on R2 again (note Tunnel0 is used as RPF interface):

```
Rack1R2#show ip mroute 239.39.39.39
<snip>

(*, 239.39.39.39), 01:52:21/stopped, RP 150.1.5.5, flags: S
  Incoming interface: Tunnel0, RPF nbr 141.1.25.5
  Outgoing interface list:
    Serial0/0, 141.1.123.3, Forward/Sparse, 01:52:21/00:03:00

(204.12.1.4, 239.39.39.39), 00:00:06/00:03:23, flags: T
  Incoming interface: Tunnel0, RPF nbr 141.1.25.5
  Outgoing interface list:
    Serial0/0, 141.1.123.3, Forward/Sparse, 00:00:06/00:03:23
```

🕮 Deep Dive

- Could PIM NBMA be usefule on broadcast interfaces, e.g. Ethernet?

Task 5.3 Solutions

🕮 Before You Begin

For comprehensive reading on multicast rate-limiting refer to the scenario **`Multicast > Multicast Rate-Limiting`**. You may also want to read the following technical document at Cisco's engineering FTP site:

ftp://ftpeng.cisco.com/ipmulticast/config-notes/multicast-rate-limit.txt

SW1:
```
interface FastEthernet0/3
 ip multicast rate-limit out 1000
```

Task 5.3 Breakdown

Although standard QoS methods could be used to meet the requirements for this section, the **`ip multicast rate-limit`** command is the simplest solution. Other options for this command can be configured to limit the amount of multicast traffic inbound, from a specific source address, and/or multicast traffic sent to a specific multicast group. If a value of 0 is used (the default), all multicast traffic will be dropped. Notice that the command without any additional parameters, specifying group or source list applies to aggregate multicast traffic going in/out of the interface, while specifying **`group-list`** or **`source-list`** will apply the command per multicast-flow matched by the list.

🕮 Deep Dive

- How does multicast-rate limit define traffic policing window?

Task 6.1 Solutions

🕮 Before You Begin

For introductory examples and breakdowns on Zone-Based Firewall configuration you may refer to VOL1 scenario **`Security > Zone Based Firewall`**.

R6:

```
!
! The host opened for HTTP/HTTPs access
!
ip access-list extended ACL_OUTSIDE_TO_INSIDE_HTTP
 permit tcp any host 204.12.1.100

!
! Combine two protocol in "OR" pair
!
class-map type inspect match-any CMAP_HTTP_HTTPS
 match protocol http
 match protocol https
!
! Class map that matches HTTP/HTTPs traffic to the server
!
class-map type inspect CMAP_OUTSIDE_TO_INSIDE_HTTP
 match access-group name ACL_OUTSIDE_TO_INSIDE_HTTP
 match class-map CMAP_HTTP_HTTPS
!
! Other protocols allowed across the firewall (DNS/ICMP)
!
class-map type inspect match-any CMAP_OTHER_PROTOCOLS
 match protocol dns
 match protocol icmp

policy-map type inspect PMAP_OUTSIDE_TO_INSIDE
 class CMAP_OUTSIDE_TO_INSIDE_HTTP
  inspect
!
! Rate-Limit DNS and ICMP
!
 class CMAP_OTHER_PROTOCOLS
  inspect
  police rate 128000 burst 8000
!
class-map type inspect match-any CMAP_INSIDE_TO_OUTSIDE
 match protocol udp
 match protocol tcp
 match protocol icmp
!
policy-map type inspect PMAP_INSIDE_TO_OUTSIDE
 class CMAP_INSIDE_TO_OUTSIDE
  inspect
```



```
!
! Create security zones
!
zone security INSIDE
zone security OUTSIDE
!
zone-pair security ZP_INSIDE_TO_OUTSIDE source INSIDE destination
OUTSIDE
service-policy type inspect PMAP_INSIDE_TO_OUTSIDE
!
zone-pair security ZP_OUTSIDE_TO_INSIDE source OUTSIDE destination
INSIDE
service-policy type inspect PMAP_OUTSIDE_TO_INSIDE

!
! Create a transit VRF to hairpin traffic
!
ip vrf FW_HAIRPIN
 rd 100:100
 route-target export 100:100
 route-target import 100:100
!
! Create tunnels linking VRFs VPN_A and FW_HAIRPIN
!
interface Loopback201
 ip address 6.6.6.6 255.255.255.255
!
interface Loopback202
 ip address 66.66.66.66 255.255.255.255
!
interface Tunnel201
 ip vrf forwarding FW_HAIRPIN
 ip address 200.200.200.1 255.255.255.0
 tunnel source Loopback201
 tunnel destination 66.66.66.66
!
interface Tunnel202
 ip vrf forwarding VPN_A
 ip address 200.200.200.2 255.255.255.0
 tunnel source Loopback202
 tunnel destination 6.6.6.6
```



```
!
! Enable RIP route exchange over the tunnels
!
router rip
 no passive-interface Tunnel201
 no passive-interface Tunnel202
 !
 address-family ipv4 vrf VPN_A
  no redistribute bgp 100 metric transparent
  network 200.200.200.0
 !
 address-family ipv4 vrf FW_HAIRPIN
  redistribute bgp 100 metric transparent
  network 200.200.200.0
!
! Enable BGP peering between the VRFs
!
router bgp 100
 address-family ipv4 vrf VPN_A
 !
 ! VPN_A no longer redistributes RIP into BGP
 ! it learns all routes from VRF FW_HAIRPIN
 !
  no redistribute rip
  neighbor 200.200.200.1 remote-as 100
  !
  ! Next-hop self needed as peering is iBGP
  ! and AS 400 is external
  !
  neighbor 200.200.200.1 next-hop-self
  !
  ! Different BGP router-id per VRF are needed
  !
  bgp router-id 200.200.200.2
  address-family ipv4 vrf FW_HAIRPIN
  !
  ! VRF FW_HAIRPIN now propagates all routes to AS 400
  !
  redistribute rip
  neighbor 200.200.200.2 remote-as 100
  neighbor 200.200.200.2 next-hop-self
  bgp router-id 200.200.200.1

!
! Assign the security zones
!
interface tunnel 202
 zone-member security INSIDE
!
interface Serial 0/0/0
 zone-member security OUTSIDE
```



```
R4:
!
! Make R4 import all routes from FW_HAIRPIN
!
ip vrf VPN_A
 no route-target import 100:1
 no route-target export 100:1
 route-target both 100:100
```

Task 6.1 Breakdown

In this scenario, we need to make zone-based firewall work with MPLS encapsulated traffic. This is not possible natively, but could be done by configuring a special hairpin solution, where incoming MPLS labeled traffic is decapsulated and looped back to the router itself using explicit tunnel interface. The filtering policy then applies to the Frame-Relay and Tunnel interfaces.

In order to accomplish hair-pinning a new VRF is being created, named FW_HAIRPIN. Two tunnels are created in R6 to link VRF VPN_A and VRF FW_HAIRPIN using explicit tunnel interfaces: Tunnel 201 and Tunnel 202. One of the interfaces belongs to VRF VPN_A and another belong to VRF FW_HAIRPIN and both tunnels essentially link the VRFs back-to-back on the same router.

After this, RIP and BGP are configured so that two VRFs may dynamically exchange routes over the tunnel interfaces. Notice that we need to tune BGP settings and change BGP router-IDs per VRF, as otherwise BGP peering sessions will not come up. Effectively, two VRFs send RIP updates and exchange BGP routes over the new hairpin tunnel interface.

After the routing exchange has been completed, we configure R4 so that it local instance of VRF VPN_A import routes from VRF FW_HAIRPIN. Effectively, all traffic going from R4 to BB1 will have to traverse the hairpin tunnel interface connecting the VRFs in R6. After this, R6 will see IP traffic entering VRF VPN_A unlabeled over a Tunnel interface (tunnel 202) and thus ZFW policies could be applied to transit traffic.

Task 6.1 Verification

Check the firewall policy:

```
Rack1R6#show policy-map type inspect zone-pair

policy exists on zp ZP_INSIDE_TO_OUTSIDE
 Zone-pair: ZP_INSIDE_TO_OUTSIDE

  Service-policy inspect : PMAP_INSIDE_TO_OUTSIDE

    Class-map: CMAP_INSIDE_TO_OUTSIDE (match-any)
      Match: protocol udp
        0 packets, 0 bytes
        30 second rate 0 bps
      Match: protocol tcp
        0 packets, 0 bytes
        30 second rate 0 bps
      Match: protocol icmp
        0 packets, 0 bytes
        30 second rate 0 bps

   Inspect
        Session creations since subsystem startup or last reset 0
        Current session counts (estab/half-open/terminating) [0:0:0]
        Maxever session counts (estab/half-open/terminating) [0:0:0]
        Last session created never
        Last statistic reset never
        Last session creation rate 0
        Maxever session creation rate 0
        Last half-open session total 0

    Class-map: class-default (match-any)
      Match: any
      Drop
        0 packets, 0 bytes

policy exists on zp ZP_OUTSIDE_TO_INSIDE
 Zone-pair: ZP_OUTSIDE_TO_INSIDE

  Service-policy inspect : PMAP_OUTSIDE_TO_INSIDE

    Class-map: CMAP_OUTSIDE_TO_INSIDE_HTTP (match-all)
      Match: access-group name ACL_OUTSIDE_TO_INSIDE_HTTP
      Match: class-map match-any CMAP_HTTP_HTTPS
        Match: protocol http
          0 packets, 0 bytes
          30 second rate 0 bps
        Match: protocol https
          0 packets, 0 bytes
          30 second rate 0 bps
```



```
   Inspect
        Session creations since subsystem startup or last reset 0
        Current session counts (estab/half-open/terminating) [0:0:0]
        Maxever session counts (estab/half-open/terminating) [0:0:0]
        Last session created never
        Last statistic reset never
        Last session creation rate 0
        Maxever session creation rate 0
        Last half-open session total 0

    Class-map: CMAP_OTHER_PROTOCOLS (match-any)
      Match: protocol dns
        0 packets, 0 bytes
        30 second rate 0 bps
      Match: protocol icmp
        0 packets, 0 bytes
        30 second rate 0 bps

   Inspect
        Session creations since subsystem startup or last reset 0
        Current session counts (estab/half-open/terminating) [0:0:0]
        Maxever session counts (estab/half-open/terminating) [0:0:0]
        Last session created never
        Last statistic reset never
        Last session creation rate 0
        Maxever session creation rate 0
        Last half-open session total 0
       Police
        rate 128000 bps,8000 limit
        conformed 0 packets, 0 bytes; actions: transmit
        exceeded 0 packets, 0 bytes; actions: drop
        conformed 0 bps, exceed 0 bps

    Class-map: class-default (match-any)
      Match: any
      Drop
        0 packets, 0 bytes

Rack1R6#show zone security
zone self
  Description: System defined zone


zone INSIDE
  Member Interfaces:
    Tunnel202
zone OUTSIDE
  Member Interfaces:
    Serial0/0/0
```


Check routing for VRF VPN_A

```
Rack1R6#show ip route vrf VPN_A

Routing Table: VPN_A
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

B    119.0.0.0/8 [20/0] via 54.1.1.254, 02:14:40
C    200.200.200.0/24 is directly connected, Tunnel202
B    118.0.0.0/8 [20/0] via 54.1.1.254, 02:14:40
B    117.0.0.0/8 [20/0] via 54.1.1.254, 02:14:40
B    116.0.0.0/8 [20/0] via 54.1.1.254, 02:14:40
B    115.0.0.0/8 [20/0] via 54.1.1.254, 02:14:40
B    114.0.0.0/8 [20/0] via 54.1.1.254, 02:14:40
     54.0.0.0/24 is subnetted, 1 subnets
C       54.1.1.0 is directly connected, Serial0/0/0
B    113.0.0.0/8 [20/0] via 54.1.1.254, 02:14:40
B    112.0.0.0/8 [20/0] via 54.1.1.254, 02:14:40
R    212.18.1.0/24 [120/1] via 54.1.1.254, 00:00:25, Serial0/0/0
R    212.18.0.0/24 [120/10] via 54.1.1.254, 00:00:25, Serial0/0/0
R    212.18.3.0/24 [120/1] via 54.1.1.254, 00:00:25, Serial0/0/0
R    212.18.2.0/24 [120/10] via 54.1.1.254, 00:00:25, Serial0/0/0
     28.0.0.0/24 is subnetted, 2 subnets
B       28.119.17.0 [20/0] via 54.1.1.254, 02:15:03
B       28.119.16.0 [20/0] via 54.1.1.254, 02:15:03
     31.0.0.0/16 is subnetted, 4 subnets
R       31.3.0.0 [120/2] via 200.200.200.1, 00:00:28, Tunnel202
R       31.2.0.0 [120/2] via 200.200.200.1, 00:00:28, Tunnel202
R       31.1.0.0 [120/2] via 200.200.200.1, 00:00:28, Tunnel202
R       31.0.0.0 [120/2] via 200.200.200.1, 00:00:28, Tunnel202
R    204.12.1.0/24 [120/1] via 200.200.200.1, 00:00:28, Tunnel202
     30.0.0.0/16 is subnetted, 4 subnets
R       30.2.0.0 [120/2] via 200.200.200.1, 00:00:28, Tunnel202
R       30.3.0.0 [120/2] via 200.200.200.1, 00:00:28, Tunnel202
R       30.0.0.0 [120/2] via 200.200.200.1, 00:00:28, Tunnel202
R       30.1.0.0 [120/2] via 200.200.200.1, 00:00:28, Tunnel202
```


Now, generate some traffic from R4 to BB1:

```
Rack1R4#sh ip route vrf VPN_A

Routing Table: VPN_A
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

B    119.0.0.0/8 [20/0] via 204.12.1.254, 00:11:27
B    200.200.200.0/24 [20/0] via 150.1.66.66, 02:12:48
B    118.0.0.0/8 [20/0] via 204.12.1.254, 00:11:27
B    117.0.0.0/8 [20/0] via 204.12.1.254, 00:11:27
B    116.0.0.0/8 [20/0] via 204.12.1.254, 00:11:27
B    115.0.0.0/8 [20/0] via 204.12.1.254, 00:11:27
B    114.0.0.0/8 [20/0] via 204.12.1.254, 00:11:27
     54.0.0.0/24 is subnetted, 1 subnets
B       54.1.1.0 [20/1] via 150.1.66.66, 02:12:48
B    113.0.0.0/8 [20/0] via 204.12.1.254, 00:11:27
B    112.0.0.0/8 [20/0] via 204.12.1.254, 00:11:27
B    212.18.1.0/24 [20/2] via 150.1.66.66, 02:12:50
B    212.18.0.0/24 [20/11] via 150.1.66.66, 02:12:50
B    212.18.3.0/24 [20/2] via 150.1.66.66, 02:12:50
B    212.18.2.0/24 [20/11] via 150.1.66.66, 02:12:50
     28.0.0.0/24 is subnetted, 2 subnets
B       28.119.17.0 [20/0] via 204.12.1.254, 00:11:29
B       28.119.16.0 [20/0] via 204.12.1.254, 00:11:29
     31.0.0.0/16 is subnetted, 4 subnets
R       31.3.0.0 [120/1] via 204.12.1.254, 00:00:12, FastEthernet0/1
R       31.2.0.0 [120/1] via 204.12.1.254, 00:00:12, FastEthernet0/1
R       31.1.0.0 [120/1] via 204.12.1.254, 00:00:12, FastEthernet0/1
R       31.0.0.0 [120/1] via 204.12.1.254, 00:00:12, FastEthernet0/1
C    204.12.1.0/24 is directly connected, FastEthernet0/1
     30.0.0.0/16 is subnetted, 4 subnets
R       30.2.0.0 [120/1] via 204.12.1.254, 00:00:12, FastEthernet0/1
R       30.3.0.0 [120/1] via 204.12.1.254, 00:00:12, FastEthernet0/1
R       30.0.0.0 [120/1] via 204.12.1.254, 00:00:12, FastEthernet0/1
R       30.1.0.0 [120/1] via 204.12.1.254, 00:00:12, FastEthernet0/1
```



```
Rack1R4#ping vrf VPN_A 212.18.1.1 source fastEthernet 0/1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 212.18.1.1, timeout is 2 seconds:
Packet sent with a source address of 204.12.1.4
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/70/72 ms

Rack1R6#show policy-map type inspect zone-pair

policy exists on zp ZP_INSIDE_TO_OUTSIDE
 Zone-pair: ZP_INSIDE_TO_OUTSIDE

  Service-policy inspect : PMAP_INSIDE_TO_OUTSIDE

    Class-map: CMAP_INSIDE_TO_OUTSIDE (match-any)
      Match: protocol udp
        0 packets, 0 bytes
        30 second rate 0 bps
      Match: protocol tcp
        0 packets, 0 bytes
        30 second rate 0 bps
      Match: protocol icmp
        3 packets, 240 bytes
        30 second rate 0 bps
```

🕮 Deep Dive

- Can you filter MPLS traffic using IP access-lists?
- Where should you perform all security and QoS decisions in the MPLS network?

Task 6.2 Solutions

🕮 Before You Begin

Refer to VOL1 scenario **Security > Preventing Packet Spoofing** with uRPF for more information and examples of using uRPF.

R4:

```
ip cef
!
interface FastEthernet0/1
 ip verify unicast reverse-path
```

OR

R4:

```
ip cef
!
interface FastEthernet0/1
 ip verify unicast source reachable-via any
```

Task 6.2 Breakdown

Unicast reverse path forwarding was developed to help solve issues concerning DoS attacks. With DoS attacks, the source IP address of the packet is usually spoofed. Basically, unicast reverse path forwarding checks the source IP address of all packets received inbound on an interface to see if that particular interface is the interface the router would use to route a packet to that source IP address. Theoretically, this is a good idea and in some situations can be helpful, but in a real network this can cause problems with asymmetrical routing. Let's look at an example of strict mode:

A packet with a source IP address of 141.X.37.100 is received inbound on R4 interface Fa0/1. With unicast RPF enabled, R4 would assume the source IP address must be spoofed as R4 will normally route via R5 to reach that particular subnet. This will cause R4 to drop this packet. Although this might be the action we want, what if R3 is routing through BB1 & BB3 (AS 54) to reach the networks behind R4, while R4 is routing through R5 & R2 to reach R3. In this situation, unicast RPF would cause packets to be incorrectly dropped.

Another example would be for a remote site with one connection to the rest of the network. Normally, to prevent spoofed IP addresses from being sent by the remote site, an access-list is created to only permit packets with the remote site's network as the source IP address. This access-list is applied inbound on the main site's router. Whenever a new network is added to the remote site, an additional route is added in the hub router and the access-list will need to be updated. For an enterprise network that does not make a lot of changes, this might not mean a lot of additional work, but for a large ISP this could possibly mean a lot of additional work. A simpler solution would be to use unicast RPF as the access-list is not needed.

For reasons we can see from the examples, unicast RPF is normally implemented on the edge of the network where symmetrical routing is more likely to occur. If unicast RPF is used in the core of a network, ensure that asymmetric routing does not occur.

In addition to strict mode, there is also the possibility of loose mode. In strict mode, the route to the source address for the received packet must be via the same interface. For loose mode, there just needs to be a matching route in the routing table, and it can be via any interface.

The 28.119 networks learned via BGP are learned from BB1, but the networks are originated on BB3. A ping to these networks will pass out via BB1, and will return to your topology via R4, since BB3 is learning routes via R4. Because of the asymmetric routing, strict mode would cause these pings to fail, whereas loose mode will allow the return traffic to enter R4's interface.

Note

Unicast RPF requires CEF to be enabled globally.

Task 6.2 Verification

Verify that uRPF is configured on the interface:

```
Rack1R4#show ip interface Fa0/1 | include verif
  IP verify source reachable-via ANY
  0 verification drops
  0 suppressed verification drops
```

Deep Dive

- What is uRPF feasible mode?

Task 6.3 Solutions

🕮 Before You Begin

For explanation of the Control Plane Policing feature refer to VOL1 scenario **Security > Control Plane Policing**.

```
R6:
!
! Select traffic not to police
!
ip access-list extended ELIGIBLE_TRAFFIC
 permit   tcp any any eq bgp
 permit   tcp any eq bgp any
 permit   tcp any any eq telnet
 permit   tcp any any eq 22
!
! Classify the control-plane traffic
!
class-map match-all ELIGIBLE_TRAFFIC
 match access-group name ELIGIBLE_TRAFFIC
!
policy-map CONTROL_PLAN_POLICING
 class ELIGIBLE_TRAFFIC
 class class-default
  police rate 1000 pps
!
! Apply control-plane policing
!
control-plane
 service-policy input CONTROL_PLAN_POLICING
```


Task 6.3 Breakdown

The only security feature that allow packet-per-second policing is the control-plane policing. In recent IOS versions it has evolved into fully-flown control-plane protection feature, which allows for addition functionality besides simply policing the traffic.

In this scenario, you need to properly select the traffic that should not be subject to policing. This includes session going toward the router CPU, that is detined to the BGP, SSH and Telnet ports. For BGP you need to account for bi-directional sessions that BGP may establish and match both incoming and returning BGP session traffic.

In real-word scenario you may probably want to be very specific about the source/destination IP addresses used for control plane traffic, to prevent spoofed traffic from taking down your router's CPU.

Task 6.3 Verification

Check the control-plane service-policy

```
Rack1R6#show policy-map control-plane input
Control Plane

  Service-policy input: CONTROL_PLAN_POLICING

    Class-map: ELIGIBLE_TRAFFIC (match-all)
      3 packets, 574 bytes
      5 minute offered rate 0 bps
      Match: access-group name ELIGIBLE_TRAFFIC

    Class-map: class-default (match-any)
      22 packets, 1397 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      police:
          rate 1000 pps, burst 244 packets
        conformed 22 packets; actions:

          transmit
        exceeded 0 packets; actions:
          drop
        conformed 1 pps, exceed 0 pps
```

🕮 Deep Dive

- Why control plane security is so important?
- What other control plane security mechanisms you know?

Task 7.1 Solutions

🕮 Before You Begin

For introduction to SNMP configuration in Cisco IOS routers refer for VOL1 scenario **System Management > SNMPv2 Server** and **System Management > SNMPv2 Access Control**.

R3 and R6:

```
!
! Configure the R/W communities
!
snmp-server community CISCO_RO RO 2
snmp-server community CISCO_RW RW 2
!
! Enable HSRP traps
!
snmp-server enable traps hsrp
!
! Specify trap destinations
!
snmp-server host 141.1.7.100 CISCO tty
snmp-server host 141.1.77.100 version 2c CISCO hsrp
!
access-list 2 permit 141.1.77.100
access-list 2 permit 141.1.7.100
```

Task 7.1 Breakdown

This configuration is similar to SNMP configurations from earlier labs with the exception of the version option. The default SNMP version is version 1. Also, note that SNMP traps relating to HSRP will only be sent to host 141.1.77.100.

Task 7.1 Verification

Verify the ACL and basic SNMP configuration:

```
Rack1R3#show access-lists 2
Standard IP access list 2
    10 permit 141.1.77.100
    20 permit 141.1.7.100

Rack1R3#show snmp
Chassis: 10578766
<output omitted>
SNMP logging: enabled
    Logging to 141.1.7.100.162, 0/10, 0 sent, 0 dropped.
    Logging to 141.1.77.100.162, 0/10, 0 sent, 0 dropped.

Rack1R3#show snmp community

Community name: ILMI
Community Index: cisco0
Community SecurityName: ILMI
storage-type: read-only  active


Community name: CISCORO
Community Index: cisco1
Community SecurityName: CISCORO
storage-type: nonvolatile        active access-list: 2


Community name: CISCORW
Community Index: cisco2
Community SecurityName: CISCORW
storage-type: nonvolatile        active access-list: 2


Community name: CISCO
Community Index: cisco3
Community SecurityName: CISCO
storage-type: nonvolatile        active

Rack1R3#show snmp host
Notification host: 141.1.7.100  udp-port: 162   type: trap
user: CISCO     security model: v1

Notification host: 141.1.77.100 udp-port: 162   type: trap
user: CISCO     security model: v2c
```


Verify that the configuration commands were accepted:

```
Rack1R3#show running-config | include snmp
snmp-server community CISCO_RO RO 2
snmp-server community CISCO_RW RW 2
snmp-server enable traps hsrp
snmp-server host 141.1.7.100 CISCO  tty
snmp-server host 141.1.77.100 version 2c CISCO  hsrp
```

Task 7.2 Solutions

🕮 Before You Begin

If you need more information about Cisco Menu configuration, refer to VOL1 scenario **`System Management > IOS Menus`**.

R2:

```
username NOC password 0 CISCO
username NOC autocommand menu NOC
!
menu NOC title # Menu for NOC users #
menu NOC prompt # Choose your selection: #
menu NOC text 1. Ping R5
menu NOC command 1. ping 150.1.5.5
menu NOC options 1. pause
menu NOC text 2. Ping R6
menu NOC command 2. ping 150.1.6.6
menu NOC options 2. pause
menu NOC text 3. Traceroute to R5
menu NOC command 3. trace 150.1.5.5
menu NOC options 3. pause
menu NOC text 4. Traceroute to R6
menu NOC command 4. trace 150.1.6.6
menu NOC options 4. pause
menu NOC text 5. Exit
menu NOC command 5. exit
menu NOC clear-screen
!
! Local authentication required to engage the auto-command
! since the auto-command applies to the user
!
line vty 0 4
 login local
```


Task 7.2 Breakdown

This is a standard menu configuration. Ensure that `login local` is configured under the VTY lines to prompt for the username along with the password. Depending on the IOS version and hardware platform, there are usually more that 5 VTY lines. You may want to ask if something should be applied to ALL VTY lines. Here, we are just applying to the first five. Also, the autocommand is needed to activate the menu once the NOC user logs in. Technically, the autocommand could have been configured under the VTY line itself to meet the requirements of this section. However, binding the autocommand to the username allows more flexibility as to which users must use the menu.

Pitfall

When using menus, ensure that the user is given an option to exit the menu.

Task 7.2 Verification

Telnet to R2 to verify the menu:

```
Rack1R3#telnet 150.1.2.2
Trying 150.1.2.2 ... Open

User Access Verification

Username: NOC
Password:
Menu for NOC users

    1.          Ping R5

    2.          Ping R6

    3.          Traceroute to R5

    4.          Traceroute to R6

    5.          Exit
```

🕮 Deep Dive

- What is the limitation for executing CLI commands from the menus?

Task 7.3 Solutions

🕮 Before You Begin

To better understand the use of IP aliasing you may check the scenario **IP Services > NAT and IP Aliasing** from VOL1.

R2:

```
ip alias 141.1.0.22 23
```


Task 7.3 Breakdown

The **ip alias** command will allow the router to answer for the aliased IP address. Although this command is not commonly used, it could be useful in a situation where users need to access a particular async line by telneting to the DNS name along with the standard TCP port of 23. Normally you see IP aliases created automatically for static NAT entries matching the locally connected subnets.

```
Rack1AS#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1AS(config)#ip alias 172.16.2.122 2008
Rack1AS(config)#ip host AS-Line8 172.16.2.122
Rack1AS(config)#^Z
Rack1AS#telnet AS-Line8
Trying AS-Line8 (172.16.2.122)... Open

Rack1SW2#

Rack1AS#show sessions
Conn Host                Address             Byte  Idle Conn Name
*  8 AS-Line8            172.16.2.122           0     0 AS-Line8

Rack1AS#
```

🕮 Deep Dive

- How aliasing is different from secondary IP addresses?

Task 7.3 Verification

Verify that the alias is configured:

```
Rack1R2#show ip aliases
Address Type              IP Address      Port
Interface                 141.1.0.2
Interface                 141.1.25.2
Interface                 150.1.2.2
Alias                     141.1.0.22      23
Interface                 141.1.123.2
```

Verify the ARP entry for the aliased IP:

```
Rack1R2#show ip arp 141.1.0.2
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  141.1.0.2               -   0004.27b5.2fa0  ARPA
FastEthernet0/0
```

Finally telnet from R5 to 141.1.0.2:

```
Rack1R5#telnet 141.1.0.2
Trying 141.1.0.2 ... Open


User Access Verification

Username: NOC
Password:
Menu for NOC users
<output omitted>
```


Task 7.4 Solution

🕮 Before You Begin

Some basic HSRP configurations along with breakdowns are provided in the VOL1 scenario **IP Services > HSRP**.

R3:

```
interface FastEthernet0/1
 standby 1 ip 141.1.36.1
 standby 1 priority 150
 standby 1 preempt
 standby 2 ip 141.1.36.2
```

R6:

```
interface FastEthernet0/0
 standby 1 ip 141.1.36.1
 standby 2 ip 141.1.36.2
 standby 2 priority 150
 standby 2 preempt
```


Task 7.4 Breakdown

This section will require HSRP with R3 active for one IP address (141.1.36.1) and R6 active for another IP address (141.1.36.2). Therefore, the appropriate DHCP server for this segment can assign one address as the default gateway to a certain pool of clients, while assigning the other address as the default gateway for another pool of clients. Note that these addresses have been arbitrarily chosen. If R3 becomes unavailable, R6 will take over for 141.1.36.1 and vice versa. The HSRP priority is set above the default of 100 along with the preempt option, so that R3 and R6 can take back over for their HSRP group after becoming available again.

Note

Without the HSRP preempt option, a router will not take over for an HSRP group even if its own priority is higher than the current active router.

Task 7.4 Verification

Verify the HSRP configuration:

```
Rack1R6#show standby brief
                     P indicates configured to preempt.
                     |
Interface   Grp  Pri P State   Active          Standby         Virtual IP
Fa0/0       1    100   Standby 141.1.36.3      local           141.1.36.1
Fa0/0       2    150 P Active  local           141.1.36.3      141.1.36.2

Rack1R6#show standby
FastEthernet0/0 - Group 1
  State is Standby
    1 state change, last state change 00:00:13
  Virtual IP address is 141.1.36.1
  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 1.368 secs
  Preemption disabled
  Active router is 141.1.36.3, priority 150 (expires in 9.108 sec)
  Standby router is local
  Priority 100 (default 100)
  IP redundancy name is "hsrp-Fa0/0-1" (default)
FastEthernet0/0 - Group 2
  State is Active
    1 state change, last state change 00:00:30
  Virtual IP address is 141.1.36.2
  Active virtual MAC address is 0000.0c07.ac02
    Local virtual MAC address is 0000.0c07.ac02 (v1 default)
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 2.112 secs
  Preemption enabled
  Active router is local
  Standby router is 141.1.36.3, priority 100 (expires in 7.112 sec)
  Priority 150 (configured 150)
  IP redundancy name is "hsrp-Fa0/0-2" (default)
```

🕮 Deep Dive

- What HSRP states are sending periodic Hello Messages?
- What is the purpose of HSRP Learn state?

Task 7.5 Solutions

🕮 Before You Begin

Refer to VOL1 scenario **System Management > Telnet Service Option** for more information on various telnet connetion options.

```
R3:
ip host R4 150.1.4.4
!
busy-message R4 "Connection Unsuccessful"
```

Task 7.5 Breakdown

The "host failed" message (aka busy-message) is used to display a custom message with a telnet connection fails to a particular host. The key to the **busy-message** is that the **busy-message** is configured on the host attempting to make the connection, and not the host that actually refused the connection. To have the router that refused the connection display a "line in use" message, use the **refuse-message** under the remote router's VTY line.

The **busy-message** only takes a hostname and will not take an IP address. This means the **ip host** command will need to be configured in conjunction with the **busy-message** command assuming a DNS server has not been defined.

Task 7.5 Verification

Shutdown Serial 1/0 on R3, and try connecting to R4 by name:

```
Rack1R3(config)#interface s1/0
Rack1R3(config-if)#shutdown

Rack1R3#R4
Translating "R4"
 Connection Unsuccessful
```


Task 8.1 Solutions

🕮 Before You Begin

For detailed description and configuration examples for the WRED feature refer to VOL1 scenario **QoS > Legacy Random Early Detection**.

```
R1:
interface FastEthernet0/0
 random-detect
 random-detect precedence 0 15 40 10
```

Task 8.1 Breakdown

The section will require Weighted Random Early Detection (WRED) to be configured. Random early detection is designed to avoid tail drop by randomly dropping packets prior to the output queue of an interface actually becoming full. When the output queue becomes full, all packets that attempt to enter the tail (end) of the queue will have to be dropped, hence the term tail drop. If these dropped packets were all TCP packets, then theoretically all TCP sessions that had dropped packets will back off and perform TCP slow start. With TCP slow start, a TCP session will start with a small TCP window size and gradually increase the window size until congestion or delay is experienced. If all TCP sessions back off and then start up again in synchronization with each other, the same situation will occur once the interface's queue to become full again. This results in periods of high congestion followed by periods of low utilization. This type of behavior is called global synchronization, and is what random early detection is designed to prevent.

Note

Weighted RED is the process of using different queues and thresholds for packets with different IP precedence values.

Task 8.1 Verification

Verify WRED settings on the interface:

```
Rack1R1#show queueing interface FastEthernet0/0
Interface FastEthernet0/0 queueing strategy: random early detection
(WRED)
    Exp-weight-constant: 9 (1/512)
    Mean queue depth: 0

  class          Random drop      Tail drop    Minimum Maximum  Mark
                  pkts/bytes       pkts/bytes   thresh  thresh   prob
      0            0/0              0/0           15      40     1/10
      1            0/0              0/0           22      40     1/10
      2            0/0              0/0           24      40     1/10
      3            0/0              0/0           26      40     1/10
      4            0/0              0/0           28      40     1/10
      5            0/0              0/0           31      40     1/10
      6            0/0              0/0           33      40     1/10
      7            0/0              0/0           35      40     1/10
   rsvp            0/0              0/0           37      40     1/10
```

🕮 Deep Dive

- What is WRED ECN marking?

Task 8.2 Solutions

🕮 Before You Begin

NBAR classification is straightforward to configure. You may read more about NBAR basic functioning in VOL1 scenario **IP Services > NBAR Protocol Discovery**.

R5:

```
!
! Matching SMTP protocol means activating NBAR engine
!
class-map match-all SMTP
 match protocol smtp
!
policy-map QoS
 class SMTP
  bandwidth 1500
!
interface FastEthernet0/1
 service-policy output QoS
```

Task 8.2 Breakdown

This is a standard MQC configuration, and dictates that SMTP traffic will be guaranteed 1500Kbps of the output queue of the FastEthernet0/1 interface in the case of congestion. SMTP may use more than 1500Kbps, however only 1500Kbps is guaranteed in the case of congestion.

🖉 Note

When matching a protocol in a class-map (and thus using NBAR), CEF should be enabled globally. CEF is on by default in newer IOS versions, but it is possible that it was disabled in a pre-loaded initial configuration file. Make sure to verify that CEF is enabled.

Task 8.2 Verification

Verify that the policy-map is attached and the class-map is configured:

```
Rack1R5#show policy-map interface Fa0/1
 FastEthernet0/1

  Service-policy output: QoS

    Class-map: SMTP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol smtp
      Queueing
        Output Queue: Conversation 265
        Bandwidth 1500 (kbps) Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0

    Class-map: class-default (match-any)
      23 packets, 2310 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

🕮 Deep Dive

- How NBAR interprets flow direction?

Task 8.3 Solutions

🕮 Before You Begin

For more information on MQC classification option and the three-color policer please refer to VOL1 scenarios **QoS > MQC Classification and Marking** and **QoS > MQC Single Rate Three Color Policer**.

```
R5:
!
! Match large packets (notice the match size is "inclusive)
!
class-map match-all ABOVE_1250_BYTES
 match packet length min 1251
!
policy-map QoS
 class ABOVE_1250_BYTES
  police cir 2500000
```

Task 8.3 Breakdown

Policing packets based on the packet's size can be useful in situations where framentation is not possible (i.e. an HDLC link). Large packets can cause excessive delay on slow speed links due to serization delay. In this section, packets larger than 1250 bytes will be limited to 2.5 Mbps. The other feature that allows matching packets based on their size is policy based routing (PBR).

☠ Pitfall

The IOS is not consistent in how QoS parameters are entered. Some commands take values in as bits, some in as bytes, and some in as kilobytes. When in the CCIE lab, use the '?' along with the command to view how the parameter is accepted by the IOS for that particular command. Do not expect the values given in the lab itself to be the same as what will need to be entered in the configuration. Entering a value in as bytes when the command takes bits will cause you to lose points for that section.

Task 8.3 Verification

Verify that the policy-map is configured and applied:

```
Rack1R5#show policy-map interface Fa0/1
 FastEthernet0/1

  Service-policy output: QoS

  <output omitted>

    Class-map: ABOVE_1250_BYTES (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: packet length min 1251
      police:
          cir 2500000 bps, bc 78125 bytes
        conformed 0 packets, 0 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          drop
        conformed 0 bps, exceed 0 bps

    Class-map: class-default (match-any)
      220 packets, 21377 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```


Simulate a packet stream from R4 and verify the statistics:

```
Rack1R4#ping 141.1.0.8 size 1250 repeat 10000 timeout 1

Type escape sequence to abort.
Sending 10000, 1250-byte ICMP Echos to 141.1.0.8, timeout is 1 seconds:
!!!!!!!!!!!!

Rack1R5#show policy-map interface Fa0/1
 FastEthernet0/1

  Service-policy output: QoS

<output omitted>

    Class-map: ABOVE_1250_BYTES (match-all)
      27 packets, 34128 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: packet length min 1251
      police:
          cir 2500000 bps, bc 78125 bytes
        conformed 0 packets, 0 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          drop
        conformed 0 bps, exceed 0 bps
<output omitted>
```



```
Rack1R4#ping 141.1.0.8 size 1251 repeat 10000 timeout 1

Type escape sequence to abort.
Sending 10000, 1251-byte ICMP Echos to 141.1.0.8, timeout is 1 seconds:
!!!!!!!!!!!!

Rack1R5#show policy-map interface Fa0/1
 FastEthernet0/1

  Service-policy output: QoS

<output omitted>

    Class-map: ABOVE_1250_BYTES (match-all)
      27 packets, 34128 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: packet length min 1250
      police:
          cir 2500000 bps, bc 78125 bytes
        conformed 27 packets, 34128 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          drop
        conformed 0 bps, exceed 0 bps
<output omitted>
```


Task 8.4 Solution

🕮 Before You Begin

For introduction to payload compression techniques over serial links, refer to VOL1 scenario **QoS > Payload Compression on Serial Links**

R4 and R5:

```
interface Serial0/1/0
 compress predictor
```

Task 8.4 Breakdown

There are two common types of compression used with PPP, stacker and predictor. Stacker is more CPU intensive but more forgiving on memory utilization. Predictor is less CPU intensive but utilizes more memory. The key to determining which to use here is based on the word “guessing” that is used in the task. Predictor will try to predict the next sequence of characters in the data stream. Although “predicting” is not the same as “guessing”, the use of the word leads us to interrupt the task to want predictor to be used over stac.

Task 8.4 Verification

Check the compression configuration status:

```
Rack1R4#show compress
 Serial0/1/0
        Software compression enabled
        uncompressed bytes xmt/rcv 327/332
        compressed bytes   xmt/rcv 0/0
        Compressed bytes sent:         0 bytes   0 Kbits/sec
        Compressed bytes recv:         0 bytes   0 Kbits/sec
        1  min avg ratio xmt/rcv 0.709/0.922
        5  min avg ratio xmt/rcv 0.709/0.922
        10 min avg ratio xmt/rcv 0.709/0.922
        no bufs xmt 0 no bufs rcv 0
        resyncs 0
```

Generate packet flow and ensure it is getting compressed:

```
Rack1R4#ping 141.1.45.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 141.1.45.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/16/16 ms

Rack1R4#show compress
 Serial0/1/0
        Software compression enabled
        uncompressed bytes xmt/rcv 837/842
        compressed bytes   xmt/rcv 170/168
        Compressed bytes sent:  170 bytes   0 Kbits/sec  ratio: 4.923
        Compressed bytes recv:  168 bytes   0 Kbits/sec  ratio: 5.011
        1  min avg ratio xmt/rcv 1.190/1.482
        5  min avg ratio xmt/rcv 1.190/1.482
        10 min avg ratio xmt/rcv 1.190/1.482
        no bufs xmt 0 no bufs rcv 0
        resyncs 0
```

🕮 Deep Dive

- How effective payload compression could be?

VOL1 Scenario Reference

Bridging and Switching > VTP
Bridging and Switching > VTP Pruning
Bridging and Switching > VTP Pruning Eligible List.
Bridging and Switching > STP Load Balancing with Port Cost.
Bridging and Switching > Unidirectional Link Detection
Bridging and Switching > STP Loop Guard
Bridging and Switching > STP Load Balancing with Port Priority
Bridging and Switching > Storm Control
QoS > Catalyst QoS Port-Based Classification
OSPF > Not So Stubby Areas
OSPF > Not So Stubby Areas and Default Routing
OSPF > Repairing Discontiguous Area with Virtual Links
IPv6 > OSPFv3
IPv6 > OSPFv3 over NBMA
IPv6 > OSPFv3 Summarization
MPLS VPN > PE-CE Routing with RIP
RIPv2 > RIPv2 Offset list
MPLS VPN > MPLS LDP
MPLS VPN > MP-BGP VPNv4
BGP > BGP Local-AS
MPLS VPN > PE-CE Routing with BGP
Multicast > AutoRP
Multicast > AutoRP – Multiple Candidate RPs
Multicast > AutoRP - Filtering Candidate RPs
Multicast > PIM NBMA-Mode
Multicast > Multicast Rate-Limiting
Security > Zone Based Firewall
Security > Preventing Packet Spoofing
Security > Control Plane Policing
System Management > SNMPv2 Server
System Management > SNMPv2 Access Control
System Management > IOS Menus
IP Services > NAT and IP Aliasing
IP Services > HSRP
System Management > Telnet Service Option
QoS > Legacy Random Early Detection
IP Services > NBAR Protocol Discovery
QoS > MQC Classification and Marking
QoS > MQC Single Rate Three Color Policer
QoS > Payload Compression on Serial Links

Deep Dive Answers

- Why VTP might be impractical in modern switching environment?
  - VTP assumes “network-wide” VLAN model, where VLANs span multiple switches and locations. This design model has been abandoned long time ago due to unstable networking environments it creates. Nowadays VLANs tend to be isolated within a single switch or functional block.
- What is the danger of using VTP?
  - VTP is “broadcast” in its nature, propagating configuration messages across the whole switching domain. This may result in sudden configuration erasures if a new switch is plugged in the domain. Using VTP passwords or VTP transparent mode is normally recommended in production environment.
- Will changing cost affect any downstream bridges?
  - Yes, changing cost on the local bridge may affect path cost to the root bridge and make downstream bridges elect different root ports. Therefore, any change to the STP costs should be planned carefully, knowing the results in advance.
- If you are not allowed to change the STP link cost, what other parameter you may tune to achieve the same effect?
  - STP link cost is based on the port speed, so you may change the speed, which will automatically reflect in the port cost.
- Is there a reason to prefer storm-control over policing to rate-limit ingress traffic?
  - Storm-control’s benefit is the fact that you may apply it to “broadcast” and “multicast” traffic, which is not possible with traffic policing. Additionally, storm-control supports rate limiting based on the packet per second rate.
- How does storm-control feature meter the incoming traffic?
  - Metering is based on 1-second averaging of traffic counters. You cannot change this interval, like you can do with traffic policing feature.
- When would you use the IP precedence to DSCP table?
  - This table is normally used when you trust IP precedence value in IP packets. The new DSCP value is used to calculate the QoS label for the packet and update the DSCP field.
- How could the DSCP to CoS mapping table be used to set CoS marking on non-IP packets?
  - There is no “set cos” operation in the 3560 switches, but you may use the “set dscp” command to change CoS. The CoS value is calculated and imposed based on the DSCP to CoS mapping table.

- What is the difference in trusting IP precedence and DSCP values?
  - Trusting IP precedence is based on the topmost 3 bits in the ToS byte and maps to DSCP values using the IPP to DSCP mapping table. Trusting DSCP does not need any mapping table and calculates the QoS label directly on the ToS byte's topmost 6 bits.
- How is IP Precedence QoS model different from DiffServ?
  - IP Precedence QoS model is based on "military" concept of precedence level, where a higher-precedence level flow preempts all lower precedence levels. Effectively, a flow with highest precedence may claim all link resources, unless there is a higher precedence flow. DiffServ introduces traffic aggregates and Assured Forwarding classes, where every AF class is guaranteed some bandwidth, and only the EF class may preempt others.
- What could be the purpose of OSPF NSSA area in real-word deployments?
  - NSSA areas normally represent DMZ blocks connecting to another network or to the ISP. It is common to see the default route learned via the NSSA area, or some routes leaked from the partner company, but the area still benefits stub property and does not need the whole domain's routing information.
- What other way you could use in place of virtual-link or tunnel to connect a non-zero area to OSPF backbone?
  - Instead of creating layer 3 tunnels, you may want to create a layer 2 virtual connection, such as PVC or VLAN and assign it to the same area that you want to connect to the backbone. This will allow for using a direct "physical" connection to maintain connectivity to the rest of the topology.
- What important improvement has been made to OSPFv3 as compared to OSPFv2?
  - OSPFv3 advertises topology and reachability information separately. The classic type-1 and type-2 LSAs are only used to advertise connections to other routers, but not the network prefixes. The network prefixes are advertised in separate LSA type, which allows for better scalability and decoupling of topology from network reachability information.
- Could you use RIPv2 in multihomed VPN scenarios with backdoor links?
  - RIPv2 is not effective in multi-homed environment as there are mechanisms to prevent MP-BGP paths other locally learned and no native mechanism to prevent redistribution routing loops.
- What other tunneling technology you can use to transport MPLS over IP cloud?

  - Layer 2 VPN solutions would work as they allow for multi-protocol transportation. Therefore, you may use L2TPv3 to run MPLS VPNs over non-MPLS enabled core.
- Can you completely replace MPLS forwarding with IP-based tunnels?
  - Yes, MPLS transport is optional to MPLS VPNs, the functionality is implemented using MP-BGP features.
- What could be the problem of carrying RIP metric in MED attribute?
  - For updates transported across a transit AS, MED value could be dropped thus resulting in loss of metric.
- Why as-override option could be dangerous?
  - AS-Override could result in routing loops in multi-homed environments.
- What solution you could use to prevent routing loops with the as-override feature?
  - BGP Site-of-Origin attribute is an extended community that could be used to tag BGP prefixes and detect redistribution routing loops.
- Why don't you need PIM RP Listener in the RP and MA?
  - RP and MA are the sources for the PIM dense groups, and they simply flood traffic out of the interface dedicated for RP/MA announcements (normally a loopback). It's up to other routers to forward multicast traffic. The other routers are the only ones needing the Auto-RP listener configuration.
- How does Auto-RP interact with static RP configuration?
  - By default, if a static RP is configured locally and Auto-RP message is received, the latter preempts the static configuration.
- Can you flood Auto-RP groups without ever using PIM dense mode?
  - Yes, if you statically configure the RPs for the Auto-RP groups. Make sure the Auto-RP announcements do not preempt the static settings though.
- Could PIM NBMA be useful on broadcast interfaces, e.g. Ethernet?
  - In some situations, for example when you are using private VLANs or protected ports, the broadcast segment is more like NBMA topology. In this case, using PIM NBMA mode could be beneficial.
- How does multicast-rate limit define traffic policing window?
  - Multicast rate-limit is not very granular; it averages the multicast traffic counters every minute and uses this as the traffic rate. It is also not very performance effective, compare the rate-limit command.
- Can you filter MPLS traffic using IP access-lists?
  - No, MPLS traffic is non-IP and you cannot filter the encapsulated packets based on the IP addresses or port numbers.

- Where should you perform all security and QoS decisions in the MPLS network?
    - Normally, security classification and marking is performed at the edge of MPLS network, and the core nodes only apply PHB behavior based on Diff-Serv policies configured based on MPLS EXP bits.
- Why control plane security is so important?
    - Network infrastructure is critical to providing endpoint connectivity, and therefore any attack mounted against the infrastructure will affect the endpoints as well.
- What other control plane security mechanisms you know?
    - General TTL Security Mechanism could be among the other ones, combined with Selective Packet Discard settings and network QoS settings to prioritize control traffic.
- What is the limitation for executing CLI commands from the menus?
    - You can only execute the exec CLI commands, but not the configuration mode commands. You are only limited to one-line commands.
- How aliasing is different from secondary IP addresses?
    - In UNIX systems you often hear term alias used for “secondary ip addressing. In Cisco IOS, IP alias is an address from the main interface address range. It’s like a second address on the same subnet, but it does not create a new subnet on an interface.
- What HSRP states are sending periodic Hello Messages?
    - All but Init, Listen and Learn. This leaves just Speak, Standby and Active – these are the stats that HSRP exchanges hello messages in.
- What is the purpose of HSRP Learn state?
    - In HSRP Learn state the router attempts to discover the standby IP address from the active router.
- What is WRED ECN marking?
    - As opposed to dropping IP packets, WRED may set special bit in the ToS byte (ECN bit) to signal the network congestion. It’s up to the receiving host to properly interpret this bit and respond with packet having ECN echo bit set. The sender should respond to the ECN echo bits by slowing down the sending rate. This allows avoiding long retransmissions and throttling and improving application performance. However, it requires participation from network hosts.
- How NBAR interprets flow direction?

- NBAR classification decision is associated with traffic flow. As soon as flow is identified as belonging to a protocol, it is matched in both ingress and egress policies that match this protocol.

Lab 5 Solutions

Task 1.1 Solutions

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

Bridging and Switching > Layer 2 Etherchannel
Bridging and Switching > 802.1q Trunking
Bridging and Switching > 802.1q Native VLAN

SW1:

```
interface Port-channel12
 switchport trunk encapsulation isl
 switchport mode trunk
!
interface Port-channel13
 switchport trunk encapsulation isl
 switchport mode trunk
!
interface Port-channel14
 switchport trunk encapsulation isl
 switchport mode trunk
!
interface range Fa0/13
 switchport trunk encapsulation isl
 switchport mode trunk
 channel-group 12 mode on
 no shutdown
!
interface range Fa0/14
 switchport trunk encapsulation isl
 switchport mode trunk
 channel-group 12 mode on
 no shutdown
!
interface range Fa0/16
 switchport trunk encapsulation isl
 switchport mode trunk
 channel-group 13 mode on
 no shutdown
```



```
interface range Fa0/17
 switchport trunk encapsulation isl
 switchport mode trunk
 channel-group 13 mode on
 no shutdown
!
interface range Fa0/19
 switchport trunk encapsulation isl
 switchport mode trunk
 channel-group 14 mode on
 no shutdown
!
interface range Fa0/20
 switchport trunk encapsulation isl
 switchport mode trunk
 channel-group 14 mode on
 no shutdown
```

SW2:

```
interface Port-channel12
 switchport trunk encapsulation isl
 switchport mode trunk
!
interface range Fa0/13
 switchport trunk encapsulation isl
 switchport mode trunk
 channel-group 12 mode on
 no shutdown
!
interface range Fa0/14
 switchport trunk encapsulation isl
 switchport mode trunk
 channel-group 12 mode on
 no shutdown
```

SW3:

```
interface Port-channel13
 switchport trunk encapsulation isl
 switchport mode trunk
!
interface range Fa0/13
 switchport trunk encapsulation isl
 switchport mode trunk
 channel-group 13 mode on
 no shutdown
!
interface range Fa0/14
 switchport trunk encapsulation isl
 switchport mode trunk
 channel-group 13 mode on
 no shutdown
```


SW4:

```
interface Port-channel14
 switchport trunk encapsulation isl
 switchport mode trunk
!
interface range Fa0/13 - 14
 switchport trunk encapsulation isl
 switchport mode trunk
 channel-group 14 mode on
 no shutdown
```


Task 1.1 Breakdown

The first step in creating a layer 2 EtherChannel is to apply the **channel-group** command to the interface. The *on* mode of the channel will disable the usage of both PAgP and LACP.

Next, configuration that should apply to both the channel interface and the member interfaces should be placed on the *port-channel* interface. In the above example the trunking configuration is shown on both the physical and logical interfaces for clarity. Options configured on the *port-channel* interface will be automatically inherited by the physical member interfaces.

The phase 'traffic sent over these trunk links should be tagged with a VLAN header' implies that ISL trunking must be used. By default only ISL tags all VLANs sent over the trunk link with a VLAN header (remember dot1q uses the native VLAN). However, in certain circumstances such as 802.1q tunneling, the native VLAN can carry a VLAN header by issuing the global command **vlan dot1q tag native**. This case will be covered in later labs. In this scenario, we are implicity told not to use any global commands, and therefore ISL trunks have been selected as the solution.

Notice that in addition to L2 Etherchannels a set of L3 links has been preconfigured among the switches.

🕮 Deep Dive

- What is the other option to ensure all frames are tagged when sent across a 802.1Q trunk?
- Why would you always want to tag frames sent across a trunk?

Task 1.1 Verification

Verify all of the Etherchannel bundles:

```
Rack1SW1#show etherchannel summary | begin Group
Group  Port-channel  Protocol    Ports
------+-------------+-----------+--------------------
12     Po12(SU)         -        Fa0/13(P)   Fa0/14(P)
13     Po13(SU)         -        Fa0/16(P)   Fa0/17(P)
14     Po14(SU)         -        Fa0/19(P)   Fa0/20(P)
41     Po41(RU)       LACP       Fa0/21(P)

Rack1SW2#show etherchannel summary | beg Group
Group  Port-channel  Protocol    Ports
------+-------------+-----------+----------------------
12     Po12(SU)         -        Fa0/13(P)   Fa0/14(P)
32     Po32(RU)       LACP       Fa0/16(P)   Fa0/17(P)   Fa0/18(P)

Rack1SW3#show etherchannel summary | beg Group
Group  Port-channel  Protocol    Ports
------+-------------+-----------+--------------------
13     Po13(SU)         -        Fa0/13(P)   Fa0/14(P)
32     Po32(RU)       LACP       Fa0/16(P)   Fa0/17(P)   Fa0/18(P)
43     Po43(RU)       LACP       Fa0/19(P)   Fa0/20(P)   Fa0/21(P)

Rack1SW4#show etherchannel summary | beg Group
Group  Port-channel  Protocol    Ports
------+-------------+-----------+--------------------
14     Po14(SU)         -        Fa0/13(P)   Fa0/14(P)
41     Po41(RU)       LACP       Fa0/15(P)
43     Po43(RU)       LACP       Fa0/19(P)   Fa0/20(P)   Fa0/21(P)
```


Verify that the Ethrchannel is formed without any protocol negotiation:

```
Rack1SW1#show etherchannel protocol
                Channel-group listing:
                ----------------------

Group: 12
----------
Protocol:   -   (Mode ON)

Group: 13
----------
Protocol:   -   (Mode ON)

Group: 14
----------
Protocol:   -   (Mode ON)

Group: 41
----------
Protocol:  LACP
```

Check that interfaces are trunking, without negotiation and ISL is the protocol:

```
Rack1SW1#show interfaces trunk

Port        Mode         Encapsulation  Status        Native vlan
Po12        on           isl            trunking      1
Po13        on           isl            trunking      1
Po14        on           isl            trunking      1

Port        Vlans allowed on trunk
Po12        1-4094
Po13        1-4094
Po14        1-4094

Port        Vlans allowed and active in management domain
Po12        1,4,27,38,162,2005
Po13        1,4,27,38,162,2005
Po14        1,4,27,38,162,2005

Port        Vlans in spanning tree forwarding state and not pruned
Po12        1,4,27,38,162,2005
Po13        1,4,27,38,162,2005
Po14        1,4,27,38,162,2005
```


Task 1.2 Solutions

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

Bridging & Switching > MAC Address Table Static Entries & Aging

```
SW2:
!
! Change port-channel load-balancing mode
!
port-channel load-balance dst-ip
!
! Change MAC-address table aging timers for VLANs 8 and 88
!
mac-address-table aging-time 10 vlan 8
mac-address-table aging-time 10 vlan 88
```

Task 1.2 Breakdown

For this task, traffic from the file server located behind BB2 will be sent across the trunk with the source MAC address of BB2's FastEthernet interface. By default, all of this traffic would use only one of the EtherChannel trunk links since the default is to load balance based on the source MAC address. With destination IP-address based load balancing enabled on SW2, this traffic will now be distributed across both links.

The Content Addressable Memory (CAM) table is where the switch stores learned MAC addresses. This table is used as a "routing" table for the switch, and is used to determine the outgoing interface for a frame. When a unicast frame comes in that does not have an entry in the CAM table, it is treated as a broadcast frame. A broadcast frame is sent out all interfaces except the one it was received on. When the CAM table is full, all excess unicast frames are treated as broadcast frames. When this occurs, it is possible for traffic to leak between VLANs. The **mac-address-table aging-time** determines how long an idle MAC address can remain in the CAM table, and defaults to five minutes. In the above task this value is adjusted to flush inactive entries out of VLANs 8 and 88 after just 10 seconds.

🕮 Deep Dive

- In what scenarios would you want to disable MAC address learning at all?

Task 1.2 Verification

Verify the load balancing configuration:

```
Rack1SW2#show etherchannel load-balance
EtherChannel Load-Balancing Operational State (dst-mac):
Non-IP: Destination MAC address
  IPv4: Destination MAC address
  IPv6: Destination IP address
```

Verify the MAC address table aging time for VLANs 8 and 88:

```
Rack1SW2#show mac-address-table aging-time vlan 8
Vlan    Aging Time
----    ----------
   8       10

Rack1SW2#show mac-address-table aging-time vlan 88
Vlan    Aging Time
----    ----------
  88       10
```


Task 2.1 Solutions

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

RIPv2 > RIPv2 Authentication
EIGRP > EIGRP Metric Weights
OSPF > OSPF Stub Areas
RIPv2 > RIPv2 Unicast Updates

R1:

```
!
! Change EIGRP weights, bw is 3 times higher than other metrics
!
router eigrp 200
 metric weights 0 3 1 1 0 0
!
! Authenticate RIPv2 updates
!
key chain RIP
 key 1
  key-string CISCO
!
interface FastEthernet0/0
 ip rip authentication mode md5
 ip rip authentication key-chain RIP
!
! Configure RIP for static neighbors
!
router rip
 version 2
 passive-interface default
 network 192.10.1.0
 neighbor 192.10.1.254
 neighbor 192.10.1.6
 no auto-summary
```

R2:

```
!
! Area 27 is stub, do not send summaries
!
router ospf 1
 area 27 stub no-summary
 network 162.1.27.2 0.0.0.0 area 27
```

R3:

```
!
! Change EIGRP weights, bw is 3 times higher than other metrics
!
router eigrp 200
 metric weights 0 3 1 1 0 0
```


R4:

```
!
! Enable RIP in R4
!
router rip
 version 2
 network 204.12.1.0
 no auto-summary
```

R6:

```
!
! Authenticate R6 RIP exchange with R1 and BB2
!
key chain RIP
 key 1
  key-string CISCO
!
interface FastEthernet0/0
 ip rip authentication mode md5
 ip rip authentication key-chain RIP
!
! Configure static RIP neighbors on R6
!
router rip
 version 2
 passive-interface default
 no passive-interface Serial0/0/0.1
 network 54.0.0.0
 network 150.1.0.0
 network 192.10.1.0
 network 162.1.0.0
 neighbor 192.10.1.1
 neighbor 192.10.1.254
 no auto-summary
```

SW1:

```
router ospf 1
 area 27 stub
 network 150.1.7.7 0.0.0.0 area 27
 network 162.1.27.7 0.0.0.0 area 27
```

SW2:

```
router eigrp 200
 metric weights 0 3 1 1 0 0
```


Task 2.1 Breakdown

The task dictates that SW1 does not meet specific reachability information about the rest of the network. As previously discussed, an LSA cannot be removed from the OSPF database on a per device basis. Instead, this must be accomplished by defining a stub area.

Since the above task also states that SW2 should only see a default route as generated by R2, it is evident that external or inter-area routing information should not be allowed into OSPF area 27. This requirement immediately eliminates the stub and not-so-stubby area types, as these two do allow inter-area reachability information to enter the area. Therefore, the only two options remaining are a totally stubby area or a not-so-totally-stubby area. As there is no redistribution occurring into OSPF area 27, the totally-stubby area has been chosen in the above sample solution. However as there is no restriction to eliminate a not-so-totally-stubby area, that would also be a valid solution.

EIGRP metric calculation uses a composite of four values, bandwidth, delay, load, and reliability. By default EIGRP only uses bandwidth and delay in its metric calculation; however this behavior can be modified by changing the **`metric weights`** under the EIGRP process.

Pitfall

Metric weighting must match between devices in the EIGRP domain in order to establish adjacency. Therefore if you modify the metric weights parameter on one EIGRP device you must do so on all EIGRP devices in that autonomous system.

Pitfall

When configuring RIPv2 authentication, make sure you created the key-chain ahead of applying it an interface. The order of operations is important since in most IOS versions change to the key-chain do not propagate to RIP process unless you re-apply the key-chain to the interface.

The EIGRP metric calculation formula is as follows:

```
(k1 * bandwidth + (k2 * bandwidth)/(256 - load) + k3 *
delay) * (k5/(reliability + k4))
```

The latter half of the calculation is only performed if k5 does not equal 0. The variable definitions of the above formula are as follows:

Bandwidth: The inverse of the lowest interface bandwidth along the path for the prefix times 2.56 x 10^{12} in bits per second.

Delay: The cumulative interface delay along the entire path of the prefix in tens of microseconds.

Reliability: Reliability of local interface as a fraction of 255.

Load: Load of local interface as a fraction of 255.

The above values can be determined for a prefix as follows:

```
Rack1R6#show ip route eigrp
D    200.0.0.0/24 [90/146432] via 54.1.1.254, 00:00:09, Serial0/0/0
D    200.0.1.0/24 [90/146432] via 54.1.1.254, 00:00:09, Serial0/0/0
D    200.0.2.0/24 [90/146432] via 54.1.1.254, 00:00:09, Serial0/0/0
D    200.0.3.0/24 [90/146432] via 54.1.1.254, 00:00:09, Serial0/0/0

Rack1R6#show ip route 200.0.0.0
Routing entry for 200.0.0.0/24
  Known via "eigrp 10", distance 90, metric 2297856, type internal
  Redistributing via eigrp 10
  Last update from 54.1.1.254 on Serial0/0/0, 00:00:19 ago
  Routing Descriptor Blocks:
  * 54.1.1.254, from 54.1.1.254, 00:00:19 ago, via Serial0/0/0
     Route metric is 2297856, traffic share count is 1
     Total delay is 25000 microseconds, minimum bandwidth is 1544 Kbit
     Reliability 255/255, minimum MTU 1500 bytes
     Loading 1/255, Hops 1
```

🕮 Deep Dive

- What type of areas would you most likely use in production?
- How can you reduce LSA flooding

Task 2.1 Verification

Verify that area 27 is a stub area:

```
Rack1R2#show ip ospf | begin Area 27
    Area 27
        Number of interfaces in this area is 1
        It is a stub area, no summary LSA in this area
          generates stub default route with cost 1
```

Verify the routing table on SW1 and confirm that SW1's Loopback has been advertised into OSPF:

```
Rack1SW1#show ip route ospf
O*IA 0.0.0.0/0 [110/2] via 162.1.27.2, 00:01:34, Vlan27

Rack1R2#show ip route 150.1.7.7
Routing entry for 150.1.7.7/32
  Known via "ospf 1", distance 110, metric 2, type intra area
  Last update from 162.1.27.7 on FastEthernet0/0, 1d06h ago
  Routing Descriptor Blocks:
  * 162.1.27.7, from 150.1.7.7, 1d06h ago, via FastEthernet0/0
      Route metric is 2, traffic share count is 1
```

Verify the EIGRP metric weights:

```
Rack1SW2#show ip protocols | beg eigrp
Routing Protocol is "eigrp 200"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Default networks flagged in outgoing updates
  Default networks accepted from incoming updates
  EIGRP metric weight K1=3, K2=1, K3=1, K4=0, K5=0
<output omitted>
```


Verify the RIP configuration on the RIP enabled routers:

```
Rack1R1#show ip protocols | beg rip
Routing Protocol is "rip"
  Sending updates every 30 seconds, next due in 0 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
  Neighbor(s):
    192.10.1.6
    192.10.1.254
  Default version control: send version 2, receive version 2
  Automatic network summarization is not in effect
  Maximum path: 4
  Routing for Networks:
    192.10.1.0
  Passive Interface(s):
    VoIP-Null0
    FastEthernet0/0
    Serial0/0/0
    Serial0/1
    Virtual-Access1
    Loopback0
  Routing Information Sources:
    Gateway         Distance      Last Update
    192.10.1.254         120      00:00:24
    192.10.1.6           120      00:00:18
  Distance: (default is 120)
```


Make sure RIP updates are authenticated:

```
Rack1R1#debug ip rip
RIP: received packet with MD5 authentication
RIP: received v2 update from 192.10.1.254 on FastEthernet0/0
     205.90.31.0/24 via 0.0.0.0 in 7 hops
     220.20.3.0/24 via 0.0.0.0 in 7 hops
     222.22.2.0/24 via 0.0.0.0 in 7 hops
     RIP: received packet with MD5 authentication
RIP: received v2 update from 192.10.1.6 on FastEthernet0/0
     54.1.1.0/24 via 0.0.0.0 in 1 hops
     150.1.6.0/24 via 0.0.0.0 in 1 hops
     162.1.6.0/24 via 0.0.0.0 in 1 hops
     212.18.0.0/24 via 0.0.0.0 in 2 hops
     212.18.1.0/24 via 0.0.0.0 in 2 hops
     212.18.2.0/24 via 0.0.0.0 in 2 hops
     212.18.3.0/24 via 0.0.0.0 in 2 hops

Rack1R1#undebug all
```

Finally verify that updates are being sent as unicast (note that we still receive multicast from BB2 and suppress null updates at R1):

```
Rack1R1(config)#access-list 100 permit udp any eq 520 any eq 520
Rack1R1#debug ip packet detail 100
Rack1R1#debug ip rip events

IP: s=192.10.1.254 (FastEthernet0/0), d=224.0.0.9, len 132, rcvd 2
      UDP src=520, dst=520
RIP: received v2 update from 192.10.1.254 on FastEthernet0/0
RIP: Update contains 3 routes
IP: tableid=0, s=192.10.1.6 (FastEthernet0/0), d=192.10.1.1
(FastEthernet0/0), routed via RIB
IP: s=192.10.1.6 (FastEthernet0/0), d=192.10.1.1 (FastEthernet0/0), len
212, rcvd 3 UDP src=520, dst=520
RIP: received v2 update from 192.10.1.6 on FastEthernet0/0
RIP: Update contains 7 routes
RIP: sending v2 update to 192.10.1.6 via FastEthernet0/0 (192.10.1.1) -
suppressing null update
RIP: sending v2 update to 192.10.1.254 via FastEthernet0/0 (192.10.1.1)
- suppressing null update
```


Task 2.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following posts at INE blog:

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

http://blog.ine.com/2008/02/19/understanding-redistribution-part-ii/

http://blog.ine.com/2008/03/17/understanding-redistribution-part-iii/

R1:

```
!
! RIP is the edge routing domain, redistribute b/w EIGRP and RIP
! on R6. There should be no routing loops with just single point
! of redistribution.
!
router eigrp 200
 redistribute rip metric 10000 1000 100 1 1500
!
router rip
 version 2
 redistribute eigrp 200 metric 1
```

R3:

```
!
! Redistribute b/w OSPF and EIGRP to obtain full reachability through
! the network.
!
router eigrp 200
 redistribute ospf 1 metric 10000 1000 100 1 1500
!
router ospf 1
 redistribute eigrp 200 subnets
```


R4:

```
!
! Summary /16 prefixes
!
interface FastEthernet0/0
 ip summary-address rip 162.1.0.0 255.255.0.0
 ip summary-address rip 150.1.0.0 255.255.0.0
!
! Summarize RIP networks into OSPF
!
router ospf 1
 redistribute connected subnets route-map CONNECTED->OSPF
 redistribute rip subnets
 summary-address 30.0.0.0 255.252.0.0
 summary-address 31.0.0.0 255.252.0.0
!
router rip
 version 2
 redistribute ospf 1 metric 1
 distribute-list prefix DENY_OSPF_SUMMARY out FastEthernet0/0
!
```

Pitfall

RIP prefixes are summarized into OSPF and R4 installs discard routes to Null0 for these summaries. When OSPF information is redistributed into RIP, it will also include these summaries, which is undesirable since it may result in suboptimal routing or routing loops. Therefore, we filter the summary prefixes when redistributing OSPF into RIP.

```
ip prefix-list DENY_OSPF_SUMMARY seq 5 deny 30.0.0.0/14
ip prefix-list DENY_OSPF_SUMMARY seq 10 deny 31.0.0.0/14
ip prefix-list DENY_OSPF_SUMMARY seq 15 permit 0.0.0.0/0 le 32

! Add interfaces not natively advertised into OSPF
!
route-map CONNECTED_TO_OSPF permit 10
 match interface Serial0/1/0 FastEthernet0/0

!
! Configure floating static routes for networks directly connected to
! R5 and redistribute it into OSPF
!
ip route 150.1.5.5 255.255.255.255 162.1.45.5 111
ip route 162.1.5.0 255.255.255.0 162.1.45.5 111
ip route 162.1.55.0 255.255.255.0 162.1.45.5 111
!
router ospf 1
 redistribute static subnets
```

R5:

```
ip route 0.0.0.0 0.0.0.0 162.1.45.4 111
```


Task 2.2 Breakdown

Static default routes are a very simple and effective way to replace more specific routing information when it is lost. A default route that is only installed in the IP routing table when another route (either dynamically learned or statically configured) is lost is called a *floating static* route.

A floating static route is a route with the same longest match as another route in the IP routing table, but which has a higher administrative distance. Therefore, the floating route will not get installed unless the primary route with the lower administrative distance leaves the IP routing table.

In the above scenario, R5 is learning a default route from R3 via OSPF. OSPF routes have an administrative distance of 110. There has also been a static default route configured on R5 pointing to R4 over the serial link with an administrative distance on 111. Unless the route with the lower administrative distance (the OSPF route with AD of 110) leaves the IP routing table, the static route will not get installed. This case will occur when R5 loses its connection to the Frame Relay cloud. Therefore, the above configured default route is a simple yet effective backup solution for R5.

In order to maintain full reachability back to R5, R4 has configured three static routes pointing to the directly connected networks of R4. As these routes have an administrative distance of 111 (higher than OSPF), they will not be installed in the routing table unless R4 loses the route through OSPF. Note that these routes must be redistributed into the OSPF domain to ensure that all other devices have a route when the Frame Relay circuit of R5 is down.

The other requirements involve routing information summarization. You need to inject two routes into OSPF for the prefixes learned via RIP on R4. This means external route summarization need to be configured in R4. Next, R4 should advertise summaries for your local networks to BB3, which translates into creating two /16 summary routes advertised into RIP.

Lastly, to obtain full reachability, you need to configure redistribution on R3, R4 and R1, since this operation connects RIP and OSPF routing domains across the EIGRP cloud. This should be enough to provide full reachability in IGP domains and keep redistribution loop free, as there are no redundant points of redistribution.

Task 2.2 Verification

Shutdown R5's Frame-Relay interface and verify routing table:

```
Rack1R5(config)#interface serial 0/0/0
Rack1R5(config-if)#shutdown


Rack1R5#show ip route static
S*   0.0.0.0/0 [111/0] via 162.1.45.4
```

Verify connectivity to R5's networks:

```
Rack1R3#ping 150.1.5.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.5.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 88/89/92 ms

Rack1R3#ping 162.1.55.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 162.1.55.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 88/89/92 ms

Rack1R5(config)#interface serial 0/0/0
Rack1R5(config-if)#no shutdown
```

Make sure OSPF domain sees only the summaries of BB3's prefixes:

```
Rack1R3#show ip route ospf  | inc ( 30\.| 31\.)
     31.0.0.0/14 is subnetted, 1 subnets
O E2    31.0.0.0 [110/20] via 162.1.0.4, 00:05:13, Serial1/0
     30.0.0.0/14 is subnetted, 1 subnets
O E2    30.0.0.0 [110/20] via 162.1.0.4, 00:05:13, Serial1/0
```


Make sure R4 announces the summary for the internal address space and the OSPF summaries are not re-advertised back into RIP domain.

```
Rack1R4#debug ip rip

RIP: sending v2 update to 224.0.0.9 via FastEthernet0/0 (204.12.1.4)
RIP: build update entries
      0.0.0.0/0 via 0.0.0.0, metric 1, tag 0
      30.0.0.0/14 via 0.0.0.0, metric 1, tag 0
      31.0.0.0/14 via 0.0.0.0, metric 1, tag 0
      150.1.0.0/16 via 0.0.0.0, metric 2, tag 0
      162.1.0.0/16 via 0.0.0.0, metric 2, tag 0
      192.10.1.0/24 via 0.0.0.0, metric 1, tag 0

BB3>show ip route rip
     54.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
R       54.1.1.0/24 [120/1] via 204.12.1.4, 00:00:15, FastEthernet0/0
R    162.1.0.0/16 [120/2] via 204.12.1.4, 00:00:15, FastEthernet0/0
R    192.10.1.0/24 [120/1] via 204.12.1.4, 00:00:15, FastEthernet0/0
     150.1.0.0/16 is variably subnetted, 3 subnets, 2 masks
R       150.1.0.0/20 [120/2] via 204.12.1.4, 00:00:15, FastEthernet0/0
```

Note that RIP on R4 sends back to BB3 a default route (originated in OSPF) and summary prefixes.

Now verify full internal connectivity using the TCL script below script:


```
foreach i {
150.1.1.1
162.1.13.1
192.10.1.1
150.1.2.2
162.1.0.2
162.1.27.2
162.1.38.3
150.1.3.3
162.1.0.3
162.1.3.3
162.1.13.3
162.1.45.4
150.1.4.4
162.1.0.4
204.12.1.4
162.1.45.5
162.1.55.5
150.1.5.5
162.1.5.5
162.1.0.5
54.1.1.6
150.1.6.6
162.1.6.6
192.10.1.6
150.1.7.7
162.1.27.7
162.1.88.8
162.1.38.8
150.1.8.8
162.1.8.8
162.1.88.8
} {puts [ exec "ping $i" ] }
```

Note that the above script does not include IP addresses of the links connecting SW2, SW3 and SW4 as those are not advertised into any IGP and are used for BGP peering.

Finally verify reachability to the backbone IGP networks using the TCL script below:


```
foreach i {
204.12.1.254
192.10.1.254
54.1.1.254
31.3.0.1
31.2.0.1
31.1.0.1
31.0.0.1
30.2.0.1
30.3.0.1
30.0.0.1
30.1.0.1
212.18.1.1
212.18.0.1
212.18.3.1
212.18.2.1
222.22.2.1
220.20.3.1BG
205.90.31.1
} {puts [ exec "ping $i" ] }
```


Task 2.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

BGP > BGP Remove Private AS
BGP > BGP Dampening
BGP > BGP Dampening with Route-Maps
BGP > BGP Communities – No Advertise

R3:

```
!
! Prevent the Private-AS number leaking
!
router bgp 300
 neighbor 162.1.0.4 remove-private-AS
 neighbor 162.1.13.1 remove-private-AS
```

R4:

```
!
! Tune dampening options
!
router bgp 100
 bgp dampening route-map DAMPENING
!
route-map DAMPENING permit 10
 set dampening 15 1000 3000 30
```

R5:

```
!
! Advertise new prefix into BGP
!
interface Loopback2
 ip address 162.1.15.5 255.255.255.0
!
! Ensure R4 does not advertise our local prefixes further
!
router bgp 500
 network 162.1.15.0 mask 255.255.255.0
 neighbor 150.1.4.4 send-community
 neighbor 150.1.4.4 route-map NO_ADVERTISE out
!
ip as-path access-list 1 permit ^$
!
route-map NO_ADVERTISE permit 10
 match as-path 1
 set community no-advertise
route-map NO_ADVERTISE permit 1000
```


SW1:

```
interface Loopback1
 ip address 162.1.7.7 255.255.255.0
!
router bgp 65001
 network 162.1.7.0 mask 255.255.255.0
```

SW2:

```
interface Loopback1
 ip address 162.1.18.8 255.255.255.0
!
router bgp 65002
 network 162.1.18.0 mask 255.255.255.0
```


Task 2.3 Breakdown

Applying for a public BGP AS number requires the justification of the need for the AS number. For networks that do not have their own block of address space, this may not be possible. For this reason, the top 1024 addresses in the BGP AS range are marked as private.

Suppose that your network has multiple connections to the same Internet Service Provider. Due to complex routing policy, you want to run BGP with this upstream provider. However, as this provider is your only connection to the Internet, you are using their address space, and do not have your own provider independent block. In the case the provider can assign you a locally significant AS number in the range of 64512 – 65535. However, as these AS numbers are not valid on the Internet, they must be removed from the AS-Path of routes you are originating when the provider passes them upstream.

This is accomplished by adding the **`remove-private-as`** keyword on to the neighbor statement of the upstream connection. In order for the private AS number to be removed, it must be the only AS in the path. In other words, the private AS must be directly connected to the AS that is trying to remove it.

The route filtering requirement in this task dictate that R4 should not pass the prefixes learned from R5 any further. Generally, by setting the well known community attributes of no-export, no-advertise, or local-as, how a route is processed by an upstream neighbor can be controlled downstream. By setting the community value to one of the aforementioned, R4 will not advertise the route on. Recall the well-known communities:

Well Known Community	Behavior
Internet	All routes belong to this community by default. The Internet community has no special behavior.
No-advertise	Do not advertise to *any* BGP neighbor.
No-export	Do not advertise to any *EBGP* neighbor.
Local-AS	Do not advertise outside of sub-AS. This is a special case of no-export for use inside of a confederation.

Thus, by letting R5 set the community value of no-advertise, we instruct R4 to hold R5's prefixes but don't advertise them any further. Notice the use of the AS-PATH access-list matchint ^$, which selects only the prefixes originated locally in R5's AS.

The scenario further requires R4 to apply route flap dampening. BGP route flap dampening (damping) is the process of suppressing consistently unstable routes from being used or advertised to BGP neighbors. Dampening is (and must be) used to minimize the amount of route recalculation performed in the global BGP table as a whole.

Command Syntax:

```
bgp dampening [half-life reuse suppress max-suppress-time]
```

To understand dampening, the following terms must first be defined:

Penalty: Every time a route flaps, a penalty value of 1000 is added to the current penalty. All prefixes start with a penalty of zero.

Half-life: Configurable time it takes the penalty value to reduce by half. Defaults to 15 minutes.

Suppress Limit: Threshold at which a route is suppressed if the penalty exceeds. Defaults to 2000.

Reuse Limit: Threshold at which a suppressed route is unsuppressed if the penalty drops below. Defaults to 750.

Max Suppress: Maximum time a route can be suppressed if it has been stable. Defaults to four times the half-life value.

Each time a route flaps (leaves the BGP table and reappears), it is assigned a penalty of 1000. As soon as this occurs, the penalty of the route starts to decay based on the half-life timer. As the penalty increases, so does the rate of decay. For example, after a single flap, it will take 15 minutes for a prefix to reduce its penalty to 500.

Once the penalty of a prefix exceeds the suppress limit, the prefix is suppressed. A suppressed prefix cannot be used locally or advertised to any BGP peer. Once the penalty decay has resulted in the penalty decreasing below the reuse limit, the prefix is unsuppressed.

Lastly, the max-suppress timer dictates the maximum amount of time a prefix can be suppressed if it has been stable. This value is useful if a number of flaps have occurred in a short period of time, after which the route has been stable.

To enable BGP route flap dampening, simply enter the command **`bgp dampening`** under the BGP process. It can also be configured with a route-map as shown here, which allows the potential for more granularity, rather than applying to all networks.

Here, we are given the values for reuse, suppress, and max-suppress time. We are not given a value for half life, but it needs to be less than the default of 30 minutes. It needs to be less than 19. Here, we have chosen 15. For the math behind this, see the command reference for the equation:

```
Max penalty = reuse-limit *2^(maximum suppress time/half
time)
```

🕮 Deep Dive

- What negative effects may community-based signaling have on best-path selection?
- Why BGP dampening could actually work against network stability?

Task 2.3 Verification

Check R5's prefix in R4's BGP table:

```
Rack1R4#show ip bgp | include 150.1.9.0
 *> 150.1.9.0/24     162.1.0.3                              0 300 i
```

Verify that R5 actually sends communities to R4:

```
Rack1R5#show ip bgp neighbors 150.1.4.4 | include Comm
  Community attribute sent to this neighbor
```

Verify that R4 does not advertise R5 prefix to any peers:

```
Rack1R4#show ip bgp 162.1.15.0
BGP routing table entry for 162.1.15.0/24, version 18
Paths: (1 available, best #1, table Default-IP-Routing-Table, not
advertised to any peer)
Flag: 0x8A0
  Not advertised to any peer
  500
    150.1.5.5 (metric 65) from 150.1.5.5 (150.1.5.5)
      Origin IGP, metric 0, localpref 100, valid, external, best
      Community: no-advertise
      Dampinfo: penalty 475, flapped 1 times in 00:01:06
```

Verify the dampening parameters:

```
Rack1R4#show ip bgp dampening parameters
 dampening 15 1000 3000 30 (route-map DAMPENING 10)
  Half-life time      : 15 mins       Decay Time       : 370 secs
  Max suppress penalty:  4000         Max suppress time: 30 mins
  Suppress penalty    :  3000         Reuse penalty    : 1000
```


To verify how dampening works, first issue the **debug ip bgp updates** command on R5. Next do shutdown,and no shutdown four times or more at interface Loopback2. Keep a little time between shutdown/no shutdown long enough, for BGP to send update (watch debug output for control).

```
Rack1R5#conf t
Rack1R5(config)#interface lo2
Rack1R5(config-if)#shutdown
BGP(0): route 162.1.15.0/24 down
BGP(0): no valid path for 162.1.15.0/24
BGP(0): nettable_walker 162.1.15.0/24 no best path
BGP(0): 150.1.4.4 send unreachable 162.1.15.0/24
BGP(0): 150.1.4.4 send UPDATE 162.1.15.0/24 - unreachable

Rack1R5(config-if)#no shutdown
BGP(0): route 162.1.15.0/24 up
BGP(0): route 162.1.15.0/24 up
BGP(0): nettable_walker 162.1.15.0/24 route sourced locally
BGP(0): 150.1.4.4 send UPDATE (format) 162.1.15.0/24, next 150.1.5.5,
metric 0, path
```

Next look on R4 for any dampened paths:

```
Rack1R4#show ip bgp dampening dampened-paths
BGP table version is 25, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          From             Reuse    Path
*d 162.1.15.0/24    150.1.5.5        00:05:49 500 i
```


Task 3.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

IPv6 > IPv6 Link-Local Addressing
IPv6 > IPv6 MP-BGP

R1:

```
!
! Configure IPv6 addressing and FR mappings
!
interface Serial0/0
 ipv6 address 2001:CC1E:1::1/64
 ipv6 address FE80::1 link-local
 frame-relay map ipv6 2001:CC1E:1::3 113 broadcast
 frame-relay map ipv6 FE80::3 113
!
ipv6 unicast-routing
!
! Configure BGP IPv6 peers, activate IPv6 sessions
!
router bgp 200
 neighbor 2001:CC1E:1::3 remote-as 300
!
 address-family ipv6
 neighbor 2001:CC1E:1::3 activate
```

R2:

```
interface Serial0/0.1 point-to-point
 ipv6 address FEC0:234::2/64
 ipv6 address FE80::2 link-local
!
ipv6 unicast-routing
!
router bgp 300
 neighbor FEC0:234::3 remote-as 300
!
 address-family ipv6
 neighbor FEC0:234::3 activate
```


R3:

```
ipv6 unicast-routing
!
interface Serial1/0
 ipv6 address FEC0:234::3/64
 ipv6 address FE80::3 link-local
 frame-relay map ipv6 FEC0:234::2 302 broadcast
 frame-relay map ipv6 FEC0:234::4 304 broadcast
 frame-relay map ipv6 FE80::2 302
 frame-relay map ipv6 FE80::4 304
!
interface Serial1/1
 ipv6 address 2001:CC1E:1::3/64
 frame-relay map ipv6 2001:CC1E:1::1 311 broadcast
 ipv6 address FE80::3 link-local
 frame-relay map ipv6 FE80::1 311
!
router bgp 300
 neighbor 2001:CC1E:1::1 remote-as 200
 neighbor FEC0:234::2 remote-as 300
 neighbor FEC0:234::4 remote-as 100
!
 address-family ipv6
  neighbor 2001:CC1E:1::1 activate
  neighbor FEC0:234::2 activate
  neighbor FEC0:234::4 activate
```

R4:

```
interface Serial0/0/0
 ipv6 address FEC0:234::4/64
 ipv6 address FE80::4 link-local
 frame-relay map ipv6 FEC0:234::2 403
 frame-relay map ipv6 FEC0:234::3 403 broadcast
 frame-relay map ipv6 FE80::2 403
 frame-relay map ipv6 FE80::3 403
!
ipv6 unicast-routing
!
router bgp 100
 neighbor FEC0:234::3 remote-as 300
!
 address-family ipv6
  neighbor FEC0:234::3 activate
```


Task 3.1 Breakdown

Frame Relay is a non-broadcast multi-access (NBMA) media. This implies that layer 3 to layer 2 resolution is required for multipoint configurations. As of current Cisco IOS releases, Inverse Neighbor Discovery is not supported and probably will not, since it require delicate multcast address handling, which is far from being optimal in Frame-Relay. Therefore, static layer 3 to layer 2 resolution must be configured with the **frame-relay map ipv6** statement. The host portion of the configured site-local networks can be determined by issuing either the **show ipv6 interface** command or the **show ipv6 interface brief** command.

Note

The output from the commands below is not necessarily based on this particular scenario and may not match the command output on your rack.

```
Rack1R1#show ipv6 interface s0/0
Serial0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::2D0:58FF:FE6E:B720
  Global unicast address(es):
    FEC0::13:2D0:58FF:FE6E:B720, subnet is FEC0:0:0:13::/64
<output omitted>

Rack1R3#show frame-relay map
Serial1/1 (up): ipv6 FEC0::13:2D0:58FF:FE6E:B720 dlci 311
(0x137,0x4C70), static,
              broadcast,
              CISCO, status defined, active
<output omitted>
```

The task solution configuration dictates how to configure Multiprotocol BGP for IPv6. The first step is to issue the BGP **neighbor** command, followed by the destination peer's IPv6 address and AS number. Next, enable IPv6 unicast processing for the neighbor by issuing the **address-family ipv6** command under the BGP process, followed by the **neighbor [ipv6 address] activate** command.


```
Rack1R3#show bgp ipv6 summary
BGP router identifier 162.1.13.3, local AS number 300
BGP table version is 2, main routing table version 2
1 network entries using 133 bytes of memory
1 path entries using 72 bytes of memory
1 BGP path attribute entries using 60 bytes of memory
1 BGP AS-PATH entries using 24 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 289 total bytes of memory
BGP activity 1/0 prefixes, 1/0 paths, scan interval 60 secs

Neighbor  V  AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down State/PfxRcd
FEC0::13:2D0:58FF:FE6E:B720
          4  200     24      23        2    0    0 00:19:52        1
```

To check the status of the MBGP peering, issue the **show bgp ipv6 summary** command. In the above output, R3 has formed a MBGP peering relationship with R1 using the destination address FEC0::13:2D0:58FF:FE6E:B720, and is learning one IPv6 prefix.

```
Rack1R3#show bgp ipv6
BGP table version is 2, local router ID is 162.1.13.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 2001:CC1E:1:1::/64
                    FEC0::13:2D0:58FF:FE6E:B720
                                             0             0 200 i
```

Note that in the above output, the prefix 2001:CC1E:1:1:/64 has been learned via BGP per the **show bgp ipv6** output, and has an associated next-hop value of FEC0::13:2D0:58FF:FE6E:B720.

```
Rack1R3#show ipv6 route FEC0::13:2D0:58FF:FE6E:B720
IPv6 Routing Table - 5 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
    O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
C   FEC0:0:0:13::/64 [0/0]
     via ::, Serial1/1
```

When a recursive lookup is performed on this next-hop value, per the above **show ipv6 route FEC0::13:2D0:58FF:FE6E:B720** output, the outgoing interface is seen to be Serial1/1, which is a multipoint interface running Frame Relay. However, when traffic is encapsulated on the interface the layer 2 address is resolved per the link-local address of the next-hop interface, not the global unicast address. This can be seen by the below **show ipv6 route bgp** output.


```
Rack1R3#show ipv6 route bgp
IPv6 Routing Table - 5 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
    O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
B   2001:CC1E:1:1::/64 [20/0]
     via FE80::2D0:58FF:FE6E:B720, Serial1/1
```

This implies that layer 3 to layer 2 resolution via the **`frame-relay map ipv6`** command must be configured for the remote link-local address FE80::2D0:58FF:FE6E:B720.

```
Rack1R1#show ipv6 int brief
Ethernet0/0                [up/up]
    FE80::230:19FF:FE69:81A0
    2001:CC1E:1:1:230:19FF:FE69:81A0
Serial0/0                  [up/up]
    FE80::2D0:58FF:FE6E:B720
    FEC0::13:2D0:58FF:FE6E:B720
<output omitted>
```

Quick Note
Global unicast address

Quick Note
Link-local address

```
Rack1R3#debug ipv6 packet
IPv6 unicast packet debugging is on
Rack1R3#debug frame-relay packet
Frame Relay packet debugging is on
Rack1R3#ping
Protocol [ip]: ipv6
Target IPv6 address: 2001:CC1E:1:1:230:19FF:FE69:81A0
Repeat count [5]: 1
Datagram size [100]:
Timeout in seconds [2]:
Extended commands? [no]:
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 2001:CC1E:1:1:230:19FF:FE69:81A0,
timeout is 2 seconds:

IPv6: SAS picked source FEC0::13:201:96FF:FE63:96C0 for
2001:CC1E:1:1:230:19FF:FE69:81A0
IPv6: nexthop FE80::2D0:58FF:FE6E:B720,
IPV6: source FEC0::13:201:96FF:FE63:96C0 (local)
      dest 2001:CC1E:1:1:230:19FF:FE69:81A0 (Serial1/1)
      traffic class 0, flow 0x0, len 100+0, prot 58, hops 64,
originating
Serial1/1:Encaps failed--no map entry link 79(IPV6)
IPv6: Encapsulation failed.
Success rate is 0 percent (0/1)
```

Quick Note
Encapsulation fails because R3 does not have a layer 3 to layer 2 mapping for the next-hop address FE80::2D0:58FF:FE6E:B720s

```
Rack1R3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#interface s1/1
Rack1R3(config-if)#frame-relay map ipv6 FE80::2D0:58FF:FE6E:B720 311
Rack1R3(config-if)#do ping 2001:CC1E:1:1:230:19FF:FE69:81A0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:1:230:19FF:FE69:81A0,
timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/17/20 ms
```

🕮 Deep Dive

- Could you activate IPv6 address-family for IPv4 neighbor in BGP?

Task 3.1 Verification

Note that link-local IPv6 addresses are configured and mapped explicitly. This is needed to configure IPv6 BGP later.

```
Rack1R4#show frame-relay map
<output omitted>
Serial0/0/0 (up): ipv6 FE80::2 dlci 403(0x193,0x6430), static,
              CISCO, status defined, active
Serial0/0/0 (up): ipv6 FE80::3 dlci 403(0x193,0x6430), static,
              CISCO, status defined, active
...
Serial0/0/0 (up): ipv6 FEC0:234::2 dlci 403(0x193,0x6430), static,
              CISCO, status defined, active
Serial0/0/0 (up): ipv6 FEC0:234::3 dlci 403(0x193,0x6430), static,
              broadcast,
              CISCO, status defined, active
<output omitted>

Rack1R4#ping fec0:234::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FEC0:234::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 136/139/140
ms

Rack1R4#ping fec0:234::3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FEC0:234::3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 60/60/60 ms
```


Note that you need to enable ipv6 unicast routing on R3 in order to ping spoke from spoke.

Verify IPv6 BGP peering status:

```
Rack1R3#show bgp ipv6 unicast summary  | begin Neighbor
Neighbor        V AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down State/PfxRcd
2001:CC1E:1::1
                4 200    5       6        7    0    0 00:01:19        0
FEC0:234::2     4 300   14      15        7    0    0 00:09:31        0
FEC0:234::4     4 100   15      13        7    0    0 00:03:22        0
```


Task 3.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

IPv6 > IPv6 MP-BGP

R1:

```
!
! Advertise IPv6 prefixes into BGP on R1,R2 and R3
!
router bgp 200
 !
 address-family ipv6
 network 2001:CC1E:1:1::/64
```

R2:

```
router bgp 300
!
 address-family ipv6
 network 2001:CC1E:1:2::/64
```

R3:

```
router bgp 300
 !
 address-family ipv6
 network 2001:CC1E:1::/64
 network 2001:CC1E:1:3::/64
 network FEC0:234::/64
!
! Apply summarization on R3 and unsuppress prefixes
! sent to R1
!
 neighbor 2001:CC1E:1::1 unsuppress-map UNSUPPRESS
 neighbor FEC0:234::2 unsuppress-map UNSUPPRESS
 aggregate-address 2001:CC1E:1::/62 summary-only
!
route-map UNSUPPRESS permit 10
```

R4:

```
!
! Advertise IPv6 prefix into BGP on R4
!
router bgp 100
 !
 address-family ipv6
  network 2001:204:12:1::/64
```

Task 3.2 Breakdown

Aside from handling the next-hops over NBMA media, there isn't much difference in working with IPv6 addresses in MP-BGP. The same commands are used to advertise the network prefixes and aggregate reachability information. In this scenario, we create an aggregate address for the prefixes 2001:CC1E:1:1::/64, 2001:CC1E:1:2::/64 and 2001:CC1E:1:3::/64. If you break the non-matching octet in binary form you will get

```
0000 0000 0000 0001
0000 0000 0000 0010
0000 0000 0000 0011
```

And effectively you need to shift the prefix left by two bits to mask the mismatching components. The resulting summary prefix would be 2001:CC1E:1::/62. Unsuppress map is used to make sure that R1 and R2 will receive the more specific prefixes as well.

Task 3.2 Verification

Verify that R4 receives only the summarized prefixes from R3:

```
Rack1R4#show bgp ipv6 unicast neighbors FEC0:234::3 routes  | begin Net
   Network          Next Hop            Metric LocPrf Weight Path
*> 2001:CC1E:1::/62 FEC0:234::3              0             0 300 i
*> FEC0:234::/64    FEC0:234::3              0             0 300 i

Total number of prefixes 2
```

Verify that R2 receives all prefixes (including the summary):

```
Rack1R2#show bgp ipv6 unicast neighbors FEC0:234::3 routes | begin Net
   Network          Next Hop            Metric LocPrf Weight Path
<output omitted>
*>i2001:CC1E:1::/64 FEC0:234::3              0    100      0 i
*>i2001:CC1E:1::/62 FEC0:234::3              0    100      0 i
*>i2001:CC1E:1:1::/64
                    2001:CC1E:1::1           0    100      0 200 i
*>i2001:CC1E:1:3::/64
                    FEC0:234::3              0    100      0 i
*>iFEC0:234::/64    FEC0:234::3              0    100      0 i
```


Task 5.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

IP Multicast > AutoRP – Multiple Candidate RPs
IP Multicast > AutoRP – Filtering Candidate RPs
IP Multicast > AutoRP – Filtering AutoRP Messages

R1:
```
interface Loopback0
 ip pim sparse-dense-mode
!
ip pim send-rp-discovery Loopback0 scope 16
ip pim rp-announce-filter rp-list 25 group-list 26
ip pim rp-announce-filter rp-list 50 group-list 51
!
access-list 25 permit 150.1.3.3
access-list 26 permit 239.0.0.0 0.255.255.255
access-list 50 permit 150.1.5.5
access-list 51 deny 224.0.0.0 1.255.255.255
access-list 51 deny 239.0.0.0 0.255.255.255
access-list 51 permit 224.0.0.0 15.255.255.255
```

R3:
```
interface Loopback0
 ip pim sparse-dense-mode
!
ip pim send-rp-announce Loopback0 scope 16 group-list 50
!
access-list 50 permit 239.0.0.0 0.255.255.255
```

R5:
```
interface Loopback0
 ip pim sparse-dense-mode
!
ip pim send-rp-announce Loopback0 scope 16 group-list 50
!
access-list 50 permit 226.0.0.0 1.255.255.255
access-list 50 permit 228.0.0.0 3.255.255.255
access-list 50 permit 232.0.0.0 3.255.255.255
access-list 50 permit 236.0.0.0 1.255.255.255
access-list 50 permit 238.0.0.0 0.255.255.255
```


Task 5.1 Breakdown

The requirement is given to have R1 map only specific multicast groups to R3 and R5. This will require the use of the **ip pim rp-announce-filter** command on the mapping agent (MA). The rp-announce filter will need to match the send-rp-announce filter used by the cRPs. If the groups requested by the RP do not match the mapping agent's filters, the cRPs requests that do not match will be discarded.

Example: If the cRP asks for 228.0.0.0/8 and 229.0.0.0/8, but the MA is allowing only 228.0.0.0/8, then the MA will filter the 229.0.0.0/8 and the cRP will be the RP for just 228/8. If the cRP asks for say 224.0.0.0/4 (all multicast groups), but the MA is only allowing 228.0.0.0/8, then the cRP's 224.0.0.0/4 announcement will be filtered by the MA and the cRP will not be the RP for any groups.

The logic of the access-list 51 used is as follows. The first step in finding out how to match all these addresses is to write them out in binary:

```
226 11100010
227 11100011
228 11100100
229 11100101
230 11100110
231 11100111
232 11101000
233 11101001
234 11101010
235 11101011
236 11101100
237 11101101
238 11101110
```

From the above addresses, it's evident that it's very close to a complete range, but is missing the 224, 225, and 239. If these numbers were included, then the range would be a contiguous block of 16 addresses (224 to 239). This address block would be matched as:

```
access-list 51 permit 224.0.0.0 15.255.255.255
```

However, now it overlaps 224, 225, and 239. These three can be removed with deny statements:

```
access-list 51 deny 224.0.0.0 0.255.255.255
access-list 51 deny 225.0.0.0 0.255.255.255
access-list 51 deny 239.0.0.0 0.255.255.255
access-list 51 permit 224.0.0.0 15.255.255.255
```

The total list is four lines now. Since we want the minimum number of lines, let's try to consolidate some of the deny statements. To do this we write out the three denied addresses in binary:


```
224 - 11100000
225 - 11100001
239 - 11101111
```

From the above address space, it is easily seen that 224 and 225 can be consolidated in one line (only the least significant bit is different), but the 239 cannot. Therefore these can be rewritten as:

```
     224 - 11100000
     225 - 11100001
Wildcard - 00000001

     239 - 11101111
Wildcard - 00000000
```

These groupings result in the following list:

```
access-list 51 deny 224.0.0.0 1.255.255.255
access-list 51 deny 239.0.0.0 0.255.255.255
access-list 51 permit 224.0.0.0 15.255.255.255
```

Now we have three lines. This may be the least amount of lines, but let's use just permit statements to be sure. Remember the addresses are as follows:

```
226 11100010
227 11100011
228 11100100
229 11100101
230 11100110
231 11100111
232 11101000
233 11101001
234 11101010
235 11101011
236 11101100
237 11101101
238 11101110
```


Now let's group them together in ranges that can be checked without overlap:

```
226 11100010
227 11100011

228 11100100
229 11100101
230 11100110
231 11100111

232 11101000
233 11101001
234 11101010
235 11101011

236 11101100
237 11101101

238 11101110
```


This gives us five permit statements.

```
226 11100010
227 11100011
230 11100110
231 11100111

228 11100100
229 11100101
236 11101100
237 11101101

232 11101000
233 11101001
234 11101010
235 11101011

238 11101110
```

This gives us four. Better, but still not three. There's no way to get under four statements just by using permits. Therefore the correct answer for this section should be:

```
access-list 51 deny 224.0.0.0 1.255.255.255
access-list 51 deny 239.0.0.0 0.255.255.255
access-list 51 permit 224.0.0.0 15.255.255.255
```

🕮 Deep Dive

- Why can’t you use “deny” statements to filter unneeded groups out of specific RP?

Task 5.1 Verification

Below is the output from the **debug ip pim auto-rp** command on R1, with R5 configured to request all groups (no group-list on R5's **ip pim send-rp-announce** command), and R1 is configured with the **ip pim rp-announce-filter** shown above.

```
Rack1R1#debug ip pim auto-rp
PIM Auto-RP debugging is on
Rack1R1#
 Auto-RP: Received RP-announce, from 150.1.5.5, RP_cnt 1, ht 181
 Auto-RP: Filtered 224.0.0.0/4 for RP 150.1.5.5
Rack1R1#
```

As we can see, R5 request all multicast groups. This request does not match R1's filters and in turn was discarded (filtered). Now, we will add the *group-list* option to R5's **ip pim send-rp-announce** command that matches R1's **ip pim rp-announce-filter** *group-list*.

```
Rack1R1#
 Auto-RP: Received RP-announce, from 150.1.5.5, RP_cnt 1, ht 181
 Auto-RP: Update (226.0.0.0/7, RP:150.1.5.5), PIMv2 v1
 Auto-RP: Update (228.0.0.0/6, RP:150.1.5.5), PIMv2 v1
 Auto-RP: Update (232.0.0.0/6, RP:150.1.5.5), PIMv2 v1
 Auto-RP: Update (236.0.0.0/7, RP:150.1.5.5), PIMv2 v1
 Auto-RP: Update (238.0.0.0/8, RP:150.1.5.5), PIMv2 v1
```



```
Rack1R1#show ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP-mapping agent (Loopback0)

Group(s) 226.0.0.0/7
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.5.5 (?), via Auto-RP
         Uptime: 00:03:13, expires: 00:02:47
Group(s) 228.0.0.0/6
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.5.5 (?), via Auto-RP
         Uptime: 00:03:13, expires: 00:02:45
Group(s) 232.0.0.0/6
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.5.5 (?), via Auto-RP
         Uptime: 00:03:14, expires: 00:02:44
Group(s) 236.0.0.0/7
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.5.5 (?), via Auto-RP
         Uptime: 00:03:14, expires: 00:02:45
Group(s) 238.0.0.0/8
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.5.5 (?), via Auto-RP
         Uptime: 00:03:15, expires: 00:02:44
Group(s) 239.0.0.0/8
  RP 150.1.3.3 (?), v2v1
    Info source: 150.1.3.3 (?), elected via Auto-RP
         Uptime: 00:01:30, expires: 00:02:30
```


Task 5.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

IP Multicast > Multicast Boundary
IP Multicast > Stub Multicast Routing & IGMP Helper
IP Multicast > PIM Sparse Mode

R1:

```
!
! Enable R1 to send multicast traffic to BB2 but block PIM
! message exchange.
!
interface FastEthernet0/0
 ip pim neighbor-filter 75
!
access-list 75 deny   192.10.1.254
access-list 75 permit any
```

R3:

```
!
! Set up multicast boundary on R3's connection SW2
!
interface FastEthernet0/0
 ip multicast boundary 51
!
access-list 51 deny   239.0.0.0 0.255.255.255
access-list 51 permit 224.0.0.0 15.255.255.255
```

R1, R2, R3, R5, SW2:

```
!
! Stop switching to SPTs for the adming. Mcast group range
!
ip pim spt-threshold infinity group-list 52
!
access-list 52 permit 239.0.0.0 0.255.255.255
```


Task 5.2 Breakdown

To restrict PIM neighbor relationship, the **ip pim neighbor-filter** interface command was used for this section. This will not allow BB2 to become a PIM neighbor with R1, but will still allow clients on the FastEthernet segment to join any multicast group as this command only affects PIM neighbor relationships and not IGMP. This configuration can be verified by using the **show ip pim neighbors** command.

Although other methods can be used the control multicast traffic, the preferred method to block multicast traffic flow at an interface to use the **ip multicast-boundary** interface command. The command requires an access-list that defines what groups are deny or permitted out the interface.

With multicasting, there normally is more than one receiver and in turn there can be more than one path the packets take through the network to reach the various receivers. These paths are commonly referred to as branches on the multicast tree. There are two types of multicast trees that can be formed in regards to this section's configuration. The default tree type is a source based tree. A source based tree is rooted at the source of the multicast stream. The tree is built using the least cost path between the source and destination or destinations. This is sometimes referred to a shortest path tree. The second type of tree, the one that this configuration will actually use, is called a shared tree. With a shared tree all multicast packets are sent to the RP and then down to the receivers.

It is interesting to note how routers perform the RPF check in regards to the two different types of trees. With a source based tree, the RPF check is done against the source of the multicast packets. With a shared tree, the RPF check is done against the RP and not against the source of the multicast stream. Using a shared tree, as opposed to the default of a source based tree, could possibly be used to workaround an issue with an RPF failure where static mroutes could not be used and the unicast routing could also not be easily changed.

For this task, the spt threshold was modified. By default, after traffic is received, there is a switchover from the shared tree via the RP to the shortest path tree. By setting the threshold to infinity, there will never be a switchover to the shortest path tree. Alternatively, another solution would be to configure bidirectional PIM for these groups, which would require configuring **ip pim bidir-enable** on the multicast routers, and configuring R3's advertisement with the bidir keyword.

🕮 Deep Dive

- What is main use of the PIM neighbor-filter command and what other option could replace it?
- Will multicast boundary affect IGMP joins?
- Where should you put the PIM SPT threshold setting?

Task 5.2 Verification

Try adding R6's interface Fa0/0 IP address to access-list 75:

```
Rack1R1#show ip access-lists 75
Standard IP access list 75
    10 deny   192.10.1.254
    15 deny   192.10.1.6 (3 matches)
    20 permit any (3 matches)
```

Next issue the following debug commands:

```
Rack1R1(config)#ip access-list standard 75
Rack1R1(config-std-nacl)#no 15

Rack1R1#debug ip pim
Rack1R1#debug ip pim hello

PIM(0): Send periodic v2 Hello on FastEthernet0/0
PIM(0): v2, PIM neighbor 192.10.1.6 explicitly denied on
FastEthernet0/0
```

Verify the multicast boundary:

```
Rack1R3#show ip multicast interface e0/0
Ethernet0/0 is up, line protocol is up
  Internet address is 162.1.38.3/24
  Multicast routing: enabled
  Multicast switching: fast
  Multicast packets in/out: 12771/43
                       51
  Multicast TTL threshold: 0
  Multicast Tagswitching: disabled

Rack1R3#show ip access-lists 51
Standard IP access list 51
    10 deny   239.0.0.0, wildcard bits 0.255.255.255
    20 permit 224.0.0.0, wildcard bits 15.255.255.255 (24 matches)
```


Join interface Fa0/0 at R5 to group 239.5.5.5, and then ping 239.5.5.5 from R1. Now check the multicast routing table at R5:

```
Rack1R5#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.5.5.5), 00:01:08/00:02:25, RP 150.1.3.3, flags: SCL
  Incoming interface: Serial0/0/0, RPF nbr 162.1.0.3
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 00:01:08/00:02:25
<output omitted>
```

Note that the 'J' flag is not set for group 239.5.5.5 and there are no SPT entries in multicast routing table of R5. Next, join e0/0 of R5 to group 226.5.5.5 and ping the group from R1 again. Now, verify the multicast routing table at R5 to compare outputs:

```
Rack1R5#show ip mroute 226.5.5.5
IP Multicast Routing Table

<output omitted>

(*, 226.5.5.5), 00:00:20/stopped, RP 150.1.5.5, flags: SJCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 00:00:20/00:02:39

(150.1.1.1, 226.5.5.5), 00:00:13/00:02:52, flags: T
  Incoming interface: Serial0/0/0, RPF nbr 162.1.0.3
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse-Dense, 00:00:13/00:02:46
<output omitted>
```

Note the 'J' flag and SPT entry with the 'T' flag set.

Task 6.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

Security > Traffic Filtering with Reflexive Access-Lists

R4:

```
!
! Allow returning routing traffic in the inbound access-list
!
ip access-list extended IN_ACL
 permit icmp any any echo-reply
 permit tcp any eq telnet any established
 permit tcp any any eq bgp
 permit tcp any eq bgp any
 permit udp any any eq rip
 evaluate MY_REFLECT
!
ip access-list extended OUT_ACL
 permit tcp any any reflect MY_REFLECT
 permit udp any any reflect MY_REFLECT
 permit icmp any any reflect MY_REFLECT
!
! Apply the reflexive access-lists
!
interface FastEthernet0/0
 ip access-group IN_ACL in
 ip access-group OUT_ACL out
```


Task 6.1 Breakdown

This task requires that ping and telnet packets be permitted to return. Since traffic originated by the router itself will not be reflected, the following static entries are needed:

```
ip access-list extended IN_ACL
 permit icmp any any echo-reply
 permit tcp any eq telnet any established
```

In addition to the entries permitting IGPs and BGP session. These access-list entries will allow telnet and ICMP echo replies inbound, even if they were not reflected. A workaround for this can be to policy route all ICMP and telnet traffic originated by the router out a loopback interface. When this occurs, the traffic will be reflected and in turn can by evaluated. Here is an example:

```
interface FastEthernet0/0
 ip access-group IN_ACL in
 ip access-group OUT_ACL out
!
ip access-list extended IN_ACL
 evaluate MY_REFLECT
ip access-list extended OUT_ACL
 permit tcp any any reflect MY_REFLECT
 permit udp any any reflect MY_REFLECT
 permit icmp any any reflect MY_REFLECT
```



```
Rack1R4#ping 204.12.1.254

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

ICMP: dst (204.12.1.4) administratively prohibited unreachable sent to
204.12.1.254.
ICMP: dst (204.12.1.4) administratively prohibited unreachable sent to
204.12.1.254.
ICMP: dst (204.12.1.4) administratively prohibited unreachable sent to
204.12.1.254.
ICMP: dst (204.12.1.4) administratively prohibited unreachable sent to
204.12.1.254.
ICMP: dst (204.12.1.4) administratively prohibited unreachable sent to
204.12.1.254.
Success rate is 0 percent (0/5)
Rack1R4#
```

As we can see from the output of the **debug ip icmp** command, the echo replies were not allowed to return. R4 denied the replies and in turn sent an administratively prohibited message to the source of the reply (BB3). Of course in a real network, we normally do not want the router telling the source that a packet was denied by an access-list and disable these replies by using the **no ip unreachables** command under the interfaces.

The workaround will involve policy routing the ICMP echo requests and telnet packets out the loopback 0 interface.

```
route-map LOCAL_TRAFFIC permit 10
 match ip address ORIGINATED
 set interface Loopback0
!
ip access-list extended ORIGINATED
 permit icmp any any
 permit tcp any any eq telnet
```

This configuration will allow the packets originated by the router itself to be reflected.

```
Rack1R4#ping 204.12.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 204.12.1.254, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/6/9 ms
Rack1R4#
```


To further verify that the policy routing is working, below is the same ping with the **debug ip policy** command enabled.

```
Rack1R4#debug ip policy
Policy routing debugging is on
Rack1R4#ping 204.12.1.254 repeat 1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 204.12.1.254, timeout is 2 seconds:
!
Success rate is 100 percent (5/5), round-trip min/avg/max = 8/9/12 ms
Rack1R4#
IP: s=204.12.1.4 (local), d=204.12.1.254, len 100, policy match
IP: route map LOCAL_TRAFFIC, item 10, permit
IP: s=204.12.1.4 (local), d=204.12.1.254 (Loopback0), len 100, policy
routed
IP: local to Loopback0 204.12.1.254
Rack1R4#
```

Alternatively, this section could be completed using CBAC.

🕮 Deep Dive

- What is the purpose of the “established” keyword used in the access-lists?
- What CBAC feature corresponds to the “policy-based routing” trick used with reflexive access-lists?

Task 6.1 Verification

Verify that R4 can ping and telnet to BB3:

```
Rack1R4#ping 204.12.1.254

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

Rack1R4#telnet 204.12.1.254
Trying 204.12.1.254 ... Open

BB3>exit

[Connection to 204.12.1.254 closed by foreign host]
```



```
Rack1R4#show ip access-lists MY_REFLECT
Reflexive IP access list MY_REFLECT
     permit tcp host 204.12.1.254 eq telnet host 162.1.0.3 eq 44287 (11
matches) (time left 287)
     permit icmp host 204.12.1.254 host 162.1.0.3  (5 matches) (time
left 206)
```

Verify that RIP and BGP are OK:

```
Rack1R4#show ip route rip
     31.0.0.0/8 is variably subnetted, 5 subnets, 2 masks
R       31.3.0.0/16 [120/1] via 204.12.1.254, 00:00:25, FastEthernet0/0
R       31.2.0.0/16 [120/1] via 204.12.1.254, 00:00:25, FastEthernet0/0
R       31.1.0.0/16 [120/1] via 204.12.1.254, 00:00:25, FastEthernet0/0
R       31.0.0.0/16 [120/1] via 204.12.1.254, 00:00:25, FastEthernet0/0
<output omitted>

Rack1R4#show ip bgp summary | begin Neighbor
Neighbor    V AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down  State/PfxRcd
2001:204:12:1::254
            4 54    46      52       0    0    0 00:40:55 (NoNeg)
150.1.5.5   4 500  195     190      26    0    0 02:59:24        1
162.1.0.3   4 300  192     190      26    0    0 02:59:36        5
204.12.1.254
            4 54   191     191      26    0    0 02:59:18       10
FEC0:234::3 4 300   66      70      26    0    0 00:35:48        0
```

Verify that internal routers can still ping and telnet to BB3:

```
Rack1R3#ping 204.12.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 204.12.1.254, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 60/62/64 ms

Rack1R3#telnet 204.12.1.254
Trying 204.12.1.254 ... Open

BB3>exit

[Connection to 204.12.1.254 closed by foreign host]
```


And finally, verify that BB3 can not initiate connections to the internal routers:

```
BB3>show ip route 150.1.3.3
Routing entry for 150.1.0.0/20
  Known via "rip", distance 120, metric 2
  Redistributing via rip
  Last update from 204.12.1.4 on FastEthernet0, 00:00:13 ago
  Routing Descriptor Blocks:
  * 204.12.1.4, from 204.12.1.4, 00:00:13 ago, via FastEthernet0
      Route metric is 2, traffic share count is 1

BB3>telnet 150.1.3.3
Trying 150.1.3.3 ...
% Destination unreachable; gateway or host down
```


Task 6.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

Security > VLAN Filters for IP Traffic

SW2:

```
!
! Drop ICMP packets from the specific subnet
!
vlan access-map ICMP_FILTER 10
 action drop
 match ip address 100
!
! Allow all other traffic
!
vlan access-map ICMP_FILTER 20
 action forward

!
! Apply the VLAN filter
!
vlan filter ICMP_FILTER vlan-list 162
!
! Access-list matching traffic to drop
!
access-list 100 permit icmp 205.90.31.0 0.0.0.255 any echo
```


Task 6.2 Breakdown

A VLAN access-list, commonly referred to as a VACL, uses the same basic logic as a route-map. A VACL contains sequence numbers along with *match* and *action* criteria like a route-map with its *match* and *set* criteria. In a VACL, the match can be an IP or MAC access-list. The action will always be forward of drop. The action will either forward the traffic that is matched, or drop the traffic that is matched. If the **vlan access-map** sequence does not contain a match statement, all traffic will match.

This means that the logic of the access-list is important in that traffic that you want to deny will need to be permitted in your access-list. By permitting the traffic in your access-list, it will match and in turn the action will be taken. Of course, you could reverse the logic and match traffic that you want to permit and use the action of *forward* on it. Then create another **vlan access-map** sequence that just has an *action drop* and no match statement. Here is an example:

```
vlan access-map ICMP_FILTER 10
 action forward
 match ip address 100
!
vlan access-map ICMP_FILTER 20
 action drop
vlan filter ICMP_FILTER vlan-list 162
!
access-list 100 deny icmp 205.90.31.0 0.0.0.255 any echo
access-list 100 permit ip any any
```

Pitfall

Because VACLs are compiled to improve performance, changes made to an access-map will not take affect till the **vlan filter** is removed and reapplied. Be sure to remove and then reapply the **vlan filter** command whenever changes are made to the **vlan access-map** commands.

In the above configuration, the access-list logic is reversed. Here access-list 100 denies ICMP sourced from 205.90.31.0/24 and permits all other IP traffic. Since the ICMP traffic from 205.90.31.0/24 is not 'positive' in the access-list, it will not match and in turn not get forwarded. The ICMP traffic will match **vlan access-map** sequence number 20, since there is not a match statement. This will, in turn, cause the ICMP traffic to be dropped.

Although this configuration appears fine, it will cause problems. The problem is that access-list 100 does not permit ARP. The IP FastEthernet and ARP FastEthernet types are not the same. ARP will not match **vlan access-map** sequence number 10 but will match 20, which has a default action of drop. This configuration will appear to work for a few hours or until a reboot, assuming the devices have already created an ARP entry before the **vlan filter** was applied. Someone could leave the lab with the assumption that their configurations are correct, but once the devices are reloaded and need to ARP, everything that relies on ARP will stop working (routing protocols, ping, etc). Here is an example:

```
Rack1R1#show arp
Protocol  Address    Age (min)  Hardware Addr   Type   Interface In-
ternet  192.10.1.254      1   00e0.1ece.4a68  ARPA   FastEthernet0/0
Internet  192.10.1.1        -   00d0.586e.b720  ARPA   FastEthernet0/0
Rack1R1#ping 192.10.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.10.1.254, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

Now the ARP cache is cleared, so that R1 will need to ARP for BB2. Then, we try to ping from R1 (192.10.1.1) to BB2 (192.10.1.254) again.

```
Rack1R1#clear arp

Rack1R1#show arp
Protocol  Address    Age (min)  Hardware Addr   Type   Interface In-
ternet  192.10.1.1        -   00d0.586e.b720  ARPA   FastEthernet0/0

Rack1R1#ping 192.10.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.10.1.254, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)
Rack1R1#
```


As we can see this configuration broke ARP. To permit ARP, a MAC access-list will need to be created that matches the ARP FastEthernet type. A new **vlan filter** sequence will also need to be added. Do not forget to remove the **vlan filter** and reapply it.

```
mac access-list extended PERMIT_ARP
 permit any any 0x806 0x0
!
!
vlan access-map ICMP_FILTER 10
 action forward
 match ip address 100
vlan access-map ICMP_FILTER 15
 action forward
 match mac address PERMIT_ARP
vlan access-map ICMP_FILTER 20
 action drop
vlan filter ICMP_FILTER vlan-list 162
!
access-list 100 deny icmp 205.90.31.0 0.0.0.255 any echo
access-list 100 permit ip any any

Rack1SW2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1SW2(config)#no vlan filter ICMP_FILTER vlan-list 162
Rack1SW2(config)#vlan filter ICMP_FILTER vlan-list 162
Rack1SW2(config)#^Z
Rack1SW2#

Rack1R1#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 80 percent (4/5), round-trip min/avg/max = 1/3/4 ms
Rack1R1#
```

🕮 Deep Dive

- Can you combine interface-level access-lists along with VLAN filters?
- Which direction do VLAN filters work?

Task 6.2 Verification

Verify that the VLAN filter is applied:

```
Rack1SW1#show vlan filter
VLAN Map ICMP_FILTER is filtering VLANs:
  162
```

Verify that R1 can still ping addresses in the 205.90.31.0/24 network

```
Rack1R1#ping 205.90.31.1

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 = 4/5/8 ms
```

To verify that the filter works add another entry to access-list 100:

```
access-list 100 permit icmp 205.90.31.0 0.0.0.255 any echo-reply
```

And try ping again:

```
Rack1R1#ping 205.90.31.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 205.90.31.1, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)

Rack1R1#ping 192.10.1.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.10.1.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```


Task 6.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

Security > Traffic Filtering with Policy Based Routing

R6:

```
!
! Match ICMP packets
!
ip access-list extended ICMP
 permit icmp any any
!
! Match outgoing interface and packet sizes
!
route-map DROP
 match ip address ICMP
 match interface FastEthernet 0/0
 match length 100 200
!
! Redirect packets to Null0 interface
!
 set interface Null0
!
interface Serial 0/0/0.1
 ip policy route-map DROP
```

Task 6.3 Solution

The scenario requires matching based on packet size, which could be done using MQC features, Flexible Packet Matching and Policy Based Routing. Out of all this options we are only left with PBR. The route-map used to accomplish this, needs to match the outgoing interface to satisfy the requirement of filtering only the packets leaving out of VLAN162 interface. It is important to remember that PBR feature working only for inbound packets, before the final routing decision has been made. Additionally, the route-map also matches the packet size range and the access-list filtering ICMP packets only.

Task 6.3 Verification

Notice that you cannot access the backbone devices in the real exam; you could always relocate the policy and try sending traffic off a different device.

```
RS.42.1.BB1#ping 192.10.1.1 size 300

Type escape sequence to abort.
Sending 5, 300-byte ICMP Echos to 192.10.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 84/84/88 ms

RS.42.1.BB1#ping 192.10.1.1 size 105

Type escape sequence to abort.
Sending 5, 105-byte ICMP Echos to 192.10.1.1, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)

Rack1R6#show route-map
route-map DROP, permit, sequence 10
  Match clauses:
    ip address (access-lists): ICMP
    interface FastEthernet0/0
    length 100 200
  Set clauses:
    interface Null0
  Policy routing matches: 18 packets, 1897 bytes
```


Task 6.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

Security > Role Based CLI

R4:

```
!
! Enable AAA and then switch to the Root view
!
aaa new-model
!
exit
!
! Enter the enable password next!
!
enable view
!
conf t
!
! Allow show run and debug commands for this view
!
parser view DEBUG
 secret CISCO
 commands exec include all debug
 commands exec include all undebug
!
! Allow interface configuration commands for this view
!
parser view INTERFACE
 secret CISCO
 commands interface include all ip
 commands configure include interface
 commands exec include configure terminal
 commands configure include interface FastEthernet0/0
!
! Create a superview out of the above created views
!
parser view SUPER superview
 secret CISCO
 view DEBUG
 view INTERFACE

!
! Make sure the console is not authenticated
!
aaa authentication login CONSOLE none
!
line console 0
 login authentication CONSOLE
```


Task 6.4 Breakdown

The task wording clearly instructs to use RBAC as a solution. The first thing require to implement this is enabling AAA for authentication. Notice that we applied special AAA list to the console line to prevent password-based authentication for console users. In the real exam, you may not have to do this, as local credentials may have been set up for you and local authentication enabled on the console line.

After enabling AAA, we routinely create three roles. The DEBUG roles is allowed to run the exec commands “debug” and “undebug” with any parameters. The INTERFACE role is allowed to run the exec command “cofigure terminal” followed by only a specific interface command. The role is further allowed to issue any interface-level command that starts with “ip”. Finally, a super-view is created to embracing the two previously created views and inheriting their functions.

Task 6.4 Verification

Switch to view “DEBUG” and try issuing some commands:

```
Rack1R4#enable view DEBUG
Rack1R4#conf t
         ^
% Invalid input detected at '^' marker.

Rack1R4#show ?
  flash:          display information about flash: file system
  parser          Show parser commands

Rack1R4#debug ?
  IUA                      ISDN adaptation Layer options
  aaa                      AAA Authentication, Authorization and
Accounting
  aal2_xgcpspi             AAL2_XGCP Service Provider Interface.
  access-expression        Boolean access expression
  acircuit                 Attachment Circuit information
  adjacency                adjacency
  alarm-interface          Alarm Interface Card events
  all                      Enable all debugging
  alps                     ALPS debug information
  appfw                    Application Firewall events
  apple                    Appletalk information
  arap                     Appletalk Remote Access
  archive                  debug archive commands
```


Now try the other view:

```
Rack1R4#enable view INTERFACE
Rack1R4#?
Exec commands:
  configure   Enter configuration mode
  credential  load the credential info from file system
  enable      Turn on privileged commands
  exit        Exit from the EXEC
  show        Show running system information
```



```
Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface fastEthernet 0/0
Rack1R4(config-if)#?
Interface configuration commands:
  channel-group      Add this interface to an Etherchannel group
  custom-queue-list  Assign a custom queue list to an interface
  delay              Specify interface throughput delay
  exit               Exit from interface configuration mode
  ip                 Interface Internet Protocol config commands
  load-interval      Specify interval for load calculation for an
interface
  locaddr-priority   Assign a priority group
  priority-group     Assign a priority group to an interface
  sap-priority       Assign a priority group

Rack1R4(config-if)#ip ?
Interface IP configuration subcommands:
  access-group         Specify access control for packets
  accounting           Enable IP accounting on this interface
  address              Set the IP address of an interface
  admission            Apply Network Admission Control
  auth-proxy           Apply authentication proxy
  authentication       authentication subcommands
  bandwidth-percent    Set EIGRP bandwidth limit
  bgp                  BGP interface commands
  broadcast-address    Set the broadcast address of an interface
  cef                  Cisco Express Forwarding interface commands
  cgmp                 Enable/disable CGMP
  ddns                 Configure dynamic DNS
  dhcp                 Configure DHCP parameters for this interface
  directed-broadcast   Enable forwarding of directed broadcasts
  dns                  Configure DNS server
  dvmrp                DVMRP interface commands
  flow                 NetFlow related commands

Rack1R4(config)#interface fastEthernet 0/1
                                          ^
% Invalid input detected at '^' marker.

Rack1R4#enable view SUPER
Rack1R4#?
Exec commands:
  configure   Enter configuration mode
  credential  load the credential info from file system
  debug       Debugging functions (see also 'undebug')
  enable      Turn on privileged commands
  exit        Exit from the EXEC
  show        Show running system information
  undebug     Disable debugging functions (see also 'debug')
```

🕮 Deep Dive

- What are benetifs of using RBAC over the privilege level authorization?
- What does the option “include-exclusive” means when adding a new command to RBAC role?

Task 6.5 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

Security > Control Plane Protection

R5:

```
!
! ACL to for fragmented IP packets
!
ip access-list extended FRAGMENTS
 permit ip any any fragment
!
class-map FRAGMENTS
 match access-group name FRAGMENTS
!
! ICMP traffic
!
ip access-list extended ICMP
 permit icmp any any
!
class-map ICMP
 match access-group name ICMP
!
! Class-map to select closed ports.
!
class-map type port-filter match-all CLOSED_PORTS
 match closed-ports
 match not port TCP 2020
 match not port TCP 2040
!
policy-map type port-filter HOST_PORT_FILTER
 class CLOSED_PORTS
  drop
!
! Control plane rate-limit policies
!
policy-map HOST_RATE_LIMIT
 class ICMP
  police rate 10 pps burst 5 packets
!
policy-map CEF_EXCEPTION_RATE_LIMIT
 class class-default
  police rate 100 pps burst 20 packets
!
policy-map TRANSIT_RATE_LIMIT
 class FRAGMENTS
  police rate 1000000 pps burst 200000 packets
```



```
!
! Applying the policies together
!
control-plane host
  service-policy input HOST_RATE_LIMIT
  service-policy type port-filter input HOST_PORT_FILTER
!
control-plane transit
  service-policy input TRANSIT_RATE_LIMIT
!
control-plane cef-exception
 service-policy input CEF_EXCEPTION_RATE_LIMIT
```

Task 6.5 Breakdown

The only feature that may restrict traffic flow toward the control plane in granular fashion is Control Plane Protection (CPPr). CPPr extended the classic control plane policing feature, treating the Route Processor (RP) as a virtual interface attached to the router. All packets redirected to the RP for some reason are classified into three categories corresponding to three subinterfaces of the virtual interface:

`Control-plane host subinterface`. This interface receives all control-plane TCP/UDP traffic that is directly destined for one of the router interfaces. Examples of control-plane host traffic include management traffic or routing protocols. Notice that non TCP/UDP control traffic will end up on the CEF-exception sub-interface. Most control plane protection features operate on this subinterface and thus this sub-interface provides most features, such as policing, port filtering and per-protocol queue thresholds.

Port-filtering allows for automatically dropping of packets destined for the TCP/UDP ports not currently open in the router. The operating system automatically detects all open ports, and you can manually configure some exceptions. The net effect is reduced load on router's CPU during the times of flooding attacks.

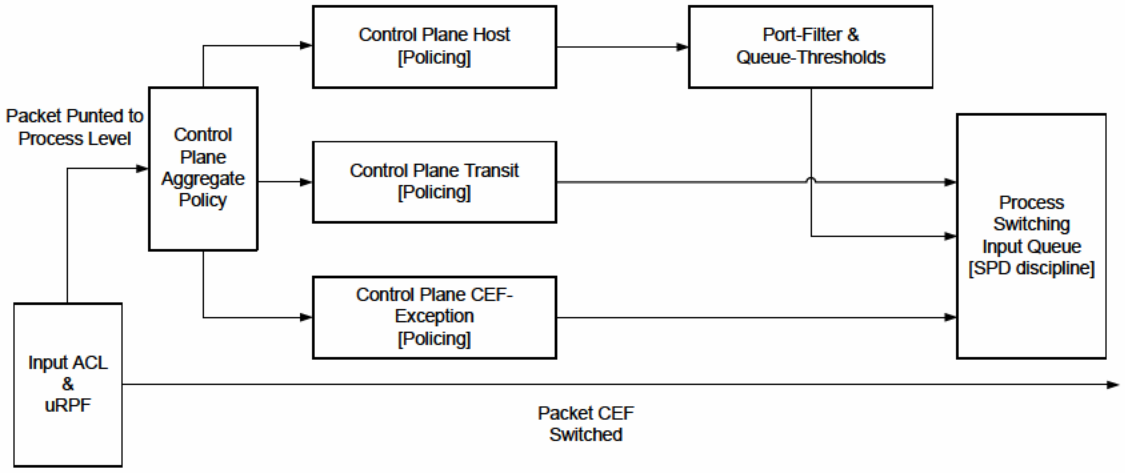
Per-protocol queue thresholds allow you setting selective queue limits for packets of different type, for example BGP, OSPF, FTP, DNS, HTTP, IGMP, SNMP etc.

`Control-plane transit subinterface`. This subinterface is intended to handle *transit* IP packets not handled via CEF mechanism and needed to be switching in the processor context. This might happen, for instance, when a packet mast be routed out of Ethernet interface and no ARP lookup has been made yet for the next-hop, making the CEF adjacency incomplete.

`Control-plane CEF exception subinterface`. This is where packets causing an exception in CEF switching path land. So far, those include non-IP router-destined traffic, such as CDP, L2 keepalive messages (e.g. Frame-Relay LMI keepalives) or ARP packets and IP packets with options or TTL <=1. Additionally, non-UDP local multicast traffic destined to the router (e.g. OSPF updates) fall into this category as well.

You may apply a separate rate-limiting policy to any of the sub-interface or have a single aggregate policy embracing all subinterfaces (classic control plane policing). It is possible to configure both the subinterface and aggregate policy, but this feature appears to work unstable in many 12.4T images. So far, it seems to be better configuring either aggregate or subinterface specific policies.

Before any packet reaches any specific control plane subinterface, it is first processed via a series of ingress features including input access-list, uRP checks, aggregate control-plane policy. After passing the subinterface-specific policy, the packet is then being enqueued onto the respective interface input queue and handled via SPD (selective packet discard) policy. Here is a visual outline of the process:



The port-filter allows for matching the ports open on the router and setting queue thresholds or drop packets selectively. You can also match all closed ports on the router as a whole unit and drop packets destined to the non-listening ports before the router every has chance to respond with ICMP unreachable or TCP RST packet.

In our solution, we start by limiting the traffic going toward the host subinterface, which means to the RP itself. The host rate-limiting policy control the rate of ICMP traffic in packet per second. Additional host-policy applies port-filter, blocking all closed ports with except to ports 2020 and 2040.

Other policies deal with transit fragmented traffic, which is matched based on an access-list. This traffic is limited to the requires packet per second rate on the transit subinterface. Finally, all packets resulting in CEF exceptions are limited to 100 packets per second using the respective control plan subinterface.

Task 6.5 Verification

Check the security features enabled for control plane:

```
Rack1R5#show control-plane features
Total 4 features configured

Control plane host path features :

-------------------------------------------------------
Control-plane Policing activated Dec XX 200X 03:0
TCP/UDP Portfilter activated Dec XX 200X 03:0

-------------------------------------------------------

Control plane transit path features :

-------------------------------------------------------
Control-plane Policing activated Dec XX 200X 03:0

-------------------------------------------------------

Control plane cef-exception path features :

-------------------------------------------------------
Control-plane Policing activated Dec XX 200X 03:0

-------------------------------------------------------

Rack1R5#show control-plane counters
Feature Path              Packets processed/dropped/errors
Aggregate                   164103/0/0
Host                         13840/0/0
Transit                       9736/0/0
Cef-exception               140558/0/0
```

🕮 Deep Dive

- Why would you want to drop packets going to closed ports?
- Why is it important to limit CEF-exception traffic rate?
- What may put heavy stress on router input queues?

Task 7.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

System Management > SNMPv2 Server
System Management > SNMPv2c Access Control

```
R6:
!
! Log unauthorized access
!
access-list 25 permit 192.10.1.101
access-list 25 deny   any log
!
snmp-server community CISCORO RO 25
snmp-server community CISCORW RW 25
!
snmp-server location San Jose, CA US
snmp-server contact CCIE Lab R6
snmp-server chassis-id 556-123456
!
snmp-server trap-source Loopback0
snmp-server host 192.10.1.101 traps CISCOTRAP
```

Task 7.1 Breakdown

Although this is a relatively standard SNMP configuration, the task requiring to configure logging could be difficult to interpret. This key is the word “logged”. Even though the router could notify the NMS via an SNMP trap by using the **snmp-server enable traps snmp authentication** command, the tasked requested that the failed attempts be logged. To *log* the failed attempts, an access-list entry was used that denied all other IP addresses and logged them. Of course, in a real network, a remote syslog server would also be configured or at least **logging buffered** would have been enabled.

🕮 Deep Dive

- How SNMP notifications are different from traps?

Task 7.1 Verification

Verify basic SNMP configuration:

```
Rack1R6#show snmp
Chassis: 556-123456
Contact: CCIE Lab R6
Location: San Jose, CA US
<output omitted>
SNMP logging: enabled
    Logging to 192.10.1.101.162, 0/10, 0 sent, 0 dropped.
```

Verify that logging is configured for access-list 25:

```
Rack1R6#show ip access-lists 25
Standard IP access list 25
    10 permit 192.10.1.101
    20 deny   any log

Rack1R6#show snmp host
Notification host: 192.10.1.101 udp-port: 162   type: trap
user: CISCOTRAP security model: v1
```

Task 7.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

System Management > System Messages Logging
System Management > Syslog Logging

R4 and R5:

```
logging trap notifications
logging origin-id hostname
logging facility sys10
logging source-interface Loopback0
logging 192.10.1.101
```

Task 7.2 Breakdown

By default, syslog messages do not include the hostname of the device in them. In a real network, this can be very useful when viewing the syslog messages on the syslog server itself. The **logging origin-id** command enables the hostname, IP address, or even (as of 12.3[1]), a user defined string in the log messages.

Task 7.2 Verification

Verify that logging is configured and the origin-id is set:

```
Rack1R5#show logging
Syslog logging: enabled (11 messages dropped, 1 messages rate-limited,
<output omitted>

    Trap logging: level notifications, 78 message lines logged
        Logging to 192.10.1.101 (udp port 514, audit disabled, link
up), 4 message lines logged, xml disabled,
              filtering disabled

Rack1R5#show running-config | include origin-id
logging origin-id hostname
```

🕮 Deep Dive

- How would you protect against console getting flooding with syslog messages?
- How would you protect against syslog message tampeing?

Task 7.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

System Management > Terminal Line Settings
System Management > IOS Authoriative DNS Server

```
R6:
!
! Configure a name server and enable DNS lookups
!
ip name-server 192.10.1.100
!
ip domain-lookup
!
line con 0
 transport preferred none
```

Task 7.3 Breakdown

Although this could be considered a Stupid Router Trick (SRT), it is a very useful command to have enabled in a lab environment or even real network when DNS is being used. The **transport preferred none** line command will stop the router from trying to resolve a mistyped command via DNS.

The router is technically attempting to resolve the string entered via DNS as it's trying to telnet to a device with that particular name. Once the **transport preferred none** command is enabled, you will need to use the **telnet** exec mode command to telnet to another device.

🕮 Deep Dive

- How does IOS DNS proxy mode work?

Task 7.3 Verification

```
Rack1R6#r1
          ^
% Invalid input detected at '^' marker.
```


Task 7.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

Security > AAA Local Command Authorization

```
R6:
!
! Create a password for "level 2"
!
enable secret level 2 CISCO
!
! Move the required command to level 2
!
privilege interface level 2 ip access-group
privilege interface all level 2 encapsulation
privilege configure level 2 hostname
privilege configure level 2 interface
privilege exec level 2 show run
```

Task 7.4 Brekadown

Just because the show run command is moved to a lower level, it doesn't mean a user at that level can see the entire configuration. A user will only be able to see the items that they have the appropriate privilege level for. Make sure to test!! You may want to add an access-list to an interface to verify that it shows up in the output of show run.

🕮 Deep Dive

- How would you assign all subcommands of a given command to a privilege level?

Task 7.4 Verification

Switch to privilege level 2 and make sure you cannot configure anything but list some of the running configuration settions.

```
Rack216>enable 2
Password:
Rack16#conf t
           ^
% Invalid input detected at '^' marker.

Rack1R6#sh run
Building configuration...

Current configuration : 326 bytes
!!
!
hostname Rack1R6
!
boot-start-marker
boot-end-marker
!
!
!
!
!
interface Loopback0
!
interface FastEthernet0/0
!
interface FastEthernet0/1
!
interface Serial0/0/0
 encapsulation frame-relay
!
!
end
```


Task 7.5 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge that could be obtained by completing the following VOL1 scenario(s):

IP Services > WCCPv2 Services

R5:

```
!
! Access-list for cache-enginer and redirected subnet
!
access-list 10 permit 162.1.5.200
access-list 10 permit 162.1.5.100
access-list 50 permit 162.1.55.0 0.0.0.255
!
! Configure a new dynamic WCCPv2 group
!
ip wccp version 2
ip wccp 99 group-address 224.0.1.100 redirect-list 50 group-list 10
password CISCO
!
! Configure interfaces to redirect traffic and listen to WCCPv2
!
interface FastEthernet 0/1
 ip wccp 99 redirect in
!
interface FastEthernet 0/0
 ip wccp 99 group-listen
```


Task 7.5 Breakdown

The task requires for configuring a dynamic service group number 99 in R5. WCCPv2 allows for intercepting and redirecting protocols other than just HTTP. The information about the dynamic group is communicated to the router by the cache engines.

In the solution, we configure two access-lists: one to control the cache engines eligible to register with the router and another to control the source subnets eligible for redirection. The dynamic WCCP group is configured with a fixed multicast address that will be used for cache-engines auto-disovery, along with associated access-lists for the cache engines and redirected subnets.

Two interfaces are configured – one for listening to the dynamic group protocol and redirect incoming requests and another to listen to the cache engine heartbeats and discovering them.

🕮 Deep Dive

- How can you ensure that outbound access-lists are checked before a packet gets redirected?

Task 7.5 Verification

Check the WCCP group status.

```
Rack1R5#show ip wccp 99
Global WCCP information:
    Router information:
        Router Identifier:                   -not yet determined-
        Protocol Version:                    2.0

    Service Identifier: 99
        Number of Service Group Clients:     0
        Number of Service Group Routers:     0
        Total Packets s/w Redirected:        0
          Process:                           0
          CEF:                               0
        Service mode:                        Open
        Service Access-list:                 -none-
        Total Packets Dropped Closed:        0
        Redirect Access-list:                50
        Total Packets Denied Redirect:       0
        Total Packets Unassigned:            0
        Group Access-list:                   10
        Total Messages Denied to Group:      0
        Total Authentication failures:       0
        Total Bypassed Packets Received:     0
```

Verify WCC interface-level settings:

```
Rack1R5#show ip wccp interfaces detail
WCCP interface configuration details:
    FastEthernet0/1
        Output services: 0
        Input services:  1
        Static:          None
        Dynamic:         099
        Mcast services:  1
        Static:          None
        Dynamic:         099
        Exclude In:      FALSE
```


Task 7.6 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

System Management > EEM Scripting: CLI Events

R6:

```
!
! Create a script matching the CLI pattern "interface Loopback"
!
no event manager applet LOOPBACK_SHUTDOWN
 event manager applet LOOPBACK_SHUTDOWN
 event cli pattern ".*interface Loopback.*" sync yes
 action 1.0 cli command "enable"
 action 1.1 cli command "configure terminal"
 action 1.2 cli command "$_cli_msg"
 action 1.3 cli command "shutdown"
 end
```

Task 7.6 Breakdown

This simple EEM script is configured to be synchronous, i.e. the actuall command that triggered the script will be delayed until the script completes. Since we don't set the script exit value to 1, the triggering command will never be executed.

The CLI pattern is set to match any command that starts with "interface Loopback" meaning it will match any command accessing a loopback interface. One trick in the script is that the triggering "$_cli_msg" is used as a command itself to pick up the proper loopback to be shut down.

🕮 Deep Dive

- How would you modify the script to ensure that user does not accidentally shutdown a WAN interface (e.g. a Serial link)?

Task 7.6 Verification

Create a new loopback interface and try accessing it:

```
Rack1R6(config)#interface loopback 80
Rack1R6(config-if)#exit

Rack1R6(config)#interface loopback 80

Rack1R6(config)#
%SYS-5-CONFIG_I: Configured from console by vty0
Rack1R6(config)#do sh run int lo 80
Building configuration...

Current configuration : 53 bytes
!
interface Loopback80
 no ip address
 shutdown
end
```

Task 8.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > Using Class Based GTS for FRTS

```
R1:
policy-map SHAPE_384K
 class class-default
  shape average 384000 38400 12800
  shape adaptive 256000
!
interface Serial0/0
 service-policy output SHAPE_384K
```

Task 8.1 Breakdown

The following values on R1 can be inferred from this task's description:

AR (port speed) = 512000 bps
CIR (average rate) = 384000bps
Tc (interval length) = 100 ms

The following values are then calculated based on the given values and the below formula:

Bc = CIR * Tc/1000
Bc = 38400 bits

Be = (AR – CIR) * Tc/1000
Be = 12800 bits

🕮 Deep Dive

- What FRTS features are not supported by MQC class-based traffic-shaping?

Task 8.1 Verification

Verify the shaping parameters:

```
R1#show policy-map interface serial 0/0

 Serial0/0

  Service-policy output: SHAPE_384K

    Class-map: class-default (match-any)
      10 packets, 797 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval
Increment
             Rate           Limit  bits/int  bits/int  (ms)
(bytes)
           384000/384000    6400   38400     12800     100       4800

        Adapt  Queue     Packets   Bytes     Packets   Bytes
Shaping
        Active Depth                         Delayed   Delayed   Active
        -      0         6         745       0         0         no
```

Task 8.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > Frame-Relay TCP and RTP Header Compression

R3:

```
interface Serial1/0
 frame-relay map ip 162.1.0.4 304 broadcast rtp header-compression
passive connections 15
```

R4:

```
interface Serial0/0/0
 frame-relay map ip 162.1.0.3 403 broadcast rtp header-compression
connections 15
```

Task 8.2 Breakdown

RTP header compression allows the reduction of the header inside an RTP packet to be reduced from 40 bytes to 2 – 5 bytes. RTP compression is best used on low speed links for real time traffic with small data payloads, such as VoIP. To configure RTP on a serial link, use the **ip rtp header-compression** command. For Frame Relay links, RTP compression can be configured on a per VC basis as in the above example. The **passive** keyword of the RTP statement means that the router will not start sending RTP compressed headers unless RTP headers are received.

🕮 Deep Dive

- Why would you want to compress TCP headers along with RTP/UDP?

Task 8.2 Verification

Verify the per-VC configured RTP header compression:

```
Rack1R4#show frame-relay map

<output omitted>

Serial0/0/0 (up): ip 162.1.0.3 dlci 403(0x193,0x6430), static,
              broadcast,
              CISCO, status defined, active
              RTP Header Compression (enabled), connections: 15
```

Task 8.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC Classification and Marking
QoS > MQC Class-Based Generic Traffic Shaping

R2:

```
ip cef
!
! Use NBAR to match SQL Server traffic
!
class-map match-all SQL
 match protocol sqlserver
!
! Shape and set the shaper buffer settings
!
 policy-map SQL_POLICY
  class SQL
   shape average 256000
   shape max-buffers 2048
!
interface Serial0/0.1 point-to-point
 service-policy output SQL_POLICY
```


Task 8.3 Breakdown

The above task uses Network Based Application Recognition to match SQL traffic (TCP port 1433). As SQL traffic leaves the serial interface, it is shaped to 256Kbps. The shaping buffers are modified to allow up to 2048 packets to sit in the shaping queue while waiting to be sent out. A large shaping queue is advantageous for delay insensitive data traffic for which you do not want to be dropped.

The above configuration is known as Generic Traffic Shaping. GTS uses the same principles and calculations as does Frame Relay Traffic Shaping, but does not adaptively shape, and is supported on non-Frame Relay interfaces

Task 8.3 Verification

Check the policy-map applied to the interface:

```
Rack1R2#show policy-map interface serial 0/0.1

 Serial0/0.1

  Service-policy output: SQL_POLICY

    Class-map: SQL (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol sqlserver
      Traffic Shaping
         Target/Average   Byte   Sustain   Excess   Interval  Increment
           Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           256000/256000    1984   7936      7936      31        992

       Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
       Active Depth                         Delayed   Delayed   Active
       -      0         0         0         0         0         no

    Class-map: class-default (match-any)
      87 packets, 8370 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```


Task 8.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

```
QoS > Catalyst QoS ACL Based Classification & Marking
QoS > Catalyst 3560 Per VLAN Classification
QoS > Catalyst 3560 Per-Port Per-VLAN Policing
```

Additional information could be found in the following posts on the INE blog:

http://blog.ine.com/2008/10/30/traffic-classification-in-the-35503560-switches/

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

SW1:

```
mls qos

!
! Access-Lists matching respective traffic (QoS ACLs)
!
ip access-list extended TCP
 permit tcp any any
!
ip access-list extended UDP
 permit udp any any
!
mac access-list extended IPX
 permit any any 0x8137 0x0
!
! First-level class-maps
!
class-map TCP
 match access-group name TCP
!
class-map UDP
 match access-group name UDP
!
class-map IPX
 match access-group name IPX

mls qos map policed-dscp 0 to 8

!
! For 2nd level policy you can only match input interfaces
!
class-map TRUNKS
 match input-interface FastEthernet 0/13 -  FastEthernet 0/21
```

```
!
! Second-level policy-map may only police, but not mark
!
policy-map INTERFACE_POLICY_TCP
 class TRUNKS
  police 1000000 16000 exceed policed-dscp-transmit

!
! Second-level policy-map may only police, but not mark
!
policy-map INTERFACE_POLICY_UDP
 class TRUNKS
  police 512000 16000 exceed drop

!
! Second-level policy-map may only police, but not mark
!
policy-map INTERFACE_POLICY_IPX
 class TRUNKS
  police 128000 16000 exceed drop

!
! 1st level policy-map may only mark, not police
! VAN aggregate policing is not possible in the 3560
!
policy-map VLAN_POLICY
 class TCP
  set dscp 0
  service-policy INTERFACE_POLICY_TCP
!
! Nothing is told about UDP marking and we need to set something.
! Thus we just select the default dscp value of zero.
!
 class UDP
  set dscp 0
  service-policy INTERFACE_POLICY_UDP
 class IPX
  set dscp cs2
  service-policy INTERFACE_POLICY_IPX
!
interface Vlan 162
 service-policy input VLAN_POLICY
!
!
! Enable VLAN-based QoS on physical interfaces
!
interface range Fa0/13 - 21
 mls qos vlan-based
```


Task 8.4 Breakdown

There are three main requirements in the scenario:

- Police traffic entering VLAN 162 on any port. This translates into VLAN-based classification.
- Provide separate policing rates for UDP, TCP and IPX traffic. Both UDP and TCP are IP-based and IPX is a non-IP protocol.
- Mark all three traffic classes and remark TCP packets if they exceed the policed rate.

The 3560 switches support hierarchical policing by means of nested policies. However, it is important to notice that these policies are not aggregate across all port members in a VLAN but apply to every ingress port individually.

We start by creating three access-lists to match TCP, UDP and IPX traffic per the task requirements. After this, we create three class-maps corresponding to these access-lists to be later used as top-level class maps in the final policy.

In order to apply hierarchical policing, a special policy map that matches just port ranges need to be created. We accomplish this by creating class-map “TRUNK” matching the trunk port range. We only match the trunks since VLAN162 is not assigned to any local port. This port range is where the policing will apply.

Next in turn we create three policy-maps matching the “TRUNKS” class and setting policy rate for every of three mentioned protocols, per the task requirements. Notice that for TCP we set action to “set policed dscp transmit” all the policies for UDP and IPX specify the “drop” action. These are the “Second-level” policies which specify the per-port policing rate for specific traffic classes.

The last, top-level policy-map matches the three initial traffic classes we created to TCP, UDP and IPX. At the top-level we can only mark traffic; hence we use the “set” commands to apply marking to traffic classes per the scenario requirements. Notice that you cannot set CoS directly in the 3560 switched so we set DSCP CS2 for these packets and this marking translates to CoS=2 by means of the default DSCP to CoS mapping table. Every class in the top-level policy then features the “class-specific” policy map we created previously limiting the traffic rate. This completes the policing/marking configuration, and we only need to apply the policy to an SVI for VLAN 162 and enable VLAN-based QoS on SW1’s trunk port.

Task 8.4 Verification

Check the policy map configuration on the interface

```
Rack1SW1#show policy-map interface
 Vlan162

  Service-policy input: VLAN_POLICY

    Class-map: TCP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name TCP

      Service-policy : INTERFACE_POLICY_TCP

        Class-map: TRUNKS (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: input-interface  FastEthernet0/13 - FastEthernet0/21

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

    Class-map: UDP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name UDP

      Service-policy : INTERFACE_POLICY_UDP

        Class-map: TRUNKS (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: input-interface  FastEthernet0/13 - FastEthernet0/21

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



```
Class-map: IPX (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name IPX

      Service-policy : INTERFACE_POLICY_IPX

        Class-map: TRUNKS (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: input-interface  FastEthernet0/13 - FastEthernet0/21

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

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

Vol1 Scenarios Reference

Bridging and Switching > Layer 2 Etherchannel
Bridging and Switching > 802.1q Trunking
Bridging and Switching > 802.1q Native VLAN
Bridging & Switching > MAC Address Table Static Entries & Aging
RIPv2 > RIPv2 Authentication
EIGRP > EIGRP Metric Weights
OSPF > OSPF Stub Areas
RIPv2 > RIPv2 Unicast Updates
BGP > BGP Remove Private AS
BGP > BGP Dampening
BGP > BGP Dampening with Route-Maps
BGP > BGP Communities – No Advertise
IPv6 > IPv6 Link-Local Addressing
IPv6 > IPv6 MP-BGP
IP Multicast > AutoRP – Multiple Candidate RPs
IP Multicast > AutoRP – Filtering Candidate RPs
IP Multicast > AutoRP – Filtering AutoRP Messages
IP Multicast > Multicast Boundary
IP Multicast > Stub Multicast Routing & IGMP Helper
IP Multicast > PIM Sparse Mode
Security > Traffic Filtering with Reflexive Access-Lists
Security > VLAN Filters for IP Traffic
Security > Traffic Filtering with Policy Based Routing
Security > Role Based CLI
Security > Control Plane Protection
System Management > SNMPv2 Server
System Management > SNMPv2c Access Control
System Management > System Messages Logging
System Management > Syslog Logging
System Management > Terminal Line Settings
System Management > IOS Authoriative DNS Server
Security > AAA Local Command Authorization
IP Services > WCCPv2 Services
System Management > EEM Scripting: CLI Events
QoS > Using Class Based GTS for FRTS
QoS > Frame-Relay TCP and RTP Header Compression
QoS > MQC Classification and Marking
QoS > MQC Class-Based Generic Traffic Shaping
QoS > Catalyst QoS ACL Based Classification & Marking
QoS > Catalyst 3560 Per VLAN Classification
QoS > Catalyst 3560 Per-Port Per-VLAN Policing

Deep Dive Answers

- What is the other option to ensure all frames are tagged when sent across a 802.1Q trunk?
  - The alternative to using ISL is enable native VLAN tagging in switches using the global command **`vlan dot1q tag native`**.
- Why would you always want to tag frames sent across a trunk?
  - For security reasons. The other end may have different native VLAN configured and this will allow traffic leaking between two different VLANs.
- In what scenarios would you want to disable MAC address learning at all?
  - When you connect just two endpoints, effectively implementing point-to-point connection. In this case, using MAC address learning is useless and only consumes switch CAM resources.
- What type of OSPF areas would you most likely use in production?
  - Totally stubby areas are preferred in production as that reduce the amount of routing information to a minimum, sacrificing some optimal routing.
- How can you reduce LSA flooding?
  - You cannot increate the LSA max-age of one hour but you may use the command **`ip ospf flooding-reduction`** on an interface to flood the LSAs with the DNA (Do Not Age) bit set.
- Could you activate IPv6 address-family for IPv4 neighbor in BGP?
  - It was possible to do that in some older versions of IOS code, as capability is technically independent of the physical connection, but the resulting configuration would never have worked. Even though IPv6 prefixes were passed, the next-hop value was IPv4 and it was never possible to use it for IPv6 RIB.
- Why can't you use "deny" statements to filter unneeded groups out of specific RP?
  - The deny statement in Auto-RP access-list signalizes that given group range is to be flooded in dense mode, ignoring any RP assignment. Therefore, setting a preferred RP would not work.
- What is main use of the PIM neighbor-filter command and what other option could replace it?
  - This command was mainly required to make a multicast interface "PIM passive", effectively allowing multicast traffic forwarding but disallowing any downstream PIM peers to signal anything. The new command found in IOS 15 allowing for practically the same functionality is **`ip pim passive`**.
- Will multicast boundary affect IGMP joins?

  - Yes, multicast boundary affects PIM and IGMP Joins and multicast traffic flow. Additionally, the AutoRP announcements are also filtered to prevent downstream nodes from ever learning the prohibited groups.
- Where should you put the PIM SPT threshold setting?
  - At the edge of the multicast domain close to the traffic receivers. It's the leaf multicast routers that switch to the SPT.
- What is the purpose of the "established" keyword used in the access-lists?
  - It only permits the TCP segments with the ACK bit set. This means only traffic for already established sessions. This feature simulates simple "stateful" functionality for TCP traffic.
- What CBAC feature corresponds to the "policy-based routing" trick used with reflexive access-lists?
  - CBAC supports router traffic inspection for TCP, UDP and ICMP traffic that could be configured in CBAC rule. Some other protocols sources locally, such as H.323 signaling are also supported.
- Can you combine interface-level access-lists along with VLAN filters?
  - Only if the directions are different. VLAN filters are always ingress – you may combine them with outgoing VLAN access-groups but cannot combine them with any ingress filtering feature.
- Which direction do VLAN filters work?
  - Ingress only, but it is important to notice that traffic entering a VLAN will necessarily be leaving it as well, unless its destined to the local switch (e.g. to an SVI).
- What are benefits of using RBAC over the privilege level authorization?
  - RBAC allows for flexible grouping of IOS commands into roles, which could follow non-hierarchical structure.
- What does the option "include-exclusive" means when adding a new command to RBAC role?
  - When adding a new command to an RBAC role, the keyword "include-exclusive" means the command could be only used in this role and cannot be reused in any other. Normally this is a security precaution to ensure a given command is only allowed to a single role.
- Why would you want to drop packets going to closed TCP ports?
  - By default, many operation systems respond with TCP RST when they receive a packet address to a closed TCP port or with ICMP port unreachable when a packet is received for close UDP port. This could be used to identify closed ports and see if they are filtered or just closed. Additionally, this puts extra stress on router resources.
- Why is it important to limit CEF-exception traffic rate?

  - CEF exceptions are processed by router's CPU and could be a burden if high packet rate is punted to the RP. Therefore it is important to protect control plane against these packets.
- What may put heavy stress on router input queues?
  - In ISP environments, BGP tables load may result in large amount of TCP ACK packets sent by neighbors and putting excessive load on router's input queue.
- How SNMP notifications are different from traps?
  - SNMP notifications are known as "reliable" SNMP traps, as they require to be acknowledge by the management station.
- How does IOS DNS proxy mode work?
  - In DNS proxy mode, an IOS router is configured with the IP address of DNS server using the command "ip name-server" and configured as DNS server using the command "ip dns-server". This allows the router to forward all DNS requests to the upstream DNS server.
- How would you assign all subcommands of a given command to a privilege level?
  - There is a special option "all" when adding a new command to privilege level. It automatically adds all subcommands that start with the keyword specified to the privilege level.
- How can you ensure that outbound access-lists are checked before a packet gets redirected to WCCP server?
  - The command "ip wccp outbound acl-check" ensures that a packet is first checked against outbound access-lists prior to being redirected to a cache engine.
- How would you modify the EEM script in the task to ensure that user does not accidentally shutdown a WAN interface (e.g. a Serial link)?
  - You may watch the syslog messages using the Syslog event detector to see if the interface goes down and then issue the "no shutdown" command on the given link to make sure it was not shut down.
- What FRTS features are not supported by MQC class-based traffic-shaping?
  - Frame-Relay Adaptive Traffic Shaping for a long time has not been supported by MQC-based generic traffic shaping. Support has been added in latest IOS versions.
- Why would you want to compress TCP headers along with RTP/UDP?
  - TCP connections are often used for VoIP signaling, e.g. with SCCP protocol. Compressing headers in this situation would be beneficial as it frees more bandwidth for RTP traffic.

Lab 6 Solutions

Task 1.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Bridging and Switching > 802.1q Trunking

SW1:

```
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface Fa0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport trunk allowed vlan except 7,77,777
!
interface Fa0/19
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport trunk allowed vlan except 7,77,777
```

☞ Quick Note
When the *allowed vlan except* option is used, the configuration will show the command without the except option displaying all of the allowed VLANs.

SW2, SW3, and SW4:

```
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
```


Task 1.1 Breakdown

The task requirements are straightforward. Effectively, the switches form start topology with SW1 being the hub node. The only non-standard requirement pertains to VLAN filtering on certain trunks. This is accomplished using allowed VLAN list. This filtering results in the situation where VLAN7, 77 and 777 traffic is not leaked down to SW3 and SW4 but remains between SW1 and SW2.

🕮 Deep Dive

- Trunking applies additional header to Ethernet frames. Why doesn't this create any MTU problems in the networks?

Task 1.1 Verification

Check the trunk status on SW1, the hub node. Look for allowed VLAN lists and trunking encapsulation.

```
Rack1SW1#show interface trunk | include (Encap|802|allowed on|4094)
Port        Mode         Encapsulation  Status        Native vlan
Fa0/13      on           802.1q         trunking      1
Fa0/16      on           802.1q         trunking      1
Fa0/19      on           802.1q         trunking      1
Port        Vlans allowed on trunk
Fa0/13      1-4094
Fa0/16      1-6,8-76,78-776,778-4094
Fa0/19      1-6,8-76,78-776,778-4094
```

Task 1.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Bridging and Switching > STP Root Bridge Election

SW1:

```
spanning-tree vlan 1-4094 root primary
```


Task 1.2 Breakdown

The wording of this scenario may appear to be tricky. First, it asks to ensure SW1 is always in STP forwarding state for all existing VLANs. Since the topology is hub-and-spoke, this means SW1 should be the root bridge for all existing VLANs.

The second requirement may be a bit confusing, since it mentions VTP domain NET12, which has never been explicitly mentioned before. Hower, as part of your existing configuration discovery, you should have learned that all switches belong to a VTP domain NET12. Therefore, the requirement simply means that SW1 should be the root bridge for ANY VLAN added to the domain in the future.

🕮 Deep Dive

- Is it possible to make a non-root bridge to always remain STP forwarding on all ports?

Task 1.2 Verification

Check that SW1 is forwarding on all ports/VLANs

```
Rack1SW1#show span vlan 1-4094 | i VLAN|root|FWD|BLK
VLAN0001
             This bridge is the root
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
Fa0/19           Desg FWD 19        128.21   P2p
Fa0/20           Desg FWD 19        128.22   P2p
Fa0/21           Desg FWD 19        128.23   P2p
VLAN0005
             This bridge is the root
Fa0/5            Desg FWD 19        128.7    P2p
Fa0/13           Desg FWD 19        128.15   P2p
Fa0/16           Desg FWD 19        128.18   P2p
Fa0/19           Desg FWD 19        128.21   P2p
VLAN0006
             This bridge is the root
Fa0/13           Desg FWD 19        128.15   P2p
Fa0/16           Desg FWD 19        128.18   P2p
Fa0/19           Desg FWD 19        128.21   P2p
VLAN0007
             This bridge is the root
Fa0/13           Desg FWD 19        128.15   P2p
VLAN0010
             This bridge is the root
Fa0/13           Desg FWD 19        128.15   P2p
Fa0/16           Desg FWD 19        128.18   P2p
Fa0/19           Desg FWD 19        128.21   P2p
VLAN0027
             This bridge is the root
Fa0/13           Desg FWD 19        128.15   P2p
Fa0/16           Desg FWD 19        128.18   P2p
Fa0/19           Desg FWD 19        128.21   P2p
VLAN0032
             This bridge is the root
Fa0/3            Desg FWD 19        128.5    P2p
Fa0/13           Desg FWD 19        128.15   P2p
Fa0/16           Desg FWD 19        128.18   P2p
Fa0/19           Desg FWD 19        128.21   P2p
VLAN0042
             This bridge is the root
Fa0/13           Desg FWD 19        128.15   P2p
Fa0/16           Desg FWD 19        128.18   P2p
Fa0/19           Desg FWD 19        128.21   P2p
VLAN0077
```



```
             This bridge is the root
Fa0/13           Desg FWD 19        128.15   P2p
VLAN0777
             This bridge is the root
Fa0/13           Desg FWD 19        128.15   P2p
Rack1SW1#

Rack1SW1#show spanning-tree root port
VLAN0001         This bridge is root
VLAN0004         This bridge is root
VLAN0005         This bridge is root
VLAN0006         This bridge is root
VLAN0007         This bridge is root
VLAN0010         This bridge is root
VLAN0027         This bridge is root
VLAN0032         This bridge is root
VLAN0040         This bridge is root
VLAN0042         This bridge is root
VLAN0045         This bridge is root
VLAN0049         This bridge is root
VLAN0077         This bridge is root
VLAN0777         This bridge is root

Rack1SW1#show interface trunk | begin allowed and active
Port        Vlans allowed and active in management domain
Fa0/13      1,5-7,10,27,32,77,777
Fa0/16      1,5-6,10,27,32
Fa0/19      1,5-6,10,27,32

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      1,5-7,10,27,32,77,777
Fa0/16      1,5-6,10,27,32
Fa0/19      1,5-6,10,27,32
```


Task 1.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Bridging and Switching > 802.1q Tunneling

In this task, SW2 and R4 are considered the customer devices and SW3 & SW4 are the “provider edge” switches.

SW3 and SW4:
```
system mtu 1504
```

SW3:
```
vlan 100
```

SW3:
```
interface FastEthernet0/18
switchport access vlan 100
switchport mode dot1q-tunnel
l2protocol-tunnel cdp
```

SW4:
```
interface FastEthernet0/4
switchport access vlan 100
switchport mode dot1q-tunnel
l2protocol-tunnel cdp
```


Task 1.3 Breakdown

The basic concept behind 802.1q tunneling (QinQ) is to allow for an additional tag to be applied to the Ethernet frame. This is commonly used by service providers to provide end-to-end transparent Ethernet services for their customers (Metro Ethernet). This additional tag, sometimes called the metro tag, allows for the service provider to carry all of the customer's traffic in a single separate VLAN without concern as to what traffic is being carried. This traffic could be unicast, broadcast, multicast, CDP, STP, or VTP.

QinQ tunneling can additionally allow the customer to trunk transparently across the service provider's network. When the customer's switch is trunking "to" the service provider's switch, all of the customer's trunks are carried inside the single metro VLAN when transiting the service provider's switches. In this case, the Ethernet frames will carry two tags. The inner tag was assigned by the customer's switches (i.e. the customers VLANs), and the outer tag is assigned by the service provider's edge switch (Metro tag). In order to support the additional extra 4 byte metro tag, the system MTU should be set to 1504. The default system MTU is 1500 bytes.

When using QinQ tunneling, CDP, STP and VTP are not carried across the *tunnel* by default. To allow for the carrying of these protocols, the interfaces on the service provider's edge switches need to be configured using the feature

This task may look like it requires simple L2 protocol tunneling on the ports connecting R4's and SW2's interfaces. However, if you look at the diagram and inspect the initial configurations you will notice that R4 has tagged sutinerface configured on its connection to SW4. This means you need the ports to be configured in QinQ mode, which requires the special port mode.

🕮 Deep Dive

- How does STP interpret a QinQ tunnel?
- For QinQ packets, do core devices still need to learn the MAC addresses of encapsulated frames?

Task 1.3 Verification

Looking at the configuration afterwards on the switches, you may notice that the command **no cdp enable** is added automatically as soon as you enabled the QinQ interface mode. Also, depending how quickly you check after configuration, you may still see an entry on R4 that shows SW4 as a neighbor. Pay attention to the holdtime values. A higher value indicates a more recently received CDP message.

```
Rack1R4#ping 191.1.48.8

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 191.1.48.8, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1R4#show cdp neighbors Fa0/1 | include SW2
Rack1SW2      Fas 0/1         121          S I       WS-C3560-2Fas 0/18

Rack1SW2#show cdp neighbors fa0/18 | include R4
Rack1R4       Fas 0/18        134          R S I     3640       Fas 0/1
```

Task 1.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Port Security

SW2:

```
interface FastEthernet0/10
 switchport mode access
 switchport port-security
 switchport port-security maximum 4
 switchport port-security violation restrict
 switchport port-security mac-address 0050.7014.8ef0
 switchport port-security mac-address 00c0.144e.07bf
 switchport port-security mac-address 00d0.341c.7871
 switchport port-security mac-address 00d0.586e.b710
!
logging 191.1.7.100
```


Task 1.4 Breakdown

Layer 2 security based on source MAC address of a frame is controlled by *port security*. Port security allows you to define either specific MAC addresses that can send traffic into a port or how many MAC addresses can send traffic into a port. The first step in enabling port security is to set the port mode to access. This is accomplished by issuing the **`switchport mode access`** command. Port security is not supported on dynamic ports. Next, enable port security by issuing the **`switchport port-security`** interface command.

By default, port security only allows one MAC address to use a port. The above task states that four MAC address should be allowed entry. The task specifically lists their addresses. Therefore, the maximum allowed addresses must be increased by issuing the **`switchport port-security maximum [num]`** command. Next, the addresses are defined by issuing the **`switchport port-security mac-address [address]`** command.

Next, the task states that other hosts which try to access the port should be logged. By default the violate action of port security is *shutdown*. This means that the port it is sent to err-disabled state when either an insecure MAC is heard, or the maximum MAC addresses is exceeded. In addition to shutdown, restrict and protect are included as additional violate actions. When the violation mode is set to protect, traffic from MAC addresses that are not secure or are in excess of the maximum value is discarded. When violation is set to restrict the behavior is the same as protect, but a syslog message an SNMP trap is generated as well. Use the interface level command **`switchport port-security violation`** command to change the violation mode.

🕮 Deep Dive

- Do switches still need to learn MAC addresses on secured ports?

Task 1.4 Verification

Verify the port-security configuration:

```
Rack1SW2#show port-security interface fa0/10
Port Security              : Enabled
Port Status                : Secure-down
Violation Mode             : Restrict
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 4
Total MAC Addresses        : 4
Configured MAC Addresses   : 4
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 0000.0000.0000:0
Security Violation Count   : 0
```

Verify the configured secure MAC addresses:

```
Rack1SW2#show port-security interface fa0/10 address
          Secure Mac Address Table
------------------------------------------------------------------------
Vlan    Mac Address       Type                          Ports   Remaining
Age
                                                                    (mins)
----    -----------       ----                          -----   -----------
  10    0050.7014.8ef0    SecureConfigured              Fa0/10        -
  10    00c0.144e.07bf    SecureConfigured              Fa0/10        -
  10    00d0.341c.7871    SecureConfigured              Fa0/10        -
  10    00d0.586e.b710    SecureConfigured              Fa0/10        -
------------------------------------------------------------------------
Total Addresses: 4
```


Task 1.5 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Bridging and Switching > STP Portfast
Bridging and Switching > STP Portfast Default

```
SW2:
!
! This command is required soe that SW2 does not send BPDUs
! out of it portfast ports until it heard a BPDU first.
!
spanning-tree portfast bpdufilter default
!
interface FastEthernet0/10
 spanning-tree portfast
```

Task 1.5 Breakdown

The task wording directly points toward using the PortFast feature. However, by default, a portfast enabled port is still sending out BPDUs. This is a safety precaution to ensure an attached bridge may recognize the sending switch. If you want to completely block sending BPDUs out of the port, you should use the interface command **spanning-tree bpdufilter enable**. The only problem is that with this command, the port will stop both sending and receiving BPDUs.

If you want a "conditional" portfast configuration, where a bridge stops sending BPDUs out of portfast ports but resumes sending them when it receives a BPDU on the port you need the "Portfast in default BPDU Filter mode". This feature is enabled using the *global* command **spanning-tree portfast bpdufilter default** and applies to every new port configured in portfast mode. With this feature, every portfast port will refrain from sending BPDUs *until* it hears a BPDU first. After this, the port immediately loses its portfast status and behaves according to STP rules..

🕮 Deep Dive

- Is there another way to block BPDUs received on an interface?

Task 2.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

OSPF > OSPF over Broadcast Media
OSPF > OSPF over Non-Broadcast Media

R1:
```
router ospf 1
 router-id 150.1.1.1
 network 191.1.125.1 0.0.0.0 area 0
 neighbor 191.1.125.2
 neighbor 191.1.125.5
```

R2:
```
interface Serial0/0
 ip ospf priority 0
!
router ospf 1
 router-id 150.1.2.2
 network 191.1.125.2 0.0.0.0 area 0
```

R5:
```
interface Serial0/0/0
 ip ospf priority 0
!
router ospf 1
 router-id 150.1.5.5
 network 191.1.125.5 0.0.0.0 area 0
```


Task 2.1 Breakdown

As the Frame Relay section dictates that R1, R2, and R5 must use the main interface for their hub-and-spoke configuration, the default OSPF network type will be non-broadcast. Additionally, since this section dictates that the **ip ospf network** command cannot be used on any of these devices, the default of non-broadcast must remain. Therefore, R1 has been configured to specify its unicast neighbors, R2 and R5, and R2 and R5 have adjusted their OSPF priority value to remove participation in the DR/BDR election. As R1 is the only device on this segment that has a direct layer 2 connection to all endpoints of the network, it is mandatory that R1 be elected the DR.

Task 2.1 Verification

Verify OSPF network type (non-broadcast) and the DR for the segment:

```
Rack1R1#show ip ospf interface s0/0
Serial0/0 is up, line protocol is up
  Internet Address 191.1.125.1/24, Area 0
  Process ID 1, Router ID 150.1.1.1,Network Type NON_BROADCAST,Cost: 64
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 150.1.1.1, Interface address 191.1.125.1
```

Verify OSPF neighbors:

```
Rack1R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.2.2         0   FULL/DROTHER    00:01:46    191.1.125.2     Serial0/0
150.1.5.5         0   FULL/DROTHER    00:01:53    191.1.125.5     Serial0/0
150.1.3.3         0   FULL/  -        00:00:35    191.1.13.3      Serial0/1
```


Task 2.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

OSPF > OSPF Not-So-Stubby Areas
OSPF > OSPF Not-So-Stubby Area and Default Routing

R2:

```
!
! Block redistribution in the area and only send default route
!
router ospf 1
 area 27 nssa no-redistribution no-summary
```

SW1:

```
!
! SW1 redistributes its loopback into OSPF
!
router ospf 1
 area 27 nssa
 redistribute connected route-map CONNECTED_TO_OSPF subnets
!
route-map CONNECTED_TO_OSPF
 match interface Loopback0
```


Task 2.2 Breakdown

The above task states that SW1 does not require specific reachability information to the rest of the IGP domain, as its only connection out is through R2. As previously demonstrated, this can be accomplished by defining the area in question as a type of stub area. The next issue that must be addressed is which type of stub area to configure.

Stub Type	Keyword	LSAs	Default Injected
Stub	area x stub	1,2,3	YES
Totally Stubby	area x stub no-summary	1,2, default of 3	YES
Not-So-Stub	area x nssa	1,2,3,7	NO
Not-So-Totally-Stubby	area x nssa no-summary	1,2, default of 3, 7	YES

The task states that the only IGP route it should see is a default route generated by R2, the ABR. The only stub area type that does not automatically generate a default route into the area is the not-so-stubby area. However, a default route can be manually generated into the NSSA area by adding the **`default-originate`** keyword on to the end of the **`area [area] nssa`** statement. Therefore, the requirement of a default route alone does not narrow our choices. The keyword for the above ask is that SW1 should not see any *other* IGP routes except this default. This requirement implies that inter-area or external reachability information should not be injected into area 27. This narrows our choices down to two stub types, the *totally stubby area* and the *not-so-totally-stubby* area.

The Loopback 0 interfaces of all routers were injected into the OSPF domain by issuing the **`redistribute connected`** command. This implies that redistribution must be allowed into area 27. This further eliminates the stub area type of totally stubby, and leaves us with our last choice of not-so-totally-stubby.

The last two stipulations on this task give us a twist that has not been previously seen. The last two requirements state that SW1 should not see a specific route to R2's Loopback 0 network. As redistribution is allowed into a not-so-totally-stubby area, this route will be seen by SW1 unless additional configuration is performed. This prefix can be removed from SW1's routing table in a variety of ways. These include filtering the route out from the IP routing table with a distribute-list or a route-map, poisoning the distance of the prefix, or stopping the route from coming into the area by disallowing redistribution into the NSSA area on the ABR. The first two options cannot be used, as the requirement specifically denies their usage. Changing the distance of the prefix is a valid solution; however it was not the intended solution for the requirement.

The **no-redistribution** keyword on the end of the **area [area] nssa** statement is specifically designed to deal with the above scenario. When redistribution is performed on an OSPF device, the route is propagated into all areas unless it is manually blocked with a stub definition or filtering. This is also true of the ABR of an NSSA area. When a route is redistributed on the ABR of an NSSA, it also becomes an ASBR. This route is therefore propagated into the NSSA area as LSA 7 (N1 or N2 route), and as LSA 5 into all other areas. The **no-redistribution** keyword allows us to stop this default behavior. Although redistribution into the NSSA is still allowed, routes redistributed into the OSPF domain on the NSSA ABR itself are not propagated into the NSSA area. As in the above case, this behavior is advantageous.

Since SW1's only connection to the rest of the routing domain is through R2, it does not need specific routing information about other areas. Instead, this information can be replaced by a default route generated by R2. Therefore, SW1 does not require the amount of memory to hold the OSPF database as well as the IP routing table as other devices in the OSPF domain. This memory usage is further reduced by disallowing routes redistributed on R2 to go into area 27, as devices in area 27 will already have default reachability through R2.

🕮 Deep Dive

- Why is it mandatory to have forwarder address in type-5 LSAs translated from type-7 LSA?

Task 2.2 Verification

Check area 27 type:

```
Rack1SW1#show ip ospf | begin Area 27
    Area 27
        Number of interfaces in this area is 4
        It is a NSSA area
<output omitted>
```

Verify the routing table on SW1:

```
Rack1SW1#show ip route ospf
O*IA 0.0.0.0/0 [110/2] via 191.1.27.2, 00:00:28, FastEthernet0/14
```

Verify that the other OSPF routers still see SW1's Loopback0 prefix:

```
Rack1R2#show ip route ospf  | inc N2
O N2    150.1.7.0/24 [110/20] via 191.1.27.7, 00:11:10, FastEthernet0/0

Rack1R3#show ip route ospf  | inc E2
O E2    150.1.4.4/32 [110/20] via 191.1.34.4, 00:25:21, Serial1/0
O E2    150.1.1.0/24 [110/20] via 191.1.23.2, 00:25:21, Serial1/2
O E2    150.1.2.0/24 [110/20] via 191.1.23.2, 00:25:21, Serial1/3
O E2    150.1.7.0/24 [110/20] via 191.1.23.2, 00:09:33, Serial1/3
```


Task 2.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

OSPF > OSPF Conditional Default Routing

R3:

```
router ospf 1
 default-information originate always route-map CONDITION
!
ip prefix-list BB2 seq 5 permit 192.10.1.0/24
!
ip prefix-list BB3 seq 5 permit 204.12.1.0/24
!
route-map CONDITION permit 10
 match ip address prefix-list BB2
!
route-map CONDITION permit 20
 match ip address prefix-list BB3
```


Task 2.3 Breakdown

The above task dictates that R3 should originate a default route into the OSPF domain. However, a stipulation is placed on its generation of this default. This default should only be generated if its connections to either BB2 or BB3 are up. This type of stipulation is known as *conditional* advertisement.

To enable the conditional advertisement of a default route in OSPF, a route-map is added onto the **`default-information originate`** statement. If the route-map indicated is true, a default route is originated. If the route-map is false, a default route is not originated. In the above example the route-map CONDITION specifies that either the prefix 192.10.1.0/24 or 204.12.1.0/24 must exist in the IP routing table. If this condition is true, the default route is originated.

Pitfall

When the **`default-information originate`** statement has a conditional route-map attached to it, the condition must be met in order to originate a default regardless of whether the *always* keyword is included. If the above route-map CONDITION is not met no default will be generated even if the *always* keyword is added. Whena route-map is used, origination of the default route no longer depends on the existence of a default route in the routing table

Task 2.3 Verification

Verify that conditional advertisement works:

```
Rack1R3#debug ip ospf lsa-generation
Rack1R3#conf t
Rack1R3(config)#int Fa0/0
Rack1R3(config-if)#shutdown
Rack1R3(config)#interface Fa0/1
Rack1R3(config-if)#shutdown

OSPF: Generate external LSA 0.0.0.0, mask 0.0.0.0, type 5, age 3600,
metric 16777215, tag 1, metric-type 2, seq 0x80000002
OSPF: 0.0.0.0/0 type: 5 is already maxaged

Rack1R3(config-if)#no shutdown
OSPF: Generate external LSA 0.0.0.0, mask 0.0.0.0, type 5, age 0,
metric 1, tag 1, metric-type 2, seq 0x80000003

Rack1R3(config)#int Fa0/0
Rack1R3(config-if)#no shutdown
```

Note that the external LSA has been purged and then reinstalled.Next, verify that the default route is advertised into the OSPF domain:

```
Rack1R5#show ip route 0.0.0.0
Routing entry for 0.0.0.0/0, supernet
  Known via "ospf 1", distance 110, metric 1, candidate default path
  Tag 1, type extern 2, forward metric 128
  Last update from 191.1.125.2 on Serial0/0/0, 00:03:23 ago
  Routing Descriptor Blocks:
  * 191.1.125.2, from 150.1.3.3, 00:03:23 ago, via Serial0/0/0
      Route metric is 1, traffic share count is 1
      Route tag 1
```


Task 2.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following INE blog post:

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

R3:

```
router ospf 1
 redistribute rip subnets route-map RIP_TO_OSPF
!
router rip
 redistribute connected metric 1 route-map CONNECTED_TO_RIP
 redistribute ospf 1 metric 1
!
ip prefix-list R6_LOOPBACK0 seq 5 permit 150.1.6.0/24
!
route-map CONNECTED_TO_RIP permit 10
 match interface FastEthernet0/0 Loopback0 Serial1/2 Serial1/3
Serial1/0
!
route-map RIP_TO_OSPF permit 10
 match ip address prefix-list R6_LOOPBACK0
```

R6:

```
router rip
 version 2
 network 150.1.0.0
 network 204.12.1.0
 redistribute connected metric 1 route-map CONNECTED_TO_RIP
 no auto-summary
!
route-map CONNECTED_TO_RIP permit 10
 match interface Loopback0
```


Task 2.4 Breakdown

The task wording means that RIP should be redistributed into OSPF, but when RIP is redistributed into OSPF the only prefix that should be allowed is R6's Loopback 0 network. This is accomplished by matching R6's loopback in a prefix-list, then matching the prefix-list in a route-map, and using this route-map to filter the redistribution of RIP into OSPF.

Pitfall

R3's Loopback 0 interface has been advertised into the OSPF domain through redistribution. Although OSPF is redistributed into RIP, this does not imply that R3's Loopback 0 interface is redistributed into RIP. Indirect redistribution between two protocols cannot be accomplished on the same local devices. For example, suppose that protocol A is redistributed into protocol B. Protocol B is then redistributed into protocol C. This does not imply that protocol A was redistributed into protocol C. Instead, protocol A must be manually redistributed into protocol C to achieve the desired effect. This can be seen in the above output since R3's Loopback 0 network is redistributed as connected into the RIP domain.

Deep Dive

- What are the main tools for routing loop prevention in redistribution scenarios?
- How many type-5 LSAs is required in OSPF to redistribute 10 external prefixes?

Task 2.4 Verification

Verify that only R6's Loopback0 is redistributed into OSPF from RIP:

```
Rack1R1#show ip route ospf | include " 30\."
Rack1R1#show ip route ospf | include 150.1.6.0
O E2    150.1.6.0/24 [110/20] via 191.1.13.3, 00:03:08, Serial0/1
```

Next verify that we have connectivity inside the OSPF domain and can ping R6's Loopback0 from every OSPF router. Execute the following TCL script on routers R1 through R5. Note that script excludes the VRF interfaces from connectivity testing as those are to be configured later.

```
foreach i {
150.1.1.1
191.1.13.1
191.1.125.1
150.1.2.2
191.1.27.2
191.1.23.2
191.1.125.2
150.1.3.3
191.1.34.3
191.1.23.3
191.1.13.3
204.12.1.3
192.10.1.3
191.1.48.4
191.1.45.4
150.1.4.4
191.1.34.4
191.1.45.5
150.1.5.5
191.1.5.5
191.1.125.5
150.1.6.6
204.12.1.6
} { puts [exec "ping $i" ] }
```

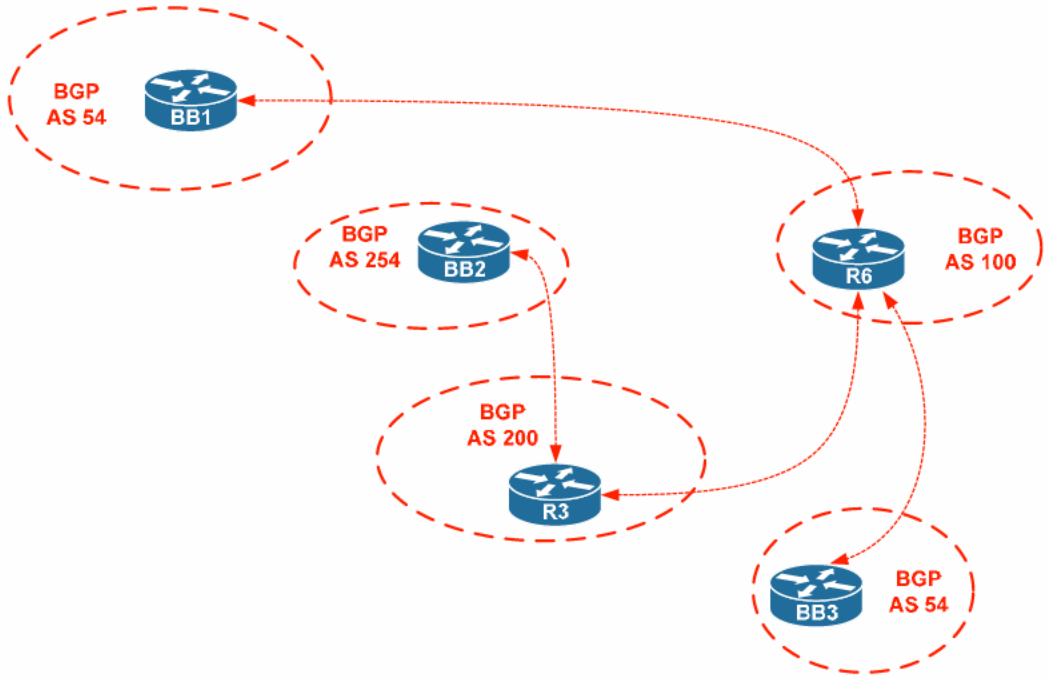

Task 2.5 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Filtering with Prefix Lists

Prior to starting the BGP section it is worth creating a BGP peering diagram, as BGP has been pre-configured for you. In some cases this wont be necessary, since BGP scenarios may appear to be simple, e.g. filtering at BGP network edge. The below is the relevant BGP digram for this scenario. Notice that R4 and R5 are not included, because they belong to an isolated BGP AS used to provide MPLS VPN services exclusively.



R6:

```
router bgp 100
 neighbor 204.12.1.254 route-map FROM_BB3 in
 neighbor 54.1.3.254 route-map FROM_BB1 in
!
route-map FROM_BB3 permit 10
 match ip address prefix-list SLASH_20_AND_UNDER
!
route-map FROM_BB1 permit 10
 match ip address prefix-list SLASH_20_AND_UNDER
!
ip prefix-list SLASH_20_AND_UNDER seq 5 permit 0.0.0.0/0 le 20
```

Task 2.5 Breakdown

Unlike the IP access-list, which was designed to match traffic, the IP *prefix-list* was designed specifically with network reachability information in mind. Prefix-lists are used to match on prefix (network) and prefix-length (subnet mask) pairs. The prefix-list has dual syntax meanings. The syntax is straightforward once you understand what it means; unfortunately the prefix-list is very sparsely documented.

Normal prefix-list syntax is as follows:

ip prefix-list [*name*] [permit | deny] [*prefix*]/[*len*]

Where *name* is any name or number, *prefix* is the exact routing prefix (network), and *len* is the exact prefix-length (subnet mask). Take the following examples:

```
ip prefix-list LIST permit 1.2.3.0/24
```

The above is an exact match for the network 1.2.3.0 with the exact subnet mask of 255.255.255.0. This list does not match 1.2.0.0/24, nor does it match 1.2.3.4/32, nor anything in between.

```
ip prefix-list LIST permit 0.0.0.0/0
```

The above is an exact match for the network 0.0.0.0 with the exact subnet mask of 0.0.0.0. This is used to match a default route.

Typical confusion about the prefix-list comes into play when the keywords "GE" (greater than or equal to) and "LE" (less than or equal to) are added to the prefix-list. This is due to the fact that the "len" value changes meaning when the GE or LE keywords are used.

This alternate syntax is as follows:

ip prefix-list [*name*] [permit | deny] [*prefix*]/[*len*] ge [*min_length*] le [*max_length*]

Where *name* is any name or number, *prefix* is the routing prefix to be checked against, *len* is the amount of bits starting from the most significant (left most) to check, *min_length* is the minimum subnet mask value, and *max_length* is the maximum subnet mask value.

When using the GE and LE values, the following condition must be satisfied:

len < GE <= LE

The above syntax, while confusing at first, simply means that a range of addresses will be matched based on the prefix and the subnet mask range.

Take the following examples:

```
ip prefix-list LIST permit 1.2.3.0/24 le 32
```

The above syntax means that the first 24 bits of the prefix 1.2.3.0 must match. Additionally, the subnet mask must be less than or equal to 32.

```
ip prefix-list LIST permit 0.0.0.0/0 le 32
```

The above syntax means that zero bits of the prefix must match. Additionally, the subnet mask must be less than or equal to 32. Since all networks have a subnet mask less than or equal to 32, and no bits of the prefix are matched, this statement equates to an explicit permit any.

```
ip prefix-list LIST permit 10.0.0.0/8 ge 21 le 29
```

The above syntax means that the first 8 bits of the prefix 10.0.0.0 must match. Additionally, the subnet mask is between 21 and 29 inclusive.

The above task states that prefixes with a subnet mask greater than /20 should not be accepted from AS 54. Therefore, zero bits of the actual prefix need to be checked. Instead, it must only be true that the subnet mask is less than or equal to /20. The syntax for this list is therefore as follows:

```
ip prefix-list SLASH_20_AND_UNDER seq 5 permit 0.0.0.0/0 le 20
```

Note

A prefix-list cannot be used to match on arbitrary bit patterns like an access-list can. Prefix-lists cannot be used to check if a number is even or odd, nor check if a number is divisible by 15, etc... Bit checking in a prefix-list is sequential, and must start with the most significant (leftmost) bit.

Deep Dive

- Construct a prefix list that rejects all class A subnets with the prefix lengths from /17 to /24

Task 2.5 Verification

Verify that we actually receive only /20 prefixes or shorter. Start by verifying BGP configuration:

```
Rack1R6#show ip protocols | begin bgp
Routing Protocol is "bgp 100"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  IGP synchronization is disabled
  Automatic route summarization is disabled
  Neighbor(s):
    Address          FiltIn FiltOut DistIn DistOut Weight RouteMap
    54.1.3.254                                            SET_WEIGHT
    191.1.46.4
    204.12.1.3
    204.12.1.254                                          FROM_BB3
  Maximum path: 1
<output omitted>

Rack1R6#debug ip bgp updates
Rack1R6#clear ip bgp 204.12.1.254 soft in
BGP(0): 204.12.1.254 rcvd UPDATE w/ attr: nexthop 204.12.1.254, origin
i, metric 0, path 54
BGP(0): 204.12.1.254 rcvd 28.119.16.0/24 -- DENIED due to: route-map;
BGP(0): 204.12.1.254 rcvd 28.119.17.0/24 -- DENIED due to: route-map;
BGP(0): 204.12.1.254 rcv UPDATE w/ attr: nexthop 204.12.1.3, origin ?,
originator 0.0.0.0, path 54 100 200 254, community , extended community
BGP(0): 204.12.1.254 rcv UPDATE about 205.90.31.0/24 -- DENIED due to:
AS-PATH contains our own AS;
BGP(0): 204.12.1.254 rcv UPDATE about 220.20.3.0/24 -- DENIED due to:
AS-PATH contains our own AS;
BGP(0): 204.12.1.254 rcv UPDATE about 222.22.2.0/24 -- DENIED due to:
AS-PATH contains our own AS;
BGP(0): 204.12.1.254 rcvd UPDATE w/ attr: nexthop 204.12.1.254, origin
i, path 54
<output omitted>
```

Finally verify that the prefix-list has only one line:

```
Rack1R6#show running-config | include ip prefix-list
ip prefix-list SLASH_20_AND_UNDER seq 5 permit 0.0.0.0/0 le 20
```


Task 2.6 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Aggregation

R3:
```
router bgp 200
 redistribute static
!
ip route 150.1.0.0 255.255.240.0 Null0
ip route 191.1.0.0 255.255.0.0 Null0
```

Task 2.6 Breakdown

There are four (previously three) ways to originate prefixes in BGP. The first is to use the **network** statement. Secondly, a route may be originated through the **redistribute** statement. Next, the **aggregate-address** command can originate a summary route based on more specific routes in the BGP table. A new method of BGP route generation is the **inject-map**, and will be covered in later scenarios.

By creating two static routes that point to Null0 and redistributing them into BGP, traffic that reaches R3 which is destined for a subset of these networks will only be forwarded if there is a more specific subnet installed in the IP routing table. Many protocols automatically generate a summary route to Null0 when aggregation is performed. This behavior is the desired behavior, and would rarely be modified for any practical reason.

🕮 Deep Dive

- Is it safe to perform route redistribution into BGP? Is there a safer way to advertise prefixes?

Task 2.6 Verification

Verify that R3 originates the summaries:

```
Rack1R3#show ip bgp regexp ^$
BGP table version is 26, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 150.1.0.0/20     0.0.0.0                  0         32768 ?
*> 191.1.0.0        0.0.0.0                  0         32768 ?
*>
```

Verify for example that R6 receives it:

```
Rack1R6#show ip bgp 191.1.0.0
BGP routing table entry for 191.1.0.0/16, version 20
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
        1
  200
    204.12.1.3 from 204.12.1.3 (150.1.3.3)
      Origin incomplete, metric 0, localpref 100, valid, external, best

Rack1R6#show ip bgp 150.1.0.0
BGP routing table entry for 150.1.0.0/20, version 19
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
        1
  200
    204.12.1.3 from 204.12.1.3 (150.1.3.3)
      Origin incomplete, metric 0, localpref 100, valid, external, best
```


Task 2.7 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Dampening
BGP > BGP Dampening with Route Map

R6:

```
router bgp 100
 bgp dampening route-map DAMPENING
!
ip prefix-list AS54_CUSTOMERS seq 5 permit 112.0.0.0/8
ip prefix-list AS54_CUSTOMERS seq 10 permit 113.0.0.0/8
!
route-map DAMPENING permit 10
 match ip address prefix-list AS54_CUSTOMERS
 set dampening 15 750 2000 60
```

☞ Quick Note
Default values. Route-map requires values to be set.

Task 2.7 Breakdown

BGP route flap dampening (damping is the term used outside Cisco) is the process of suppressing consistently unstable routes from being used or advertised to BGP neighbors. Dampening is (and must be) used to minimize the amount of route recalculation performed in the global BGP table as a whole.

To understand dampening, the following terms must first be defined:

Penalty: Every time a route flaps, a penalty value of 1000 is added to the current penalty. All prefixes start with a penalty of zero.

Half-life: Configurable time it takes the penalty value to reduce by half. Defaults to 15 minutes.

Suppress Limit: Threshold at which a route is suppressed if the penalty exceeds. Defaults to 2000.

Reuse Limit: Threshold at which a suppressed route is unsuppressed if the penalty drops below. Defaults to 750.

Max Suppress: Maximum time a route can be suppressed if it has been stable. Defaults to four times the half-life value.

Each time a route flaps (leaves the BGP table and reappears), it is assigned a penalty of 1000. As soon as this occurs, the penalty of the route starts to decay based on the half-life timer. As the penalty increases, so does the rate of decay. For example, after a single flap, it will take 15 minutes for a prefix to reduce its penalty to 500.

Once the penalty of a prefix exceeds the suppress limit, the prefix is suppressed. A suppressed prefix cannot be used locally or advertised to any BGP peer. Once the penalty decay has resulted in the penalty decreasing below the reuse limit, the prefix is unsuppressed.

Lastly, the max-suppress timer dictates the maximum amount of time a prefix can be suppressed if it has been stable. This value is useful if a number of flaps have occurred in a short period of time, after which the route has been stable.

To enable BGP route flap dampening, simply enter the command **`bgp dampening`** under the BGP process.

🕮 Deep Dive

- Why is it important to differentiate dampening options based on prefix types?

Task 2.7 Verification

Verify dampening settings:

```
Rack1R6#show ip bgp dampening parameters
 dampening 15 750 2000 60 (route-map DAMPENING 10)
  Half-life time      : 15 mins       Decay Time       : 2320 secs
  Max suppress penalty: 12000         Max suppress time: 60 mins
  Suppress penalty    :  2000         Reuse penalty    : 750

Rack1R6#show route-map DAMPENING
route-map DAMPENING, permit, sequence 10
  Match clauses:
    ip address prefix-lists: AS54_CUSTOMERS
  Set clauses:
    dampening 15 750 2000 60
  Policy routing matches: 0 packets, 0 bytes
```

In general, you may want to do a final check to verify that you can reach your BGP networks from the various devices. Due to the summaries generated in task 3.4, and the earlier default route originated into OSPF by R3, you will have reachability to the BGP networks, even from non-BGP devices, such as the switches. For R6, a ping will need to be sourced from the loopback interface as shown below, since the interface networks are not being advertised into BGP.

```
foreach i {
112.0.0.1
113.0.0.1
114.0.0.1
115.0.0.1
116.0.0.1
117.0.0.1
118.0.0.1
119.0.0.1
205.90.31.1
222.22.2.1
} { ping $i source lo0 }
```


Task 3.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IPV6 > Link Local Addressing
IPv6 > Unique Local Addressing
IPv6 > Global Aggregatable Addressing

R1:
```
ipv6 unicast-routing
!
interface Serial0/0
 ipv6 address 2001:CC1E:1:125::1/64
 frame-relay map ipv6 2001:CC1E:1:125::2 102 broadcast
 frame-relay map ipv6 2001:CC1E:1:125::5 105 broadcast
```

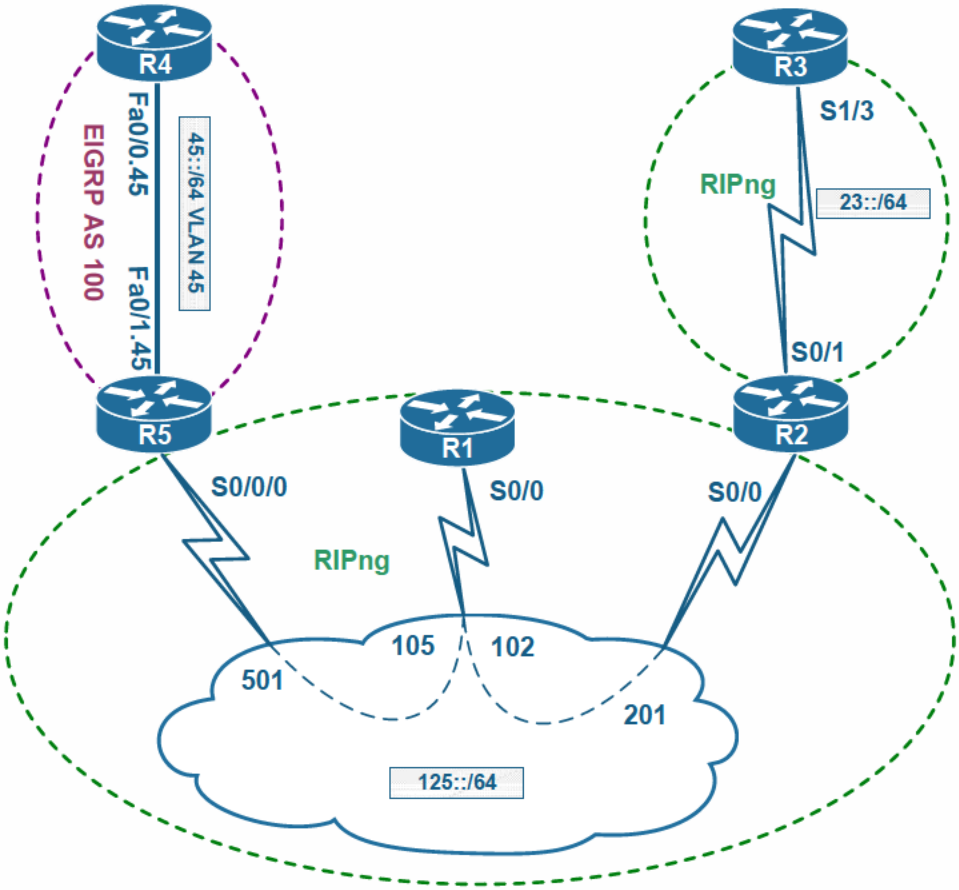
R2:
```
ipv6 unicast-routing
!
interface Serial0/0
 ipv6 address 2001:CC1E:1:125::2/64
 frame-relay map ipv6 2001:CC1E:1:125::1 201 broadcast
 frame-relay map ipv6 2001:CC1E:1:125::5 201
!
interface Serial0/1
 ipv6 address 2001:CC1E:1:23::2/64
```

R3:
```
ipv6 unicast-routing
!
interface Serial1/3
 ipv6 address 2001:CC1E:1:23::3/64
```

R5:
```
ipv6 unicast-routing
!
interface FastEthernet0/0
 ipv6 address 2001:CC1E:1:5::5/64
!
interface Serial0/0/0
 ipv6 address 2001:CC1E:1:125::5/64
 frame-relay map ipv6 2001:CC1E:1:125::1 501 broadcast
 frame-relay map ipv6 2001:CC1E:1:125::2 501
```


Task 3.1 Breakdown

It may be helpful drawing a small IPv6 diagram in some cases. In this scenario, IPv6 topology spans R1, R2, R3, R4 and R5 and includes NBMA and point-to-point links. If you look in the IPv6 scenarios that follow, you will notice that RIPng and EIGRPv6 are used as routing protocols on these linsk.



Notice that we have omitted the major prefix 2001:CC1E:X: from the diagram to ease address representation.

Task 3.1 Verification

Check IPv6 address assignment using the following command (repeat on every device to validate your settings:

```
Rack1R1#show ipv6 interface  brief
FastEthernet0/0            [administratively down/down]
Serial0/0                  [up/up]
    FE80::20D:BDFF:FEF6:2F80
    2001:CC1E:1:125::1
Serial0/1                  [up/up]
Loopback0                  [up/up]

Rack1R2#show ipv6 interface brief
FastEthernet0/0            [up/up]
Serial0/0                  [up/up]
    FE80::211:92FF:FE0E:5300
    2001:CC1E:1:125::2
Serial0/1                  [up/up]
    FE80::211:92FF:FE0E:5300
    2001:CC1E:1:23::2

Loopback0                  [up/up]

Rack1R3#show ipv6 interface brief
<output omitted>
Serial1/3                  [up/up]
    FE80::250:73FF:FE1C:7761
    2001:CC1E:1:23::3
<output omitted>
```

Test connectivity at the link level:

```
Rack1R3#ping ipv6 2001:CC1E:1:23::3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:23::3, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/0 ms
```



```
Rack1R5#ping 2001:CC1E:1:125::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:125::1, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/61 ms

Rack1R5#ping 2001:CC1E:1:125::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:125::2, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 160/167/185
ms
```

Task 3.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IPv6 > RIPng
IPv6 > RIPng over NBMA
IPv6 > RIPng Default Routing

R1:

```
!
! Enable RIPng and map link-local addresses
!
interface Serial0/0
 ipv6 rip RIPng enable
 ipv6 address FE80::1 link-local
 frame-relay map ipv6 FE80::5 105
 frame-relay map ipv6 FE80::2 102
!
ipv6 router rip RIPng
 no split-horizon
```

R2:

```
!
! Enable RIPng and map link-local addresses
!
interface Serial0/0
 ipv6 rip RIPng enable
 ipv6 address FE80::2 link-local
 frame-relay map ipv6 FE80::1 201
 frame-relay map ipv6 FE80::5 201
!
interface Serial0/1
 ipv6 rip RIPng enable
```


R3:

```
!
! Enable RIPng and send default information only
!
interface Serial1/3
 ipv6 rip RIPng enable
 ipv6 rip RIPng default-information only
!
!
interface Loopback100
 ipv6 address 2001:220:20:3::1/64
 ipv6 rip RIPng enable
!
interface Loopback100
 ipv6 address 2001:222:22:2::1/64
 ipv6 rip RIPng enable
!
interface Loopback100
 ipv6 address 2001:205:90:31::1/64
  ipv6 rip RIPng enable
```

R5:

```
interface FastEthernet0/0
 ipv6 rip RIPng enable
!
interface Serial0/0/0
 ipv6 rip RIPng enable
 ipv6 address FE80::5 link-local
 frame-relay map ipv6 FE80::1 501
 frame-relay map ipv6 FE80::2 501
```


Task 3.2 Breakdown

A small trick in this scenario is remembering to disable split-horizon for RIPng process, as otherwise R1 wont be able to relay RIPng updates from spoke to spoke. This scenario also demonstrates how to originate an IPv6 default route via RIPng with the interface level command **`ipv6 rip [process-id] default-information [originate | only]`**. When the *only* keyword is used, all other more specific networks are suppressed in RIPng advertisements on the interface. As seen in the below output, an IPv6 default route is expressed as the prefix ::/0.

```
Rack1R2#show ipv6 route rip
IPv6 Routing Table - 11 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
    O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
R   ::/0 [120/2]
     via FE80::250:73FF:FE1C:7761, Serial0/1
R   2001:CC1E:1:5::/64 [120/3]
     via FE80::204:27FF:FEB5:2F60, Serial0/0
```

🕮 Deep Dive

- Is there a way to overcome the split-horizon problem without disabling split-horizon feature globally?

Task 3.2 Verification

Verify that R3 sends only a default route to R2:

```
Rack1R2#show ipv6 route rip
IPv6 Routing Table - 8 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
R   ::/0 [120/2]
     via FE80::250:73FF:FE1C:7761, Serial0/1
R   2001:CC1E:1:5::/64 [120/3]
     via FE80::1, Serial0/0
```

Verify that R1 has split-horizon turned off for IPv6 RIPng:

```
Rack1R1#show ipv6 rip
RIP process "RIPng", port 521, multicast-group FF02::9, pid 133
     Administrative distance is 120. Maximum paths is 16
     Updates every 30 seconds, expire after 180
     Holddown lasts 0 seconds, garbage collect after 120
     Split horizon is off; poison reverse is off
     Default routes are not generated
     Periodic updates 8, trigger updates 3
  Interfaces:
    Serial0/0
  Redistribution:
    None
```


Task 3.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IPv6 > EIGRPv6
IPv6 > EIGRPv6 Summarization

R4:

```
ipv6 unicast-routing
!
ipv6 router eigrp 100
 no shutdown
!
interface Loopback101
 ipv6 address 2001:CC1E:1:444::4/64
 ipv6 eigrp 100
!
interface Loopback102
 ipv6 address 2001:CC1E:1:454::4/64
 ipv6 eigrp 100
!
interface Loopback103
 ipv6 address 2001:CC1E:1:484::4/64
 ipv6 eigrp 100
!
interface FastEthernet 0/0.45
 ipv6 address 2001:CC1E:1:45::4/64
 ipv6 eigrp 100
 ipv6 summary-address eigrp 100 2001:CC1E:1:400::/56
```



```
R5:
ipv6 unicast-routing
!
ipv6 router eigrp 100
 no shutdown
 redistribute rip RIPng metric 1 1 1 1 1
!
ipv6 router rip RIPng
 redistribute eigrp 100 metric 1
!
interface FastEthernet 0/1.45
 ipv6 address 2001:CC1E:1:45::5/64
 ipv6 eigrp 100
```

Task 3.3 Breakdown

The only special requirement in this task is summarizing the prefixes advertised at R4. The summarization procedure follows the same rules as for IPv4 prefixes. You may consult the INE blog post:

http://blog.ine.com/2010/03/17/a-simple-ipv4-prefix-summarization-procedure/

For information about procedure that makes summarizing discontiguous prefixes easier. Even though the blog post refers to IPv4 prefixes mainly, it perfecty applies to IPv6 addresses based on hexadecimal arithmetics.

Redistribution in this task is straightforward and does not require any special precautions as there are no multiple points of mutual redistribution.

Task 3.3 Verification

Check EIGRP neighbors; verify that R4 receives the default route from RIPng domain and the subnet for the link between R2 and R3:

```
Rack1R4#show ipv6 eigrp neighbors
IPv6-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   Link-local address:  Fa0/0.45         13 00:28:38    1    300  0  14
    FE80::213:1AFF:FE68:B04D

Rack1R4#show ipv6 route eigrp
IPv6 Routing Table - Default - 12 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
EX  ::/0 [170/2560002816]
     via FE80::213:1AFF:FE68:B04D, FastEthernet0/0.45
EX  2001:CC1E:1:23::/64 [170/2560002816]
     via FE80::213:1AFF:FE68:B04D, FastEthernet0/0.45
D   2001:CC1E:1:400::/56 [5/128256]
     via Null0, directly connected
```

Verify that R5 receives only the summary EIGRPv6 from R4:

```
Rack1R5#show ipv6 route eigrp
IPv6 Routing Table - Default - 10 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
D   2001:CC1E:1:400::/56 [90/156160]
     via FE80::215:62FF:FE41:FD9C, FastEthernet0/1.45
```


Verify that R5 redistributes into RIPng its connected Fa0/1.45 running EIGRPv6:

```
Rack1R1#show ipv6 route rip
IPv6 Routing Table - 9 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   ::/0 [120/3]
     via FE80::2, Serial0/0
R   2001:CC1E:1:5::/64 [120/2]
     via FE80::5, Serial0/0
R   2001:CC1E:1:23::/64 [120/2]
     via FE80::2, Serial0/0
R   2001:CC1E:1:45::/64 [120/2]
     via FE80::5, Serial0/0
R   2001:CC1E:1:400::/56 [120/2]
     via FE80::5, Serial0/0

Rack1R1#ping 2001:CC1E:1:45::4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:45::4, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/61 ms
```


Task 4.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

MPLS VPN > PE-CE Routing with EIGRP
Security > Traffic Filtering with Extended Access-Lists

R6:

```
!
! Configure the VRF routing process
!
router eigrp 1
eigrp router-id 150.1.6.6
 no auto-summary
 address-family ipv4 vrf VPN_A
 autonomous-system 10
 network 54.1.1.6 0.0.0.0
!
! Only permit one host to send us EIGRP packets
!
ip access-list extended EIGRP_FROM_BB1_ONLY
 permit eigrp host 54.1.1.254 any
 deny eigrp any any
 permit ip any any
!
! Enable authentication
!
key chain EIGRP
 key 1
  key-string CISCO
!
interface Serial0/0/0.1
 ip access-group EIGRP_FROM_BB1_ONLY in
 ip authentication key-chain eigrp 10 EIGRP
 ip authentication mode eigrp 10 md5
```


Task 4.1 Breakdown

This is a classic EIGRP configuration scenario, with only difference that it is now VRF specific. The only confusing part could be the requirement to get rid of "Neighbor Not on Common Subnet" syslog message. Such messages typically signify the fact that there are multiple IP networks on the same NBMA/broadcast segment and they don't match. In our case, this is caused by the fact that R6 learns multiple PVCs from the frame-relay switch and they all by defaut get assigned to the physical interface. This makes R6 receive all broadcast packets sent by BB1, every one being sent from a different source IP address. You may discover that using the command **`debug ip packet`** (don't forget to use an access-list to filter out unwanted traffic).

Based on the above observation, a solution to make R6 ignore the uninteresting EIGRP packet is either to move the IP subnet to a separate subinterface with only a single PVC mapped or apply a packet filter dropping all EIGRP packets but needed. Since normally changing interfaces is not allowed in the CCIE lab, we chose the second option.

🕮 Deep Dive

- Will EIGRP detect subnet mask discrepancy based on the HELLO packet exchange?

Task 4.1 Verification

Verify the EIGRP neighbors and EIGRP routes:

```
Rack1R6#show ip eigrp vrf VPN_A neighbors
IP-EIGRP neighbors for process 10
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q
Seq
                                            (sec)         (ms)
Cnt Num
0   54.1.1.254              Se0/0/0.1           11 20:45:20   25   200
0  140

Rack1R6#show ip route vrf VPN_A eigrp
D    200.0.0.0/24 [90/2297856] via 54.1.3.254, 00:03:16, Serial0/0/0
D    200.0.1.0/24 [90/2297856] via 54.1.3.254, 00:03:16, Serial0/0/0
D    200.0.2.0/24 [90/2297856] via 54.1.3.254, 00:03:16, Serial0/0/0
D    200.0.3.0/24 [90/2297856] via 54.1.3.254, 00:03:16, Serial0/0/0
```


Task 4.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

MPLS VPN > PE-CE Routing with OSPF
MPLS VPN > OSPF Sham Link

R4:

```
router bgp 100
 address-family ipv4 vrf VPN_B
  network 150.1.44.44 mask 255.255.255.255
!
router ospf 100 vrf VPN_B
 area 90 sham-link 150.1.44.44 150.1.55.55
 redistribute bgp 100 subnets
!
interface FastEthernet 0/0.45
 mpls ip
!
router bgp 100
 address-family ipv4 vrf VPN_B
  redistribute ospf 100
!
interface Serial0/0/0
 ip ospf cost 10000
```

R5:

```
router bgp 100
 address-family ipv4 vrf VPN_B
  network 150.1.55.55 mask 255.255.255.255
!
router ospf 100 vrf VPN_B
 area 90 sham-link 150.1.55.55 150.1.44.44
 redistribute bgp 100 subnets
!
interface FastEthernet 0/1.45
 mpls ip
!
router bgp 100
 address-family ipv4 vrf VPN_B
  redistribute ospf 100
```

SW3 & SW4:

```
interface Vlan 109
 ip ospf cost 9999
```


Task 4.2 Breakdown

In short, the scenario looks as following: R4 and R5 are the PE routers for the mentioned VPN. SW3 and SW4 use the VRF-lite feature to isolate the customer topology. By default, this situation results in the following:

1. R4 and R5 receive the remote end OSPF routes in MP-BGP and re-originate them to their CE routers as OSPF inter-area prefixes.
2. SW3 and SW4 by default will ignore the inter-ares prefixes learned from R4 and R5, because they have the Down bit set.

A typical solution to this problem would be enabling VRF lite capability in OSPF processes for SW3 and SW4 or change OSPF domain IDs in R4/R5 to onvert the OSPF IA prefixes into external routers.

However, there is an additional requirement in the task that changes the whole picture. It requires that SW3 and SW4 prefer reaching each other over the SP core link between R4 and R5. To accomplish this, we need to run a OSPF sham-link between R4 and R5 in the same area the SW3 and SW4 belong to. As soon as sham link is configured, the routes are perceived by R4 and R5 as intra-area, even though BGP forwarding information is still used. This automatically eliminates the above problem of SW3/SW4 rejecting the inter-area prefixes. Just don't forget to increase the cost of the link connecting SW3 and SW4 to make sure the sham-link is being preferred for OSPF routing.

🕮 Deep Dive

- What can you use in place of a OSPF sham-link?

Task 4.2 Verification

Verify that SW3 sees SW4 and R4 as neighbors:

```
Rack1SW3#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.10.10     1     FULL/BDR        00:00:38    191.1.109.10    Vlan109
150.1.44.44     1     FULL/DR         00:00:35    191.1.49.4      Vlan49

Rack1SW3#show ip route vrf VPN_B ospf
     191.1.0.0/24 is subnetted, 3 subnets
O       191.1.50.0 [110/3] via 191.1.49.4, 00:04:18, Vlan49
     150.1.0.0/16 is variably subnetted, 4 subnets, 2 masks
O E2    150.1.55.55/32 [110/1] via 191.1.49.4, 00:04:18, Vlan49
O E2    150.1.44.44/32 [110/1] via 191.1.49.4, 00:04:18, Vlan49
O       150.1.10.10/32 [110/4] via 191.1.49.4, 00:04:18, Vlan49
```

Check the routing table on SW4 and confirm that it prefers reaching SW3's prefix via R5:

```
Rack1SW4#show ip route vrf VPN_B ospf
     191.1.0.0/24 is subnetted, 4 subnets
O       191.1.49.0 [110/3] via 191.1.50.5, 00:38:11, Vlan50
     150.1.0.0/16 is variably subnetted, 4 subnets, 2 masks
O E2    150.1.55.55/32 [110/1] via 191.1.50.5, 00:38:11, Vlan50
O E2    150.1.44.44/32 [110/1] via 191.1.50.5, 00:38:11, Vlan50
O       150.1.9.9/32 [110/4] via 191.1.50.5, 00:38:11, Vlan50
```

Use the traceroute command to validate routing paths inside the VPN:

```
Rack1SW3#traceroute vrf VPN_B 150.1.10.10

Type escape sequence to abort.
Tracing the route to 150.1.10.10

  1 191.1.49.4 9 msec 0 msec 0 msec
  2 191.1.50.5 0 msec 0 msec 0 msec
  3 191.1.50.10 0 msec *  0 msec

Rack1SW4#traceroute vrf VPN_B 150.1.9.9

Type escape sequence to abort.
Tracing the route to 150.1.9.9

  1 191.1.50.5 0 msec 0 msec 0 msec
  2 191.1.49.4 0 msec 0 msec 0 msec
  3 191.1.49.9 0 msec *  0 msec
```


Just for completeness, check the sham-link status between R4 and R5:

```
Rack1R4#show ip ospf sham-links
Sham Link OSPF_SL0 to address 150.1.55.55 is up
Area 90 source address 150.1.44.44
  Run as demand circuit
  DoNotAge LSA allowed. Cost of using 1 State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40,
    Hello due in 00:00:05
    Adjacency State FULL (Hello suppressed)
    Index 2/2, retransmission queue length 0, number of retransmission
0
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 0, maximum is 0
    Last retransmission scan time is 0 msec, maximum is 0 msec

Rack1R4#show ip ospf sham-links
Sham Link OSPF_SL0 to address 150.1.55.55 is up
Area 90 source address 150.1.44.44
  Run as demand circuit
  DoNotAge LSA allowed. Cost of using 1 State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40,
    Hello due in 00:00:05
```

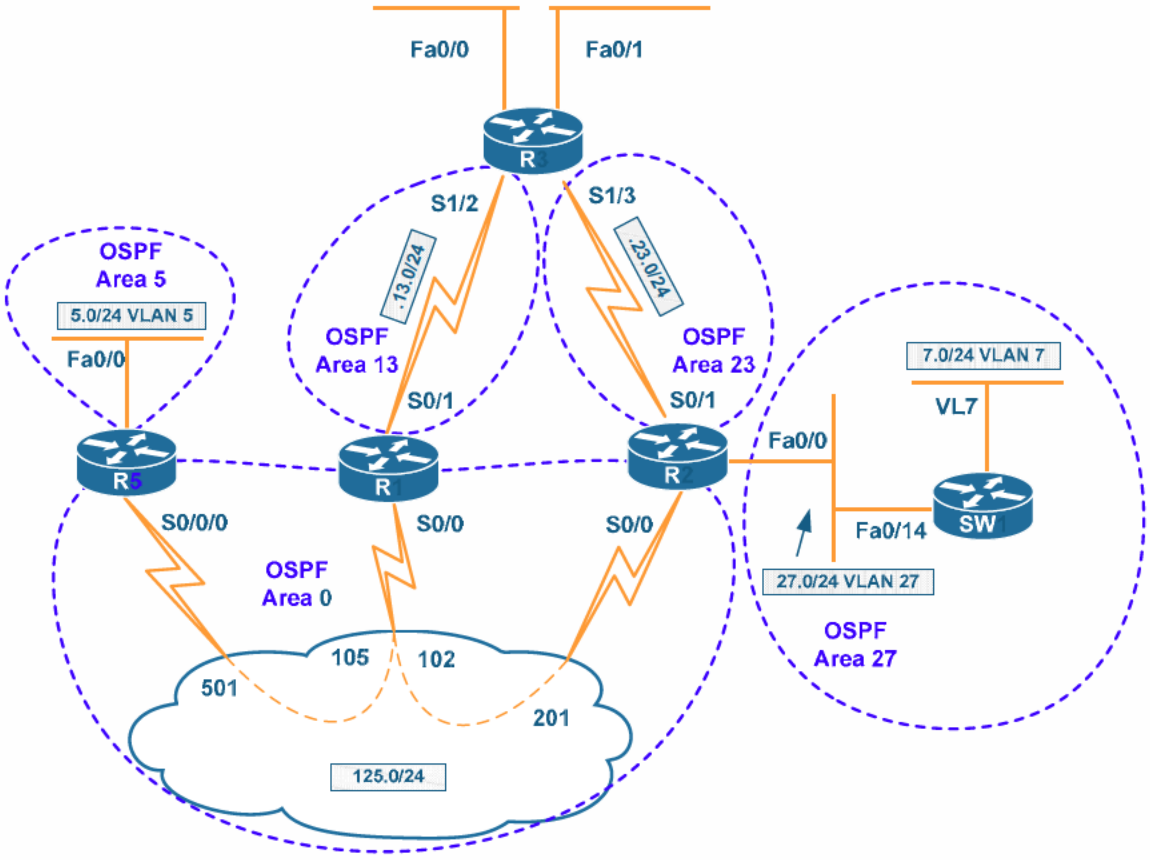

Task 5.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

`Multicast > Stub Multicast Routing & IGMP Helper`

Below is the existing multicast topology that you may discover using the show ip pim neighbor and show ip pim interface commands. It seems that all multicast links belong to the same IGP domain, which simplifies multicast configuration.



```
R1:
interface Serial0/0
 ip pim neighbor-filter 1
!
access-list 1 deny 191.1.125.5
access-list 1 permit any

R5:
interface FastEthernet0/0
 ip pim dense-mode
 ip igmp helper-address 191.1.125.1
```

Task 5.1 Breakdown

This configuration is called multicast stub routing. With multicast stub routing, a stub router will not be allowed to become a PIM neighbor. This is accomplished by using the **ip pim neighbor-filter** interface command. The **ip pim neighbor-filter** command takes an access-list as an option. The access-list should deny the IP addresses of the neighboring multicast devices that should not become PIM neighbors and permit all other IP addresses. Another option could be to use the reverse logic and permit only the IP addresses that are allowed to become PIM neighbors.

Since R5 will not form a PIM neighbor relationship, R5 will need to proxy for multicast clients connected to its FastEthernet0/0 interface by forwarding their IGMP host reports and IGMP leave messages to R1.

🕮 Deep Dive

- Can you implement Stub Multicast routing in multi-hop fashion?

Task 5.1 Verification

Verify the PIM neighbors on R1 (note that R5 does not show up):

```
Rack1R1#show ip pim neighbor
PIM Neighbor Table
Neighbor      Interface          Uptime/Expires    Ver   DR
Address                                                  Prio/Mode
191.1.13.3    Serial0/1          00:11:03/00:01:30 v2    1 / S
```

At the same time, R5 sees R1 as PIM neighbor:

```
Rack1R5#show ip pim neighbor
PIM Neighbor Table
Neighbor      Interface          Uptime/Expires    Ver   DR
Address                                                  Prio/Mode
191.1.125.1   Serial0/0          00:11:27/00:01:36 v2    1 / S
```

Lastly verify that R5 could actually forward IGMP joins to R1:

```
Rack1R5#show ip igmp interface FastEthernet0/0 | inc help
  IGMP helper address is 191.1.125.1
```


Task 5.2 Solution

R3:

```
interface FastEthernet0/1
 ip igmp version 1
```

Task 5.2 Breakdown

The default IGMP version is 2. The Cisco IOS supports IGMP versions 1, 2, and 3. To change the IGMP version, the **ip igmp version** interface command is needed.

The basic difference between IGMP version 1 and IGMP version 2 is that IGMP version 2 incorporated an IGMP leave message to allow a host to notify the multicast router that it does not want to receive traffic for a particular multicast group. In IGMP version 1, there is not an explicit IGMP leave message. When a host wants to leave a multicast group, it just stops sending IGMP reports for the IGMP queries sent by the multicast router.

🕮 Deep Dive

- What is the main difference between designated querier election in IGMPv1 and IGMPv2?

Note

There are three types of IGMP message that relate to multicast router and multicast client interaction.

1 = Host Membership Query
2 = Host Membership Report
3 = Leave Group

The IGMP query messages are sent by multicast enabled routers every 60 seconds (default) to all-hosts (224.0.0.1) in order to discover which multicast groups have hosts that would like to receive a particular multicast group.

The IGMP report messages are sent by hosts in response to IGMP queries reporting each multicast group to which they belong.

The IGMP leave messages are sent by hosts to notify a multicast router that it no longer wants to receive traffic for a particular multicast group. RFC 2236 (Internet Group Management Protocol, Version 2) states that the leave message is only mandatory if the host responded to the last IGMP query message for the group it wanted to leave. If the host was not the last to respond, RFC 2236 states that it is not mandatory to send an IGMP leave message.

Technically in IGMP version 2 there is a fourth message type, a version 1 membership report. This message is used for backward compatibility with IGMP version 1 clients.

Task 5.2 Verification

Verify the IGMP version on the interface:

```
Rack1R3#show ip igmp interface Fa0/1 | include version
  Current IGMP host version is 1
  Current IGMP router version is 1
```


Task 5.3 Solution

SW1:

```
!
interface Vlan7
 ip igmp static-group 225.25.25.25
```

Task 5.3 Breakdown

IGMP static groups could be used to implement static multicast routing, that does not rely on dynamic signaling for multicast distribution tree construction. This approach is very unscalable, but offers maximum security in the sense that no multicast groups other than configured could be joined in the network.

A well-known diference between using **ip igmp join** and **ip igmp static-group** is that the former command makes the router join the group and process all packets received for the group as opposed to simple data-plane switching.

Task 5.3 Verification

Verify that SW1 joined the multicast group:

```
Rack1SW1#show ip igmp groups
IGMP Connected Group Membership
Group Address    Interface            Uptime    Expires   Last Reporter
225.25.25.25     Vlan7                00:00:15  stopped   0.0.0.0
224.0.1.40       FastEthernet0/2      00:15:12  00:02:29  191.1.27.2
```


Task 6.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > VLAN Filters for Non-IP Traffic

SW3 and SW4:

```
vlan access-map NO_DEC-SPANNING 10
 action drop
 match mac address DEC-SPANNING
!
vlan access-map NO_DEC-SPANNING 20
 action forward
!
vlan filter NO_DEC-SPANNING vlan-list 363
!
mac access-list extended DEC-SPANNING
 permit any any dec-spanning
```

Task 6.1 Breakdown

This section is requiring a VACL to be configured within VLAN 363 that filters off any DECnet spanning tree BPDUs.

Ensure that there is an additional **vlan access-map** that forwards all other traffic or at least all other DECnet traffic. If this is not added, all DECnet traffic would be denied. The logic of the VACL is that as long as you do not deny a certain protocol or all protocols, it will not be affected by the VACL.

🕮 Deep Dive

- Can you think of a reason for using DEC STP variant with cisco bridges other than compatibility issues with legacy equipment?

Task 6.1 Verification

Verify the VLAN filter configuration:

```
Rack1SW3#show vlan access-map  NO_DEC-SPANNING
Vlan access-map "NO_DEC-SPANNING"  10
  Match clauses:
    mac address: DEC-SPANNING
  Action:
    drop
Vlan access-map "NO_DEC-SPANNING"  20
  Match clauses:
  Action:
    forward

Rack1SW3#show vlan filter access-map  NO_DEC-SPANNING
VLAN Map NO_DEC-SPANNING is filtering VLANs:
  363
```


Task 6.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Filtering Packets with Dynamic Access-Lists

R2:

```
username CLI password 0 CISCO
username TELNET password 0 CISCO
username TELNET autocommand access-enable timeout 5
!
ip access-list extended DYNAMIC1
 dynamic PERMIT_TELNET permit tcp any any eq telnet
 deny   tcp any host 191.1.27.7 eq telnet
 deny   tcp any host 191.1.7.7 eq telnet
 deny   tcp any host 191.1.77.7 eq telnet
 deny   tcp any host 191.1.177.7 eq telnet
 deny   tcp any host 150.1.7.7 eq telnet
 permit ip any any
!
interface Serial0/0
 ip access-group DYNAMIC1 in
!
interface Serial0/1
 ip access-group DYNAMIC1 in
!
line vty 0 4
 login local
```


Task 6.2 Breakdown

The task calls for the use of legacy Lock and Key feature, which has been superseded by authentication proxy and identity-based network solutions, e.g. 802.1X or VPN tunneling.

However, the Dynamic ACLs (other name for Lock and Key) is still an IOS feature that may encounter in the exam. Besides, it's easy to configure and implement in cases where a quick security fix is required.

In this case, the tricky part is properly figuring the access-list format to be used. The task requires blocking telnet traffic to SW1 ONLY, until a user authenticates. Therefore, we need to create an access-list that denies telnet traffic to ANY of SW1's IP addresses and permits everything else. Upon a user authentication, a **`permit tcp <host> any eq telnet`** entry will be inserted on top of the access-list, temporary allowing access. Finally, the task mentions the five-minute idle timeout, which is configured along with the **`access-enable`** command.

Task 6.2 Verification

To verify the dynamic ACL, first telnet to SW1 without the previous authentication on R2:

```
Rack1R3#telnet 150.1.7.7
Trying 150.1.7.7 ...
% Destination unreachable; gateway or host down
```

Next telnet and login to R2:

```
Rack1R3#telnet 150.1.2.2
Trying 150.1.2.2 ... Open

User Access Verification

Username: TELNET
Password:
[Connection to 150.1.2.2 closed by foreign host]
```

Verify the dynamic ACL on R2:

```
Rack1R2#show ip access-lists
Extended IP access list DYNAMIC
    10 Dynamic PERMIT TELNET permit tcp any any eq telnet
       permit tcp any any eq telnet (4 matches) (time left 275)
    20 deny tcp any host 191.1.27.7 eq telnet
    30 deny tcp any host 191.1.7.7 eq telnet
    40 deny tcp any host 191.1.77.7 eq telnet
    50 deny tcp any host 191.1.177.7 eq telnet
    60 deny tcp any host 150.1.7.7 eq telnet (2 matches)
    70 permit ip any any (184 matches)
```

Finally telnet to SW1:

```
Rack1R3#telnet 150.1.7.7
Trying 150.1.7.7 ... Open


User Access Verification

Password:
Rack1SW1>
```


Task 7.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

System Management > SNMPv2c Server
System Management > SNMPv2c Access-Control

R3:
```
access-list 25 permit 191.1.7.100
access-list 25 permit 191.1.77.100
access-list 50 permit 191.1.7.100
!
snmp-server community CISCORO RO 25
snmp-server community CISCORW RW 50
snmp-server system-shutdown
snmp-server host 191.1.7.100 CISCOTRAP
snmp-server host 191.1.77.100 CISCOTRAP
snmp-server enable traps
```

Task 7.1 Breakdown

Although this section does not explicitly state that SNMP traps need to be enabled, the wording of the task indicated that not only should the community be set to CISCOTRAP but SNMP traps should be enabled. To enable SNMP traps the **snmp-server enable traps** command was configured.

To allow a device to be reloaded via SNMP, the **snmp-server system-shutdown** will need to be configured. Technically, the device will not be shutdown, but will be reloaded. The network management station will also need RW access via SNMP to reload the device. This is why the first network management station was given RW access in this section.

Task 7.1 Verification

Verify that the SNMP traps are configured:

```
Rack1R3#show snmp | begin logging
SNMP logging: enabled
    Logging to 191.1.7.100.162, 0/10, 0 sent, 0 dropped.
    Logging to 191.1.77.100.162, 0/10, 0 sent, 0 dropped.
```

Make sure that system shutdown via SNMP is enabled:

```
Rack1R3#show running-config | include snmp.*shut
snmp-server system-shutdown
```

Verify configured hosts and SNMP access-lists:

```
Rack1R3#show snmp host
Notification host: 191.1.7.100  udp-port: 162   type: trap
user: CISCOTRAP security model: v1

Notification host: 191.1.77.100 udp-port: 162   type: trap
user: CISCOTRAP security model: v1

Rack1R3#show snmp community

Community name: ILMI
Community Index: cisco0
Community SecurityName: ILMI
storage-type: read-only  active

Community name: CISCORO
Community Index: cisco1
Community SecurityName: CISCORO
storage-type: nonvolatile        active access-list: 25

Community name: CISCORW
Community Index: cisco2
Community SecurityName: CISCORW
storage-type: nonvolatile        active access-list: 50

Community name: CISCOTRAP
Community Index: cisco3
Community SecurityName: CISCOTRAP
storage-type: nonvolatile        active
```


Task 7.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

System Management > RMON Alarms

R1:

```
logging 191.1.7.100
!
rmon event 1 log
rmon alarm 1 ifEntry.10.3 60 delta rising-threshold 80000 1 falling-
threshold 40000 1
```

R3:

```
logging 191.1.7.100
!
rmon event 1 log
rmon alarm 1 ifEntry.10.5 60 delta rising-threshold 80000 1 falling-
threshold 40000 1
```


Task 7.2 Breakdown

The key to this section is the reference to the word 'average'. RMON can monitor two values, absolute or delta. The absolute value is the value since the last reload of a device or resetting (if available) of the value's counters. Delta on the other hand is monitoring the rate of change in a value.

Certain values like CPU utilization are normally monitored for the absolute value and not the delta value. It would be more useful to know when the one minute CPU utilization rises above 75% (absolute), than it is when the one minute CPU utilization changes 10% in value (delta). Input or output interface values (i.e. input octets) normally are monitored for their rate of change. This is done by taking the delta value. In this section, a log message will be generated whenever the delta values rise above 80000 or falls below 40000.

🕮 Deep Dive

- In a few words, how would you describe difference between RMON and SNMP?

Task 7.2 Verification

Verify the RMON configuration:

```
Rack1R3#show rmon alarms
Alarm 1 is active, owned by config
 Monitors ifInOctets.5 every 60 second(s)
 Taking delta samples, last value was 0
 Rising threshold is 840000, assigned to event 1
 Falling threshold is 40000, assigned to event 1
 On startup enable rising or falling alarm

Rack1R3#show rmon events
Event 1 is active, owned by config
Description is
Event firing causes log,
last event fired at  0y0w0d,00:00:00,
Current uptime       0y0w0d,07:49:51
```

After configuring, you may see a message on the console at the next polling interval, since your value is below the falling-threshold value.

```
Rack1R3#
%RMON-5-FALLINGTRAP: Falling trap is generated because the value of
ifInOctets.5 has fallen below the falling-threshold value 40000
Rack1R3#
```

In this particular task, we were given the explicit information to use for the interface. If it is necessary to look up the interface, you can use the **show snmp mib ifmib ifindex** command to see the index numbers assigned to your interfaces.

```
Rack1R1#show snmp mib ifmib ifindex
FastEthernet0/0: Ifindex = 1
Loopback0: Ifindex = 6
Null0: Ifindex = 5
Serial0/0: Ifindex = 2
VoIP-Null0: Ifindex = 4
Serial0/1: Ifindex = 3
```


Task 7.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

System Management > CDP

R4:
```
cdp source-interface Loopback0
cdp timer 5
cdp holdtime 15
```

SW2:
```
cdp timer 5
cdp holdtime 15
```


Task 7.3 Breakdown

Cisco Discovery Protocol is a media and protocol independent layer 2 protocol. CDP advertisements include useful information such as device type, device name, and local and remote interface connections. CDP can also be used to transport routing information when used with On Demand Routing (ODR).

CDP is enabled on all Cisco devices by default, and can be globally disabled with the **no cdp run** command, or disabled on a per interface basis with the **no cdp enable** interface level command.

CDP advertisement intervals are controlled by the global configuration commands **cdp timer** and **cdp holdtime**. The **cdp source-interface** command can be used to modify which IP address information is included with CDP advertisements.

🕮 Deep Dive

- What is the standard-based protocol equivalent to CDP in functions?

Task 7.3 Verification

```
Rack1R4#show cdp
Global CDP information:
        Sending CDP packets every 5 seconds
        Sending a holdtime value of 15 seconds
        Sending CDPv2 advertisements is  enabled
        Source interface is Loopback0
```


Task 7.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Traffic Filtering with Extended Access-Lists
IP Services > IOS Small Services & Finger

SW2:

```
service udp-small-servers
!
interface FastEthernet0/18
 ip access-group 100 in
!
access-list 100 deny   udp any any eq discard
access-list 100 deny   udp any any eq 19
access-list 100 permit ip any any
```

Task 7.4 Breakdown

TCP and UDP small servers are simple diagnostic utilities for testing network reachability. These services include echo, chargen, discard, and daytime for TCP, and echo, chargen, and discard for UDP. Typically, these services are disabled in order to avoid various security vulnerabilities that are associated with them. To enable these services, issue the **service tcp-small-servers** or **service udp-small-servers** global configuration commands.

Task 7.4 Verification

Verify that the chargen/discard ports are reachable. Traceroute can be used to generate traffic with a UDP port destination.

```
Rack1R4#traceroute
Protocol [ip]:
Target IP address: 191.1.48.8
Source address:
Numeric display [n]:
Timeout in seconds [3]:
Probe count [3]:
Minimum Time to Live [1]:
Maximum Time to Live [30]:
Port Number [33434]: 9
Loose, Strict, Record, Timestamp, Verbose[none]:
Type escape sequence to abort.
Tracing the route to 191.1.48.8

  1 191.1.48.8 [AS 200] !A  *  0 msec
```

After testing both ports 9 and 19, check the ACL matches:

```
Rack1SW2#show ip access-lists 100
Extended IP access list 100
    10 deny udp any any eq discard (1 match)
    20 deny udp any any eq 19 (1 matches)
    30 permit ip any any (747 matches)
```

Verify that UDP small-services are enabled:

```
Rack1SW2#show running-config | include small
service udp-small-servers
```


Task 8.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC LLQ and Remaining Bandwidth Reservations
QoS > MQC Classification and Marking

R3 and R4:

```
class-map match-all RTP
 match protocol rtp
!
policy-map QOS
 class RTP
  priority percent 25
```

R3:

```
interface Serial1/0
 service-policy output QOS
```

R4:

```
interface Serial0/0/0
 service-policy output QOS
```

Task 8.1 Breakdown

This type of priority queueing is known as Low Latency Queueing. Unlike the legacy priority-list, LLQ can prioritize traffic, while at the same time ensure that other traffic gets serviced. In the legacy priority queue, all packets in the upper queues are serviced before lower queues are checked for packets. This can, and does, result in packets in the lower queues being starved of bandwidth. The LLQ prevents this case by setting a maximum bandwidth threshold for which traffic will be prioritized.

The above MQC configuration dictates that RTP packets will always be dequeued first out the Frame Relay connections of R3 and R4 up to 25% of the bandwidth. When RTP traffic exceeds 25% of the output queue, the excess of 25% does not receive low latency. In the case that there is congestion on the link, traffic in excess of this 25% may be dropped.

Prior to IOS 12.2, the **bandwidth percent** and the **priority percent** commands were *relative* reservations based on what the available bandwidth of the interface. In newer IOS releases, these reservations are *absolute* reservations. The difference between these reservations can be seen as follows.

Caution

The bandwidth value that this percentage reservation is based off of is the configured bandwidth value of the interface. For a practical implementation, the **bandwidth** value of the interface should be modified to reflect the provisioned rate of the layer 2 circuit

The *available bandwidth* of an interface is calculated as:

```
Available_Bandwidth = (Configured_Bandwidth * max-reserved-
bandwidth/100) - (LLQ - RTP - RSVP)
```

Where *Configured_Bandwidth* is the bandwidth value of the interface as specified by the **bandwidth** command, and where **max-reserved-bandwidth** is the configured **max-reserved-bandwidth** of the interface (defaults to 75%). This reservable value is put into place to ensure that necessary network traffic (layer 2 keepalives, layer 3 routing) gets the service that it requires.

To see what the available bandwidth of an interface is, issue the **show queue [interface]** command:


```
Rack1R3#show queue Fa0/0
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: weighted fair
  Output queue: 0/1000/64/0 (size/max total/threshold/drops)
     Conversations  0/1/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 75000 kilobits/sec
```

From the above output, it is evident that this interface is a 100Mbps FastEthernet interface (default configured bandwidth value of 100Mbps). The available bandwidth is 75000Kbps, which is 75% of the default interface bandwidth of 100Mbps. This above router is running 12.2(15)T5, in which a reservation is always *absolute*. The following demonstrates so:

```
ip cef
!
class-map match-all FTP
  match protocol ftp
!
 policy-map QOS
  class FTP
   bandwidth percent 25
!
interface FastEthernet0/0
 service-policy output QOS

Rack1R1#show queue Fa0/0 | in Available
     Available Bandwidth 50000 kilobits/sec
```

Notice from the above output that the available bandwidth value just decreased by 25 Mbps, or 25% of 100Mbps. This is an *absolute* reservation. This has the same effect as if the **bandwidth percent 25** statement actually **said bandwidth 25000**, as seen as follows:

```
policy-map QOS
  class FTP
   bandwidth 25000

Rack1R3#show queue Fa0/0 | include Available
     Available Bandwidth 50000 kilobits/sec
```

Notice the same output. This is still an *absolute* reservation. In **older** IOS releases, percentage reservations were *relative*, as follows:

```
Rack1R3#show queue Fa0/0 | include Available
     Available Bandwidth 75000 kilobits/sec
```

Here, we see the same FastEthernet interface with no prior reservations. As max-reserved-bandwidth is 75 by default there is an available bandwidth of 75Mbps. Now apply the same configuration as before:


```
class-map match-all FTP
  match protocol ftp
!
policy-map QOS
  class FTP
   bandwidth percent 50
!
interface FastEthernet0/0
 service-policy output QOS
!

Rack1R3#show queue Fa0/0 | include Available
     Available Bandwidth 75000 kilobits/sec
```

Although 50% of the bandwidth on this interface is reserved for FTP, it is a relative reservation of what is available. Since the available bandwidth on the interface is 7.5Mbps, FTP is effectively guaranteed a minimum of 37.5Mbps (50% of 75% of 10Mbps). In order to actually reserve 50 Mbps for FTP in this case there are three options.

1. Set 'max-reserved-bandwidth' to 100

```
interface FastEthernet0/0
 max-reserved-bandwidth 100
 service-policy output QOS

Rack1R3#show queue Fa0/0 | include Available
     Available Bandwidth 100000 kilobits/sec
```

Since 100 Mbps is now available on the interface, FTP is guaranteed 50 Mbps (50% of 100 Mbps). This method should be used with caution, as reserving too much of the output queue of an interface can result in delay or loss of necessary layer 2 and layer 3 network control packets.

2. Do an absolute **`bandwidth [kbps]`** reservation

```
class-map match-all FTP
  match protocol ftp
!
policy-map QOS
  class FTP
   bandwidth 50000
!
interface FastEthernet0/0
 service-policy output QOS
```



```
Rack1R3#show queue Fa0/0 | include Available
     Available Bandwidth 25000 kilobits/sec
```

bandwidth [kbps] and **priority [kbps]** are always absolute reservations regardless of the IOS version, and are not based on the available bandwidth of the interface. It is evident that after configuring **bandwidth 50000** under the FTP class, only 25 Mbps is now available on the interface.

3. Change the configured **bandwidth** value on the interface

While not very practical, the bandwidth value on the interface can be adjusted so that the following would be true:

```
Interface_bandwidth = configured_bandwidth * max-reserved-bandwidth/100

Configured_bandwidth = interface_bandwidth * 100/max-reserved-bandwidth

interface FastEthernet0/0
 bandwidth 133334
 service-policy output QOS

Rack1R3#show queue Fa0/0 | include Available
     Available Bandwidth 100000 kilobits/sec
```

While the third option is a roundabout solution, the point of the exercise is to show that the available bandwidth is based on the configured **bandwidth** keyword, and not a function of the physical interface.

🕮 Deep Dive

- What benefits do relative bandwidth reservation settings offer?

Task 8.1 Verification

Verify the policy-map configuration:

```
Rack1R4#show policy-map interface serial 0/0/0

 Serial0/0/0

  Service-policy output: QOS

    queue stats for all priority classes:

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

    Class-map: RTP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol rtp
      Priority: 25% (386 kbps), burst bytes 9650, b/w exceed drops: 0


    Class-map: class-default (match-any)
      25 packets, 1195 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 25/1248
```


Task 8.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > Classification and Marking
QoS > MQC WRED

R3 and R4:

```
class-map match-all NOT_HTTP
 match not protocol http
!
! MQC HQF (CBWFQ replacement) allows for fair-queue in any class
!
policy-map QOS
  class NOT_HTTP
   bandwidth remaining percent 30
   fair-queue
   random-detect
```

Task 8.2 Breakdown

The above exercise is designed to show the usage of the **match not** keyword in the class-map, and to illustrate how random early detection works within the modular quality of service. To configure WRED and or fair-queueing in the MQC, one of two conditions must be met. There must either be a **bandwidth** reservation made within a class, or the default-class must be running weighted fair queuing.

🕮 Deep Dive

- What type of fair queueing is supported in HQF?

Task 8.2 Verification

Check the policy-map applied on R4:

```
R4#show policy-map interface serial 0/0/0
 Serial0/0/0

  Service-policy output: QOS

    queue stats for all priority classes:

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

    Class-map: RTP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol rtp
      Priority: 25% (386 kbps), burst bytes 9650, b/w exceed drops: 0


    Class-map: NOT_HTTP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: not protocol http
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops/flowdrops) 0/0/0/0
      (pkts output/bytes output) 0/0
      bandwidth remaining 30% (347 kbps)
      Fair-queue: per-flow queue limit 16
        Exp-weight-constant: 9 (1/512)
        Mean queue depth: 0 packets
```



```
        class     Transmitted       Random drop      Tail/Flow drop
Minimum Maximum Mark
                  pkts/bytes    pkts/bytes       pkts/bytes    thresh
thresh  prob

        0                 0/0               0/0              0/0
20            40  1/10
        1                 0/0               0/0              0/0
22            40  1/10
        2                 0/0               0/0              0/0
24            40  1/10
        3                 0/0               0/0              0/0
26            40  1/10
        4                 0/0               0/0              0/0
28            40  1/10
        5                 0/0               0/0              0/0
30            40  1/10
        6                 0/0               0/0              0/0
32            40  1/10
        7                 0/0               0/0              0/0
34            40  1/10

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

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

Task 8.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > Generic TCP/UDP Header Compression

R1:
```
interface Serial0/1
 ip tcp header-compression
 ip tcp compression-connections 64
```

R3:
```
interface Serial1/2
 ip tcp header-compression
 ip tcp compression-connections 64
```

Task 8.3 Breakdown

The simplest bandwidth optimization technique available with VoIP is header compression. The same logic applies to interactive TCP sessions, in other words to any flows that have small payload, comparable in size with the header. Such flows effective bandwidth usage is significantly affected by the header sizes.

The command for enabling header comparession are simple. The only thing to remember is that every flow requires two contexts, one in very direction. Therefore, for 32 flows we would need 64 compression contexts.

Task 8.3 Verification

```
Rack1R3#show ip tcp header-compression
TCP/IP header compression statistics:
  Interface Serial1/2 (compression on, VJ)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 64 rx slots, 64 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 64 free
contexts
```


VOL1 Scenarios Reference

Bridging and Switching > 802.1q Trunking
Bridging and Switching > STP Root Bridge Election
Bridging and Switching > 802.1q Tunneling
Security > Port Security
Bridging and Switching > STP Portfast
Bridging and Switching > STP Portfast Default
OSPF > OSPF over Broadcast Media
OSPF > OSPF over Non-Broadcast Media
OSPF > OSPF Not-So-Stubby Areas
OSPF > OSPF Not-So-Stubby Area and Default Routing
OSPF > OSPF Conditional Default Routing
BGP > BGP Filtering with Prefix Lists
BGP > BGP Aggregation
BGP > BGP Dampening
BGP > BGP Dampening with Route Map
IPV6 > Link Local Addressing
IPv6 > Unique Local Addressing
IPv6 > Global Aggregatable Addressing
IPv6 > RIPng
IPv6 > RIPng over NBMA
IPv6 > RIPng Default Routing
IPv6 > EIGRPv6
IPv6 > EIGRPv6 Summarization
MPLS VPN > PE-CE Routing with EIGRP
Security > Traffic Filtering with Extended Access-Lists
MPLS VPN > PE-CE Routing with OSPF
MPLS VPN > OSPF Sham Link
Multicast > Stub Multicast Routing & IGMP Helper
Security > VLAN Filters for Non-IP Traffic
Security > Filtering Packets with Dynamic Access-Lists
System Management > SNMPv2c Server
System Management > SNMPv2c Access-Control
System Management > RMON Alarms
System Management > CDP
Security > Traffic Filtering with Extended Access-Lists
IP Services > IOS Small Services & Finger
QoS > MQC LLQ and Remaining Bandwidth Reservations
QoS > MQC Classification and Marking
QoS > Classification and Marking
QoS > MQC WRED
QoS > Generic TCP/UDP Header Compression

Deep Dive Answers

- Trunking applies additional header to Ethernet frames. Why doesn't this create any MTU problems in the networks?
    - The trunking header is local between two devices, which are both aware of the additional overhead on a particular physical link. Therefore, both can properly process the oversized tagged frames.
- Is it possible to make a non-root bridge to always remain STP forwarding on all ports?
    - Yes, the main idea is to isolate the switch from exiting STP topology. This could be accomplished using BPDU filtering or L2 protocol tunneling. Of course, neither is recommended in real life scenarios.
- How does STP interpret a QinQ tunnel?
    - From STP standpoint, QinQ tunnel looks like a shared segment with multiple attached bridges. There is a caveat in this approach: RSTP assumes that every full-duplex link is a point-to-point connection. If such a link is connected to a QinQ cloud, wrong assumption could be made, as in reality the connection is multipoint. However, in most cases this discrepancy does not have any serious side effects.
- For QinQ packets, do core devices still need to learn the MAC addresses of encapsulated frames?
    - Yes, since the VLAN tags are not used for packet forwarding, rather for selecting the active physical connections. It is possible to change the hardware programming so that VLAN tags are interpreted the same way as Frame-Relay PVCs and used for forwarding decision. This turns VLANs into point-to-point circuits and eliminates the need for MAC address learning. This approach is known as EVCs (Ethernet VCs) and commonly used in Metro Ethernet deployments.
- Do switches still need to learn MAC addresses on secured ports?
    - Yes, unless all MAC addresses are statically programmed on the port. The switch always learns up to the maximum allowed MAC addresses, counting both dynamic and static MACs.
- Is there another way to block BPDUs received on an interface, other than BPDU Filter?
    - Yes, you may use a VLAN filter for instance. Another way would be enabling L2 protocol tunneling, which eliminates local BPDU processing.

- Why is it mandatory to have forwarder address in type-5 LSAs translated from type-7 LSA?
    - NSSA area is allowed to have just one translator, in order to avoid permanent translation loops. This could leave such an area with only a single entry point, which is a suboptimal case. By using the forwarder address, translated LSAs may hint the recipients on selecting the optimum entry point for the area.
- What are the main tools for routing loop prevention in redistribution scenarios?
    - While performing redistribution, your main goal is to ensure that routing information from one routing domain does not re-enter the same domain via another routing domain. This is commonly achieved by tagging the routing information and filtering it on re-entry points. Another way is using administrative distance manipulation to ensure external routing information is always less preferred compared to internal.
- How many type-5 LSAs is required in OSPF to redistribute 10 external prefixes?
    - OSPF creates a separate type-5 LSA for every external prefix redistributed. There is no information packing and type-5 LSA has largest size among the others. This is why you should be careful when injecting large amount of external information into OSPF – it can easily overflow OSPF database.
- Construct a prefix list that rejects all class A subnets with the prefix lengths from /17 to /24
    - Class A subnets have the topmost bit in the left octet set to zero. This could be matched by a base prefix 0.0.0.0/1. The resulting prefix range is defined by the following entry:

```
permit 0.0.0.0/1 ge 17 le 24
```

- Is it safe to perform route redistribution into BGP? Is there a safer way to advertise prefixes?
    - This is not considered a safe practice, unless tight control is enforced by means of route tagging and use of route-maps. Optimally, you advertise routes into BGP by creating local static routes to Null0 and advertising them into BGP using the **network** command.

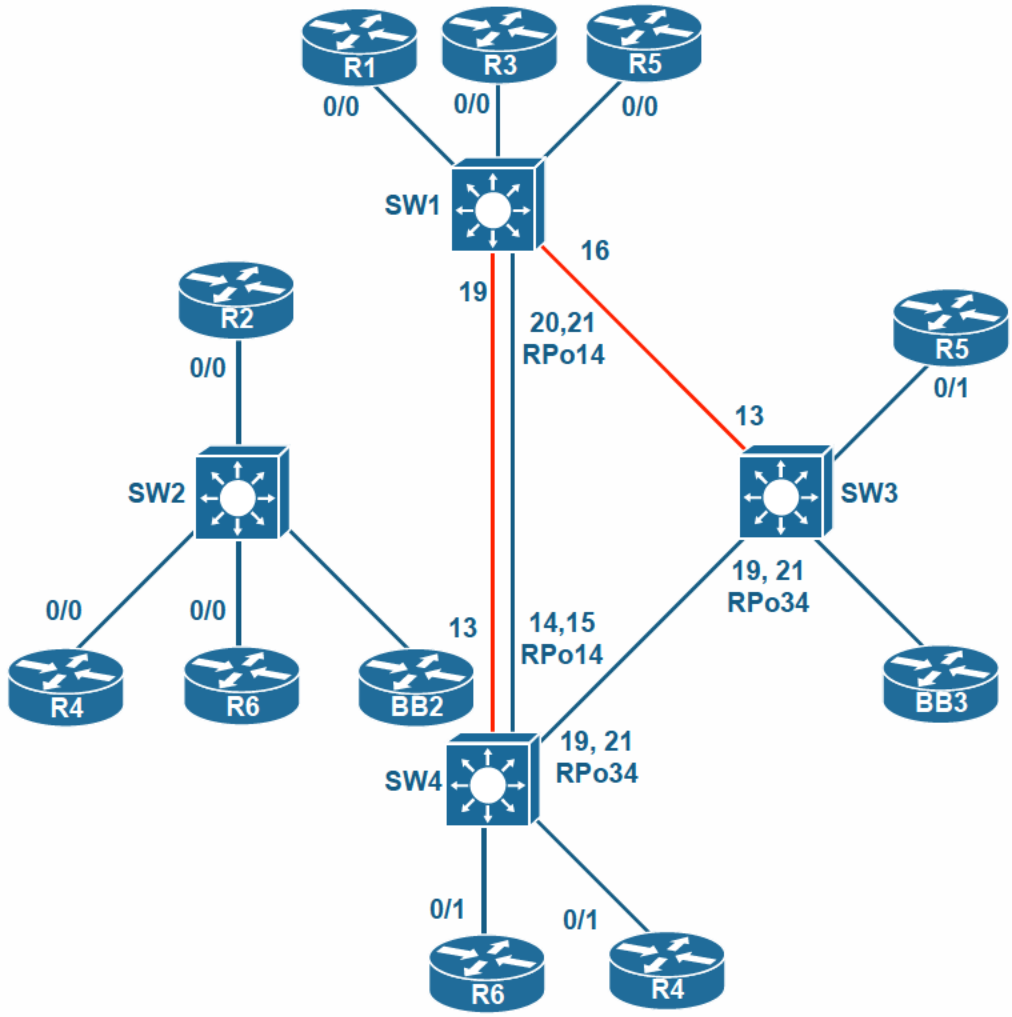
- Why is it important to differentiate dampening options based on prefix types?
    - Typically, aggressive dampening settings are applied to long prefixes, such as /19, /20 or /24, as damping those is less disruptive then damping large aggregates such as /8. The massively aggregated blocks typically receive less, if no dampening at all – the high level of aggregation guarantees more stable prefix behavior.
- Is there a way to overcome the split-horizon problem without disabling split-horizon feature globally?
    - Yes, you may either use unicast updates between the spokes, if those are supported, or use tunnel interfaces between the spokes.
- Will EIGRP detect subnet mask discrepancy based on the HELLO packet exchange?
    - No always. EIGRP only checks that the source IP in received packet belongs to the subnets on the local interface. The remote side may have different network mask and this may cause erratic routing.
- What can you use in place of a OSPF sham-link?
    - You may create a manual tunnel and explicitly assign it to the corresponding area.
- Can you implement Stub Multicast routing in multi-hop fashion?
    - Yes, by simply cascading the routes and enabling PIM passive mode using neighbor filtering. The last router in the stub chain should forward IGMP messages to the first true PIM router in the chain. Though such scheme is possible, it is rarely used.
- What is the main difference between designated querier election in IGMPv1 and IGMPv2?
    - IGMPv1 selects PIM DR as the designated querier, which is by default based on highest priority and IP address. IGMPv2 implements internal election procedure, based on lowest IP address on the segment. This helps reducing load on the multicast routers, by splitting some functionality.

- Can you think of a reason for using DEC STP variant with cisco bridges other than compatibility issues with legacy equipment?
    - You may need another STP flavor when you want to tunnel STP over another STP topology. For example if you want to bridge layer 2 protocols between two VLANs but don't want the overlaid spanning-tree to interact with the "underlying" STP, as this may result in unwanted port blocking. As a matter of fact, this was the reason Cisco came with a special "Inter-VLAN" bridge spanning-tree flavor, which uses different frame format and multicast address.
- In a few words, how would you describe difference between RMON and SNMP?
    - RMON is primarily aimed at collecting statistical information on traffic flows while SNMP is designed to manage and read the devices settings. RMON hasn't become widely adapted, as integrated network analyzers never provided enough information to become a substitution of standalone products.
- What is the standard-based protocol equivalent to CDP in functions?
    - LLDP or Link Layer Discovery Protocol is IEEE standard, vendor neutral protocol to discover devices on the same link and learn their capabilities.
- What benefits do relative bandwidth reservation settings offer?
    - Such configuration easily scales with link changes and could be templatized and applied to multiple links on the network without the need of any changes.
- What type of fair queueing is supported in HQF?
    - HQF, as a replacement of CBWFQ/WFQ treats all traffic flows equally when enabled for fair-queueing. HQF allows fair scheduling within any user-defined class, not just the class-default like it was supported with CBWFQ.

Lab 7 Solutions

Troubleshooting

Use the commands **`show interfaces status`** and **`show etherchannel summary`** to discover the existing connections in L2 topology, prior to starting with L2 technologies section and/or troubleshooting. We are demonstrating the layer 2 diagram AFTER troubleshooting has been completed. Red lines are trunks, other lines represent access or layer 3 routed ports. Notice that some links are bundled into etherchannels already and marked as "RPo*XY*" ports, where R means Routed and P means port-channel.



Issues

R3 – wrong subnet mask on Serial1/2 interface.
R4 – keepalives disabled on Serial0/0/0 interface.
SW4 – Interfaces Fa0/19 and Fa0/21 are not configured for Port-Channel 34

Notice that even after troubleshooting SW2 is isolated from the rest of the topology. This is addresses in the first scenario in Layer 2 Technologies section, where you are required connecting SW2 to SW1.

Task 1.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Bridging and Switching > 802.1q Tunneling
Bridging and Switching > Layer 2 Etherchannel
Bridging and Switching > Storm Control

SW1 and SW2:

```
!
! Configure port-channel in SW1/SW2
! Filter VLANs per the show command output
!
interface Port-channel13
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 1-6,8-4094
 switchport mode trunk
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 1-6,8-4094
 switchport mode trunk
 channel-group 13 mode on
 no shut
!
interface FastEthernet0/14
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 1-6,8-4094
 switchport mode trunk
 channel-group 13 mode on
 no shut
!
interface FastEthernet0/15
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 1-6,8-4094
 switchport mode trunk
 channel-group 13 mode on
 no shut
```


SW1:

```
!
! Enable L2 protocol tunneling on SW1.
! SW1 connects to R1 and R3, different VLANs are used for segregation
!
interface FastEthernet0/1
 switchport access vlan 100
 switchport mode access
 l2protocol-tunnel cdp
 no cdp enable
!
interface FastEthernet0/3
 switchport access vlan 101
 switchport mode access
 l2protocol-tunnel cdp
 no cdp enable
```

SW4:

```
interface FastEthernet0/17
 switchport access vlan 101
 switchport mode access
 l2protocol-tunnel cdp
 no cdp enable
 no shutdown
!
interface FastEthernet0/18
 switchport access vlan 100
 switchport mode access
 l2protocol-tunnel cdp
 no cdp enable
 no shutdown
```

SW1:

```
!
! Enable unicast rate-limiting on SW1's connection to R5
!
interface FastEthernet0/5
 storm-control unicast level 25.00
```


Task 1.1 Breakdown

There is a set of requirements in this scenario. The first one asks for configuring a port-channel interface between SW1 and SW2. The second relates to L2 protocol tunneling between R1/SW2 and R3/SW2. The last requirement is about storm-control configuration, which is a result of rate-limit requirement. The reason for using storm-control is the task requirement to limit the port-rate in percentage of the access speed.

Setting up the port-channel per the task requirement is straight-forward. Make sure you read all the show command output correctly and identify correct settings, such as allowed vlans, trunking encapsulation, native VLAN and so forth.

For the L2 protocol tunneling part you need to decide whether you need QinQ tunneling or simple L2 protocol tunnel with an access VLAN. Since both R1 and R2 ports are not configured for tagged subinterfaces, you may configured the “provider” edge just for access VLANs and tunnel CDP to make the customer devices think they are directly connected.

After comelting this task, you should have all switches connected at layer 2, forming a hub-and-spoke topology from L2 perspective, with SW1 being the hub device. This loop-less topology with no blocked ports.

Task 1.1 Verification

Verify that R1 sees SW2 as a CDP neighor.

```
Rack1R1#show cdp neighbors fa0/0
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater

Device ID      Local Intrfce    Holdtme   Capability  Platform  Port ID
Rack1SW2       Fas 0/0           125      R S I       WS-C3560- Fas 0/21

Rack1R3#show cdp neighbors fa0/0
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater

Device ID      Local Intrfce    Holdtme   Capability  Platform  Port ID
Rack1SW2       Fas 0/0           155      R S I       WS-C3560- Fas 0/20
```

Apply the same vefication for R3/SW2:

```
Rack1SW2#show cdp neighbors fa0/20
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone

Device ID    Local Intrfce    Holdtme  Capability    Platform    Port ID
Rack1R3        Fas 0/20        153          R S I     2611XM     Fas 0/0


Rack1SW2#show cdp neighbors fa0/21
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone

Device ID    Local Intrfce    Holdtme  Capability    Platform    Port ID
Rack1R1        Fas 0/21        129          R S I     2610XM     Fas 0/0
```


Verify that output for port-channels matches the task requirements:

```
Rack1SW1#show etherchannel 13 port-channel
                Port-channels in the group:
                ---------------------------

Port-channel: Po13
------------

Age of the Port-channel   = 00d:00h:26m:52s
Logical slot/port   = 2/13          Number of ports = 3
GC                  = 0x00000000      HotStandBy port = null
Port state          = Port-channel Ag-Inuse
Protocol            =    -

Ports in the Port-channel:
Index   Load   Port     EC state        No of bits
------+------+------+------------------+-----------
  0     00     Fa0/13   On/FEC             0
  0     00     Fa0/14   On/FEC             0
  0     00     Fa0/15   On/FEC             0

Time since last port bundled:    00d:00h:26m:17s    Fa0/15

Rack1SW1#show interfaces po13 trunk

Port        Mode         Encapsulation  Status        Native vlan
Po13        on           802.1q         trunking      1

Port        Vlans allowed on trunk
Po13        1-6,8-4094

Port        Vlans allowed and active in management domain
Po13        1,3-6,42,57,100-101,263

Port        Vlans in spanning tree forwarding state and not pruned
Po13        1,3-6,42,57,100-101,263
```

Verify storm-control configuration:

```
Rack1SW1#show storm-control fastEthernet 0/5 unicast
Interface  Filter State   Upper         Lower         Current
---------  -------------  ------------  ------------  ----------
Fa0/5      Forwarding      25.00%        25.00%         0.00%
```


Task 1.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Bridging and Switching > Voice VLAN

SW2:

```
!
! Enable MLS QoS to trust marking
!
mls qos
!
interface FastEthernet0/7
 switchport voice vlan 4
 switchport access vlan 5
 switchport mode access
 switchport priority extend cos 1
 mls qos trust cos
!
interface FastEthernet0/8
 switchport voice vlan 4
 switchport access vlan 5
 switchport mode access
 switchport priority extend cos 1
 mls qos trust cos
```


Task 1.2 Breakdown

Since ports on the 3560/3550 series switches default to dynamic mode, installing Cisco IP Phones into the network is a very straightforward process. When a phone is connected the switch will automatically form an 802.1q trunk to the phone. Traffic destined for the PC attached to the IP phone will be carried in the access VLAN. Voice traffic destined for the IP phone itself will be carried in the voice VLAN. These VLANs are defined with the **`switchport access vlan`** and **`switchport voice vlan`** command respectively.

For this task since the CallManager server will be located in VLAN 4 the VoIP traffic from the IP phone should also be placed in VLAN 4. Although technically the voice VLAN could be a different VLAN than the CallManager server is located in, the task asked for the minimal configuration to be used for this task. This ruled out other possible configurations.

Pitfall

Unlike a data VLAN a voice VLAN will not automatically be created when it is assigned. Be sure to create the voice VLAN in the VLAN database before assigning it.

Since VoIP traffic requires special treatment throughout the network, a carefully designed end-to-end QoS policy is required in a network utilizing voice over data. In order to facilitate in creating this policy, QoS must extend down to the access layer. Traffic marking at the access layer is supported through layer 2 Class of Service (CoS) values. By default the 3560/3550 does not process CoS values, and rewrites all frames with a CoS value of zero. To enable the processing of CoS, QoS must be enabled globally by issuing the **`mls qos`** global configuration command.

One QoS is enabled you must decide how the switch will process frames that already have a CoS value set. Typically you would want to set the switch to trust the CoS value that is coming from the IP phone. This is accomplished by issuing the **`mls qos trust cos`** interface level command.

In order to prevent the device attached to the phone from getting better service throughout the network and interfering with VoIP traffic, the Cisco IP phone by default will re-tag all frames received from its extension with a CoS value of zero. To tag them with a different value or to leave them untagged, use the interface level command **`switchport priority extend [ trust | cos (value) ]`**. In the above case all traffic received from the PC attached to the IP phone is remarked with a CoS value of one.

Task 1.2 Verification

Verify Voice/Access VLAN settings on the interface

```
Rack1SW2#show interfaces fa0/7 switchport | include Voice
Voice VLAN: 4 (VLAN0004)

Rack1SW2#show interfaces fa0/8 switchport | include Voice
Voice VLAN: 4 (VLAN0004)

Rack1SW2#show interfaces fa0/7 switchport | include Access|Appliance
Access Mode VLAN: 5 (VLAN0005)
Appliance trust: 1

Rack1SW2#show interfaces fa0/8 switchport | include Access|Appliance
Access Mode VLAN: 5 (VLAN0005)
Appliance trust: 1
```

Verify QoS settings

```
Rack1SW2#show mls qos interface fa0/7
FastEthernet0/7
trust state: trust cos
trust mode: trust cos
trust enabled flag: ena
COS override: dis
default COS: 0
DSCP Mutation Map: Default DSCP Mutation Map
Trust device: none
qos mode: port-based

Rack1SW2#show mls qos interface fa0/8
FastEthernet0/8
trust state: trust cos
trust mode: trust cos
trust enabled flag: ena
COS override: dis
default COS: 0
DSCP Mutation Map: Default DSCP Mutation Map
Trust device: none
qos mode: port-based
```


Task 1.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Bridging and Switching > Private VLANs

For additional information on Private VLANs you may considere reading the following INE blog post:

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

SW1 and SW2:

```
vtp mode transparent
!
vlan 500
  private-vlan isolated
!
vlan 42
  private-vlan primary
  private-vlan association 500
!
interface FastEthernet0/9
 switchport private-vlan host-association 42 500
 switchport mode private-vlan host
```

SW2:

```
interface FastEthernet0/4
 no switchport access vlan 42
 switchport private-vlan mapping 42 500
 switchport mode private-vlan promiscuous
!
interface FastEthernet0/24
 no switchport access vlan 42
 switchport private-vlan mapping 42 500
 switchport mode private-vlan promiscuous
```


Task 1.3 Breakdown

The task requires isolating communications between two ports on different switches. This rules out the protected port feature and requires using Private VLANs for traffic separation. Since both new added hosts should be able to communicate to R4 and BB2, which belong to VLAN 42 (an existing VLAN) this VLAN should be made primary and R4/BB2 ports should be configured as promiscuous.

The rest of the configuration consists of creating a single isolated VLAN, number 500 per the task restrictions. This VLAN should be associated with the primary on both switches and designated as isolated. This will prevent the newly added hosts from communicating to each other, once secondary VLAN 500 is mapped to their ports.

🕮 Deep Dive

- Why do you need VTP Transparent Mode with Private VLANs?

Task 1.3 Verification

Ensure R4 can still reach BB2:

```
Rack1R4#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 = 4/4/8 ms

Rack1SW2#show vlan private-vlan

Primary Secondary Type              Ports
------- --------- ----------------- ------------------------------------
42      500       isolated          Fa0/4, Fa0/9, Fa0/24
```

Validate Private VLAN mappings

```
Rack1SW1#show vlan private-vlan

Primary Secondary Type              Ports
------- --------- ----------------- ------------------------------------
42      500       isolated          Fa0/9
```


Task 1.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Frame Relay > Frame Relay End-to-End Keepalives

R4:
```
interface Serial0/0/0
 frame-relay interface-dlci 405
  class DLCI_405
!
map-class frame-relay DLCI_405
 frame-relay end-to-end keepalive mode reply
```

R5:
```
interface Serial0/0/0.54 point-to-point
 frame-relay interface-dlci 504
  class DLCI_504
!
map-class frame-relay DLCI_504
 frame-relay end-to-end keepalive mode request
```

Task 1.4 Breakdown

Since Frame Relay uses *virtual* circuits a failure in the Frame Relay cloud may not be detected by all switches in the transit path. Therefore, it is the end node's (i.e. router's) responsibility to check the availability of the circuit by using Frame Relay end-to-end keepalives (EEK). To enable EEK, use the map-class subcommand **frame-relay end-to-end keepalive mode [*mode*],** where *mode* is request, reply, bidirectional, or passive-reply.

🕮 Deep Dive

- What other Layer 2 down detection mechanism are available with Frame-Relay?

Task 1.4 Verification

Check the Frame-Relay EEK status on any of the endpoints

```
Rack1R4#show frame-relay end-to-end keepalive

End-to-end Keepalive Statistics for Interface Serial0/0/0 (Frame Relay
DTE)

DLCI = 405, DLCI USAGE = LOCAL, VC STATUS = ACTIVE (EEK UP) ← status

RECEIVE SIDE STATISTICS

Send Sequence Number: 254,      Receive Sequence Number: 254
Configured Event Window: 3,     Configured Error Threshold: 2
Total Observed Events: 6,       Total Observed Errors: 3
Monitored Events: 1,            Monitored Errors: 0
Successive Successes: 1,        End-to-end VC Status: UP ← status
```

Further Reading

Frame-Relay End-to-End Keepalives at Cisco.com

http://www.cisco.com/en/US/docs/ios/12_0t/12_0t5/feature/guide/FRKeep.html

Task 2.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

RIPv2 > RIPv2 Filtering with Prefix-List
RIPv2 > RIPv2 Manual Summarization
RIPv2 > RIPv2 Filtering with Standard Access-List

R2:

```
!
! Accept RIP updates only from R6
!
router rip
 distribute-list gateway R6 in
!
ip prefix-list R6 seq 5 permit 204.12.1.6/32
```

```
R4:
!
! Authenticate RIP packets with BB2
!
key chain RIP
 key 1
  key-string CISCO
!
! Advertise summaries only to BB2
!
interface FastEthernet0/0
 ip rip authentication mode md5
 ip rip authentication key-chain RIP
 ip summary-address rip 163.1.0.0 255.255.192.0
 ip summary-address rip 150.1.0.0 255.255.240.0
!
router rip
 version 2
 network 192.10.1.0
 no passive-interface FastEthernet0/0
!
! Advertise summaries only to BB2
! Do not accept any RIP routes from BB2
!
 distribute-list prefix RIP_SUMMARY out FastEthernet0/0
 distribute-list 1 in FastEthernet0/0
 no auto-summary
!
ip prefix-list RIP_SUMMARY seq 5 permit 163.1.0.0/18
ip prefix-list RIP_SUMMARY seq 10 permit 150.1.0.0/20
!
access-list 1 deny any

R6:
router rip
!
! Only accept RIP routes from R2
!
 distribute-list gateway R2 in
!
! Do not advertise the 54.X.0.0 prefix to R2
!
 distribute-list prefix RIP out FastEthernet0/1
!
ip prefix-list R2 seq 5 permit 204.12.1.2/32
!
ip prefix-list RIP seq 5 permit 163.1.6.0/24
ip prefix-list RIP seq 10 permit 150.1.6.0/24
```


Task 2.1 Breakdown

There is a number of requirements in this task. Some requirements deals with prefix filtering while other deal with summarization.

1. R2 and R6 should only accept RIP prefixes from each other, and not from BB3. Normally route filtering is applied to the updates received, not the updates source. Alternatively, an IP prefix-list could be used in the **`distribute-list gateway`** statement t filter on route source. This allows prefixes to be filtered as they are received based on the source of the update. In the above task this syntax is used on both R2 and R6 to only accept RIP updates from each other. This allows updates learned from both BB1 and BB3 to be denied, but still allows updates to be received from R2 and R6 respectively.
2. R4 should advertise only summary prefix information to BB2 and does no accept any RIP prefixes from BB2. This is accomplished using the summarization procedure described below and an inbound distribute list blocking all RIP updates.

The following summarization procedure is based on binary manipulation. There is an alternative approach that does not utilize binary breakdowns. This approach is described in the following publication:

http://blog.ine.com/2010/03/17/a-simple-ipv4-prefix-summarization-procedure/

The first step in properly summarizing the internal address space is to list all known addresses sequentially. The addresses used in this network are as follows:

```
163.1.4.0/24
163.1.5.0/24
163.1.6.0/24
163.1.7.0/24
163.1.12.0/24
163.1.13.0/24
163.1.18.0/24
163.1.35.0/24
163.1.38.0/24
163.1.45.0/24
163.1.54.0/24
163.1.57.0/24
```

From this list it is evident that the first two octets are the same. Therefore the minimum summary that will encompass all of this address space is 163.1.0.0/16. To determine what the maximum summarization is that will encompass all of the above address space, next write out all addresses in the third octet in binary:

	128	64	32	16	8	4	2	1
4	0	0	0	0	0	1	0	0
5	0	0	0	0	0	1	0	1
6	0	0	0	0	0	1	1	0
7	0	0	0	0	0	1	1	1
12	0	0	0	0	1	1	0	0
13	0	0	0	0	1	1	0	1
18	0	0	0	1	0	0	1	0
35	0	0	1	0	0	0	1	1
38	0	0	1	0	0	1	1	0
45	0	0	1	0	1	1	0	1
54	0	0	1	1	0	1	1	0
57	0	0	1	1	0	1	0	1

Next, count how many bit positions are consistent. From the above output it is evident that two places, the 128 and 64 bits, are consistent. Add these two bits onto the previous summarization of /16, and are resulting summary is /18. Therefore the final summary for this task is 163.1.0.0/16

🕮 Deep Dive

- What other way you can use in RIP to filter based on route source other than gateway filtering?

Task 2.1 Verification

Verify that R2 receives prefixes for R6 Loopback0 and Fast0/0 interfaces:

```
Rack1R2#show ip route rip
     163.1.0.0/24 is subnetted, 2 subnets
R       163.1.6.0 [120/1] via 204.12.1.6, 00:00:01, FastEthernet0/0
     150.1.0.0/24 is subnetted, 2 subnets
R       150.1.6.0 [120/1] via 204.12.1.6, 00:00:01, FastEthernet0/0
```

Verify that R6 does not receive any prefix from the backbone routers:

```
Rack1R6#show ip route rip

Rack1R6#
```

Verify that R4 only sends the summary prefixes to BB2:

```
Rack1R4#debug ip rip

RIP: sending v2 update to 224.0.0.9 via FastEthernet0/0 (192.10.1.4)
   RIP: build update entries
   150.1.0.0/20 via 0.0.0.0, metric 3, tag 0
   163.1.0.0/18 via 0.0.0.0, metric 1, tag 0
```

Verify that we do not receive any routing information from BB2 via RIP:

```
Rack1R4#show ip route rip | include via 192.10.2.254
Rack1R4#
```


Task 2.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

OSPF > Repairing Discontiguous OSPF Areas with Virtual Link
OSPF > OSPF Network Type Non-Broadcast
OSPF > OSPF Filtering with Distribute Lists
OSPF > OSPF Internal Summarization

R1:

```
!
! Configure a logical/tunnel connection between R1 and R5 and run OSPF
! area 0 on it. This is a replacement for virtual link.
!
interface Tunnel0
 ip address 163.1.15.1 255.255.255.0
 tunnel source Loopback0
 tunnel destination 163.1.35.5
!
! Configure the OSPF network type as non-broadcast so that packets are
! unicasted and there is DR/BDR election
!
interface Serial0/1
 ip ospf network non-broadcast
!
! Summarize R4 and R5 Loopbacks
!
router ospf 1
 router-id 150.1.1.1
 area 0 range 150.1.4.0 255.255.254.0
 network 163.1.12.1 0.0.0.0 area 2
 network 163.1.13.1 0.0.0.0 area 1
 network 163.1.15.1 0.0.0.0 area 0
 neighbor 163.1.13.3
 neighbor 163.1.12.2
```


R2:

```
!
! Configure R2 so that id does not participate in DR/BDR election
! process on Serial0/0 circuit
!
interface Serial0/0
 ip ospf priority 0
!
! Configure OSPF on Serial0/0 circuit
!
router ospf 1
 router-id 150.1.2.2
 network 163.1.12.2 0.0.0.0 area 2
```

R3:

```
!
! Configure R3 so that id does not participate in DR/BDR election
! process on Serial1/2 circuit
!
interface Serial1/2
 ip ospf network non-broadcast
 ip ospf priority 0
!
! Configure OSPF on Serial1/2 circuit
!
router ospf 1
 router-id 150.1.3.3
 network 163.1.13.3 0.0.0.0 area 1
```


R4:

```
!
! Advertise Loopback0 and VLAN 4 into OSPF
!
router ospf 1
 network 150.1.4.4 0.0.0.0 area 0
 network 163.1.4.4 0.0.0.0 area 0
```

R5:

```
!
! Advertise Loopback0 and VLAN5 into OSPF
!
router ospf 1
 area 0 range 150.1.4.0 255.255.254.0
 network 150.1.5.5 0.0.0.0 area 0
 network 163.1.5.5 0.0.0.0 area 0

!
! Configure a logical/tunnel connection between R1 and R5 and run OSPF
! area 0 on it
!
interface Tunnel0
 ip address 163.1.15.5 255.255.255.0
 tunnel source Serial0/0/0.35
 tunnel destination 150.1.1.1
!
! The tunnel belongs to Area 0
!
router ospf 1
 network 163.1.15.5 0.0.0.0 area 0
```

SW1:

```
!
! Filter out IA prefix 10.3.3.3/32, because SW4 cannot summarize it
!
router ospf 1
 network 163.1.0.1 0.0.0.0 area 0
 distribute-list 1 in
!
access-list 1 deny   10.3.3.3
access-list 1 permit any
```



```
SW3:
!
! Advertise SW3 Loopback into NEW area 34 (area is arbitraryr)
!
router ospf 1
 network 10.3.3.3 0.0.0.0 area 34

SW4:
!
! Summarize SW4's Loopback to send to SW1.
!
router ospf 1
 area 34 range 10.0.0.0 255.0.0.0
 network 10.4.4.4 0.0.0.0 area 34
```

Task 2.2 Breakdown

This is a complicated scenario with multiple requirements. We are going to deal with them sequentially, starting with the most challenging one.

- The first requirement is making R1 the DR on both connections to R3 and R2. We achieve this by making both links NBMA segments from OSPF standpoint, since DR election is required and multicasting is disallowed. By tuning OSPF priorities on R3 and R2 we ensure R1 is always the DR on both NBMA links.
- The second requirement is advertising R4 and R5 interfaces into area 0. This does not create any problems. However, there is a requirement for R3 to see R4's and R5's Loopback prefixes as a single summary. We could do this by configuring summarization at R5. However, there is another requirement in the task – connecting R1 and R2 using Areas 1 and 2. This configuration will disconnect Area 2 from the backbone area, unless we configure a virtual link from R1 to R5.

  At this point we have two implicit requirements: area 1 should be transit and carry a virtual link and area 0 prefixes should be summarized in the transit area. Per OSPF functionality, it is impossible to summarize prefixes entering a transit area, as this may result in routing loops. Therefore, summarization requirement and virtual link do not work together well.

  This solution to this problem is using a tunnel in area 0 as opposed to a virtual-link. The tunnel will encapsulate transit packets and hide them from R3, therefore preventing any possible forwarding loops due to summarization. At the same time, it is possible now to summarize R4's and R5's Loopback0 prefixes at R1 and R5 to fulfill the requirement for R3 to see only the summary prefix.

  As for tunnel configuration, you need to be careful as well. Tunnel source and destination should belong to Area 1, otherwise a recursive routing may occur over the tunnel. This is why we source the tunnel off the Frame-Relay interface on R5.

- The third requirement is making SW3 and SW4 summarize their Loopback0 interfaces advertised to SW1. However, all switches share the same area 0. Therefore, summarization is not possible unless the Loopback0 interfaces are in different areas or advertised via redistribution. However, the latter is prohibited by the task requirements.

  Based on this, we create a new area on SW3/SW4 and advertise SW3's and SW4's Loopbacks into this area. You may use the same area number or different areas, it does not really matter. How would you make SW1 seeing only the summary prefix for those two subnets?

  As you first option, you may summarize routes form the new area on both SW3 and SW4. However, this will make SW3's prefix unreachable, as SW4 will lose detailed information about SW3's specific network. Therefore, configure summarization on SW4 only. This leaves a problem – SW1 still sees the inter-area route for SW3's Loopback0 prefix as SW4 obviously does not summarize the inter-area routes in OSPF Area 0. To resolve this last issue, configure RIB filter in SW1's OSPF process, blocking the prefix 10.X.3.3/32 from entering the routing table on SW1.

🕮 Deep Dive

- How can routing loops form in OSPF transit area if you allow summarizing into it?
- Why OSPF does not summarize inter-area routes and only applies summarization to intra-area prefixes?

Task 2.2 Verification

Verify that VLAN4, VLAN5 and Loopback subnets are advertised into area 0:

```
Rack1R4#show ip route 150.1.5.5
Routing entry for 150.1.5.5/32
  Known via "ospf 1", distance 110, metric 65, type intra area
  Last update from 163.1.54.5 on Serial0/0/0, 00:28:35 ago
  Routing Descriptor Blocks:
  * 163.1.54.5, from 150.1.5.5, 00:28:35 ago, via Serial0/0/0
      Route metric is 65, traffic share count is 1

Rack1R4#show ip route 163.1.5.0
Routing entry for 163.1.5.0/24
  Known via "ospf 1", distance 110, metric 65, type intra area
  Last update from 163.1.54.5 on Serial0/0/0, 00:29:37 ago
  Routing Descriptor Blocks:
  * 163.1.54.5, from 150.1.5.5, 00:29:37 ago, via Serial0/0/0
      Route metric is 65, traffic share count is 1

Rack1R5#show ip route 150.1.4.4
Routing entry for 150.1.4.4/32
  Known via "ospf 1", distance 110, metric 65, type intra area
  Last update from 163.1.54.4 on Serial0/0/0.54, 00:38:24 ago
  Routing Descriptor Blocks:
  * 163.1.54.4, from 150.1.4.4, 00:38:24 ago, via Serial0/0/0.54
      Route metric is 65, traffic share count is 1

Rack1R5#show ip route 163.1.4.0
Routing entry for 163.1.4.0/24
  Known via "ospf 1", distance 110, metric 65, type intra area
  Last update from 163.1.54.4 on Serial0/0/0.54, 00:29:15 ago
  Routing Descriptor Blocks:
  * 163.1.54.4, from 150.1.4.4, 00:29:15 ago, via Serial0/0/0.54
      Route metric is 65, traffic share count is 1
```


Verify that the summary LSA for R4/R5's Loopback0 prefixes is generated:

```
Rack1R5#show ip ospf database summary 150.1.4.0

            OSPF Router with ID (150.1.5.5) (Process ID 1)

                Summary Net Link States (Area 1)

  LS age: 50
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.4.0 (summary Network Number)
  Advertising Router: 150.1.5.5
  LS Seq Number: 80000002
  Checksum: 0x1AE4
  Length: 28
  Network Mask: /23
        TOS: 0  Metric: 1

  LS age: 1714
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 150.1.4.0 (summary Network Number)
  Advertising Router: 150.1.1.1
  LS Seq Number: 80000001
  Checksum: 0xDE96
  Length: 28
  Network Mask: /23
        TOS: 0  Metric: 11112
```

Verify the summary route on R3:

```
Rack1R3#show ip route 150.1.4.0
Routing entry for 150.1.4.0/23
  Known via "ospf 1", distance 110, metric 782, type inter area
  Last update from 163.1.35.5 on Serial1/0, 00:01:48 ago
  Routing Descriptor Blocks:
  * 163.1.35.5, from 150.1.5.5, 00:01:48 ago, via Serial1/0
      Route metric is 782, traffic share count is 1
```


Check that tunnel is working:

```
Rack1R1#ping 163.1.15.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 163.1.15.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5),round-trip min/avg/max=104/106/108 ms
```

Verify the OSPF neighbors. Verify the neighbors' state to be sure that R1 is the DR.

```
Rack1R1#show ip ospf neighbor

Neighbor ID     Pri State           Dead Time   Address         Interface
150.1.5.5         0 FULL/  -        00:00:34    163.1.15.5      Tunnel0
150.1.3.3         0 FULL/DROTHER    00:01:32    163.1.13.3      Serial0/1
150.1.2.2         0 FULL/DROTHER    00:01:39    163.1.12.2      Serial0/0
```

Verify the network types on R1 interfaces:

```
Rack1R1#show ip ospf interface s0/0
Serial0/0 is up, line protocol is up
  Internet Address 163.1.12.1/24, Area 2
  Process ID 1, Router ID 150.1.1.1,Network Type NON_BROADCAST,Cost: 64
<output omitted>

Rack1R1#show ip ospf interface s0/1
Serial0/1 is up, line protocol is up
  Internet Address 163.1.13.1/24, Area 1
  Process ID 1, Router ID 150.1.1.1,Network Type NON_BROADCAST,Cost: 64
```

Verify OSPF routes on R1 to ensure we receive prefixes from R4 and R5:

```
Rack1R1#show ip route ospf
     163.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
O       163.1.35.0/24 [110/845] via 163.1.13.3, 00:05:11, Serial0/1
O       163.1.54.0/24 [110/11175] via 163.1.15.5, 00:03:31, Tunnel0
     150.1.0.0/16 is variably subnetted, 5 subnets, 3 masks
O       150.1.4.0/23 is a summary, 00:03:31, Null0
O       150.1.5.5/32 [110/11112] via 163.1.15.5, 00:03:31, Tunnel0
O       150.1.4.4/32 [110/11176] via 163.1.15.5, 00:03:31, Tunnel0
```


Verify that R3 still only has the summary prefix 150.1.4.0/23:

```
Rack1R3#show ip route 150.1.4.0
Routing entry for 150.1.4.0/23
  Known via "ospf 1", distance 110, metric 782, type inter area
  Last update from 163.1.35.5 on Serial1/0, 00:04:22 ago
  Routing Descriptor Blocks:
  * 163.1.35.5, from 150.1.5.5, 00:04:22 ago, via Serial1/0
      Route metric is 782, traffic share count is 1
```

Verify that SW3 and SW4 Loopbacks appear as asked in SW1 routing table:

```
Rack1SW1#show ip route ospf | include _10
O IA 10.0.0.0/8 [110/2] via 163.1.0.4, 00:10:30, Port-channel14
```

Task 2.3 Solution

🕮 Before You Begin

Redistribution topology has multiple points of mutual redistribution. This is a complex scenario, which requires good understanding of redistribution process. The breakdown for this scenario provides enough information on the subject, but we recommend reading the following post on the INE blog to get better understanding of redistribution: “Understanding Redistribution Part I, II and III”:

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

http://blog.ine.com/2008/02/19/understanding-redistribution-part-ii/

http://blog.ine.com/2008/03/17/understanding-redistribution-part-iii/

R1:

```
router ospf 1
 redistribute rip subnets
!
router rip
 redistribute connected route-map CONNECTED->RIP
 redistribute ospf 1 metric 1
!
! Loopback0 is not advertised into RIP in the initial configs
!
route-map CONNECTED->RIP permit 10
 match interface Loopback0
!
! The prefixes R1 needs to see via RIP
!
ip access-list standard RIP_PREFIXES
 permit 163.1.38.0
 permit 150.1.8.0
!
! Adjust the distance for selected prefixes
!
router rip
 distance 109 0.0.0.0 255.255.255.255 RIP_PREFIXES
!
interface FastEthernet 0/0
 ip summary-address rip 150.1.4.0 255.255.254.0
!
! Make network /24 for routing consistency
!
interface Loopback0
 ip ospf network point-to-point
```


R2:

```
router ospf 1
 redistribute rip subnets
!
router rip
 redistribute connected route-map CONNECTED->RIP
 redistribute ospf 1 metric 1
!
! This route map is already present in the configuration
! We need to advertise VLAN 263 to get BGP next-hop reachability
!
route-map CONNECTED_TO_OSPF permit 10
 match interface Loopback0 FastEthernet0/0
!
route-map CONNECTED->RIP permit 10
 match interface Loopback0

router rip
 redistribute connected route-map CONNECTED_TO_RIP
!
!
! Make network /24 for routing consistency
!
interface Loopback0
 ip ospf network point-to-point
```

R3:

```
router ospf 1
 redistribute rip subnets
!
router rip
 redistribute connected route-map CONNECTED->RIP
 redistribute ospf 1 metric 1
!
route-map CONNECTED->RIP permit 10
 match interface Loopback0
!
! The below networks should be reachable via RIP
!
ip access-list standard RIP_PREFIXES
 permit 163.1.28.0
 permit 150.1.8.0
!
router rip
 distance 109 0.0.0.0 255.255.255.255 RIP_PREFIXES
!
interface FastEthernet 0/0
 ip summary-address rip 150.1.4.0 255.255.254.0
!
! Make network /24 for routing consistency
!
interface Loopback0
 ip ospf network point-to-point
```


R4:

```
router ospf 1
 redistribute rip subnets
!
router rip
 redistribute ospf 1 metric 1
 distance 109 163.1.45.5 0.0.0.0 RIP_PREFIXES
!
! The below prefixes should be reachable via RIP
!
ip access-list standard RIP_PREFIXES
 permit 150.1.7.0
 permit 163.1.57.0
 permit 163.1.7.0
 permit 163.1.0.0
 permit 163.1.0.128
 permit 10.0.0.0
 permit 163.1.3.0
!
! Make network /24 for routing consistency
!
interface Loopback0
 ip ospf network point-to-point
```

R5:

```
router rip
 distance 109 0.0.0.0 255.255.255.255 RIP_PREFIXES
!
ip access-list standard RIP_PREFIXES
 permit 150.1.7.0
 permit 163.1.7.0
 permit 192.10.1.0
 permit 163.1.0.0
 permit 163.1.0.128
 permit 10.0.0.0
 permit 163.1.3.0
!
! Make network /24 for routing consistency
!
interface Loopback0
 ip ospf network point-to-point
```


SW1:

```
router ospf 1
 default-information originate
!
router rip
 redistribute ospf 1 metric 1
```

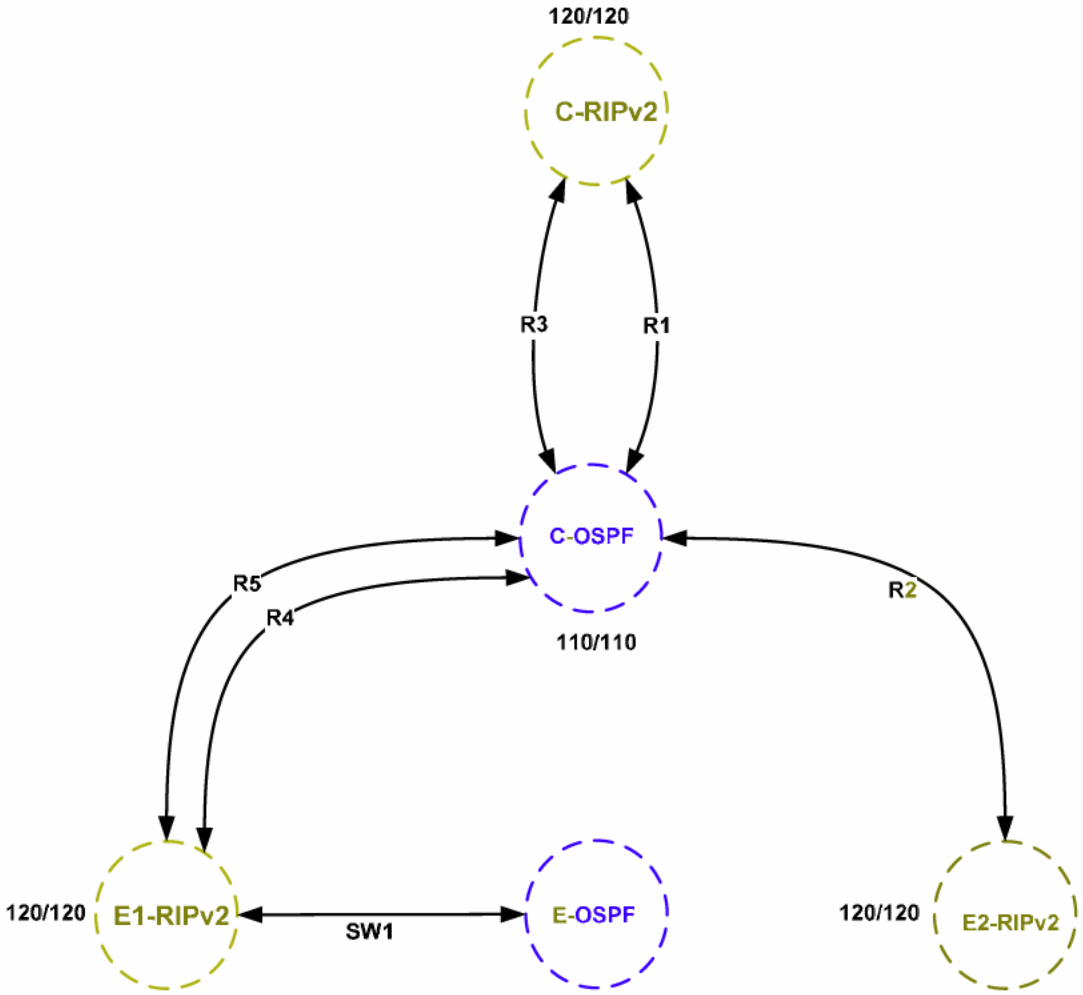
SW2:

```
router rip
 offset-list 1 in 8 FastEthernet0/20
 offset-list 0 in 4 FastEthernet0/21
!
! The below networks are to be preferred via R1
!
access-list 1 permit 150.1.6.0
access-list 1 permit 150.1.2.0
access-list 1 permit 150.1.1.0
access-list 1 permit 163.1.6.0
access-list 1 permit 163.1.12.0
access-list 1 permit 204.12.1.0
```


Task 2.3 Breakdown

The redistribution requirements are complicated with multiple redistribution points, so we need to draw a redistribution diagram, like we did in Lab 2.

3) Outline all routing domains used in the scenarios along with their names. In our case there are four domains: C-OSPF (Core OSPF, spanning R1, R2, R3, R4 and R5), C-RIPv2 (Core RIPv2, spans R1, R2 and SW2), E1-RIPv2 (Edge1 RIPv2, spans R4 and R5) and finally E2-RIPv2 (spans R2 and R6).
4) Connect the routing domains using lines that represent routers performing redistribution. In our case, we have the following routers performing mutual redistribution: R1, R2, R4, R5 and SW1.



The arrow endpoints represent the redistribution direction. The diagram demonstrates that OSPF is the central transit routing protocol, usaged to connect the edge RIPv2 and OSPF domains. The situation becomes complicated because there is C-RIPv2 domain exchanging prefixes with C-OSPF.

Let's look at this diagram and see if we can come up with a quick solution. Start with the default AD values displayed on the diagram. With such configuration, routes leaked from C-OSPF into C-RIPv2 will remain in the RIP domain and never preempt any of the original OSPF prefixes on R1 or R3. Thus, if we keep the AD values for C-OSPF and C-RIPv2 per the diagram, there is no routing feedback loop ever formed. However, we need to tune RIP processes on R1 and R3 to make sure they prefer reaching SW2's prefixes via RIP.

Now for the second dual-attached domain: E1-RIPv2. This one does not contain many routes per se, but has some routes injected from E-OSPF as we'll see later. When those are leaked into C-OSPF they will preempt RIPv2 native prefixes on R4 and R5 and result in routing oscillations. To prevent this, we need to tune RIPv2 processes on R4 and R5 to prefer the selected "native" RIPv2 prefixes via RIP, by lowering their AD. At the same time, C-OSPF routes leaked into E1-RIPv2 will never leak back into C-OSPF by the virtue of AD blocking.

Finally, E-OSPF has only a single point of attachment via SW1, and therefore creates no additional problems with redistribution. However, keep in mind that E-OSPF routes are redistributes into E1-RIPv2 and therefore R4 and R5 see them as "native" RIPv2 routes. This makes us adjust prefix AD tuning on R4 and R5 and make these edge routers prefer prefixes coming originally from E-OSPF via RIP. As for routes sent to SW3 and SW4, we know that R5 is sending a default route to SW1, as configured in the previous section. Thus, we may simply re-advertise this route down to OSPF domain.

Lastly, we configure all Loopback0 interfaces in the OSPF domain as OSPF P2P networks, to make them advertised as /24 by OSPF. The reason being, is that the same Loopbacks may leak into another routing protocol (e.g. RIP) as native /24 while OSPF would treat them as /32 prefixes. This may lead to routing confusion and improper path selection. It is always a good idea to ensure consistent view of the given subnet in all routing domain, e.g. summarize prefix to all domains if it is summarized in one of them. For this reason, we summarize R4 and R5 Loopback0 interfaces when redistributing them into RIP on R1 and R3, to prevent R1 and R3 from selecting SW2 as the next-hop for R4 and R5's Loopback subnets.

SW2 has two connections to the rest of the routing domain, one through R1 and the other through R3. Since there are many more routes learned from behind R3 than from behind R1, the easiest way to achieve optimal routing is to match the routes learned from behind R1 and prefer these through R1. This is accomplished by creating a standard access-list that matches these prefixes. Next, the metric of these routes is incremented by as they are learned from R3. All other routes learned from R1 have their metric indiscriminately bumped up too. Based on this, SW2 will prefer the routes per the task requirements.

🕮 Deep Dive

- How is control-plane routing loop different from data-plane loop?

Task 2.3 Verification

Verify connectivity for internal network with following TCL script *(run it on routers R1 through R6):*

```
foreach i {
150.1.1.1
163.1.15.1
163.1.13.1
163.1.12.1
163.1.18.1
150.1.2.2
163.1.12.2
204.12.1.2
163.1.35.3
163.1.38.3
150.1.3.3
163.1.13.3
163.1.45.4
163.1.54.4
150.1.4.4
163.1.4.4
192.10.1.4
163.1.35.5
163.1.45.5
163.1.54.5
150.1.5.5
163.1.57.5
163.1.5.5
163.1.15.5
150.1.6.6
163.1.6.6
204.12.1.6
150.1.7.7
163.1.57.7
163.1.7.7
163.1.0.1
163.1.38.8
150.1.8.8
163.1.18.8
163.1.0.133
10.3.3.3
163.1.0.4
163.1.0.134
10.4.4.4
} {puts [exec "ping $i"]}
```


Next verify best path selection on SW2.

```
Rack24SW2#show ip route | include FastEthernet0/21
C       163.1.18.0/24 is directly connected, FastEthernet0/21
R       163.1.12.0/24 [120/5] via 163.1.18.1, 00:00:12,
FastEthernet0/21
R       163.1.6.0/24 [120/5] via 163.1.18.1, 00:00:12, FastEthernet0/21
R    204.12.1.0/24 [120/5] via 163.1.18.1, 00:00:12, FastEthernet0/21
R       150.1.6.0/24 [120/5] via 163.1.18.1, 00:00:12, FastEthernet0/21
R       150.1.2.0/24 [120/5] via 163.1.18.1, 00:00:12, FastEthernet0/21
R       150.1.1.0/24 [120/5] via 163.1.18.1, 00:00:12, FastEthernet0/21

Rack24SW2#show ip route | include FastEthernet0/20
R    192.10.1.0/24 [120/1] via 163.1.38.3, 00:00:10, FastEthernet0/20
R       163.1.0.128/25 [120/1] via 163.1.38.3, 00:00:10,
FastEthernet0/20
R       163.1.57.0/24 [120/1] via 163.1.38.3, 00:00:10,
FastEthernet0/20
R       163.1.54.0/24 [120/1] via 163.1.38.3, 00:00:10,
FastEthernet0/20
R       163.1.45.5/32 [120/1] via 163.1.38.3, 00:00:10,
FastEthernet0/20
R       163.1.45.0/24 [120/1] via 163.1.38.3, 00:00:10,
FastEthernet0/20
R       163.1.35.0/24 [120/1] via 163.1.38.3, 00:00:10,
FastEthernet0/20
C       163.1.38.0/24 is directly connected, FastEthernet0/20
R       163.1.15.0/24 [120/1] via 163.1.38.3, 00:00:10,
FastEthernet0/20
R       163.1.13.0/24 [120/1] via 163.1.38.3, 00:00:10,
FastEthernet0/20
R       163.1.0.0/25 [120/1] via 163.1.38.3, 00:00:10, FastEthernet0/20
R       163.1.7.0/24 [120/1] via 163.1.38.3, 00:00:10, FastEthernet0/20
R       163.1.5.0/24 [120/1] via 163.1.38.3, 00:00:10, FastEthernet0/20
R       163.1.4.0/24 [120/1] via 163.1.38.3, 00:00:10, FastEthernet0/20
R       150.1.7.0/24 [120/1] via 163.1.38.3, 00:00:11, FastEthernet0/20
R       150.1.4.0/23 [120/1] via 163.1.38.3, 00:00:11, FastEthernet0/20
R       150.1.3.0/24 [120/1] via 163.1.38.3, 00:00:11, FastEthernet0/20
```

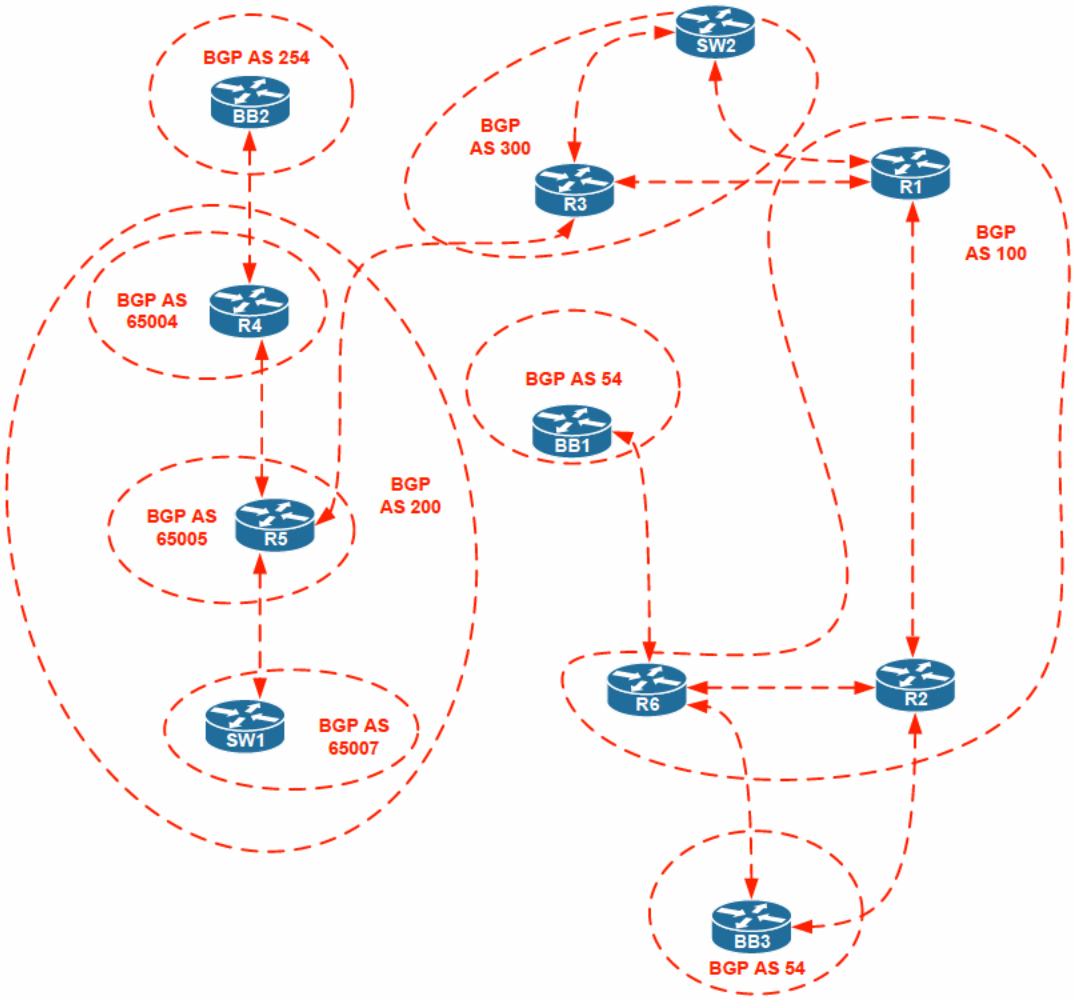

Task 2.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

```
BGP > Establishing iBGP Peering
BGP > Establishing eBGP Peering
BGP > Multihop eBGP Peering
BGP > iBGP Confederations
BGP > BGP Bestpath Selection – Local Preference
```

The below is BGP peering diagram built based on the existing BGP confederation and the information presented in this scenario.



R4:

```
router bgp 65004
 no synchronization
 bgp router-id 150.1.4.4
 bgp confederation identifier 200
 bgp confederation peers 65005
 neighbor 150.1.5.5 remote-as 65005
 neighbor 150.1.5.5 ebgp-multihop
 neighbor 150.1.5.5 update-source Loopback0
 neighbor 192.10.1.254 remote-as 254
 neighbor 192.10.1.254 password CISCO
 no auto-summary
```

R5:

```
router bgp 65005
 no synchronization
 bgp router-id 150.1.5.5
 bgp confederation identifier 200
 bgp confederation peers 65004 65007
 neighbor 150.1.4.4 remote-as 65004
 neighbor 150.1.4.4 ebgp-multihop
 neighbor 150.1.4.4 update-source Loopback0
 neighbor 163.1.35.3 remote-as 300
 neighbor 163.1.57.7 remote-as 65007
 no auto-summary
```

R6:

```
router bgp 100
 neighbor 54.1.7.254 route-map BB1 in
 neighbor 204.12.1.254 route-map BB3 in
!
ip as-path access-list 1 permit _54$
!
route-map BB1 permit 10
 match as-path 1
 set local-preference 200
!
route-map BB1 permit 20
!
route-map BB3 permit 10
 match as-path 1
!
route-map BB3 permit 20
 set local-preference 200
```

SW1:

```
router bgp 65007
 no synchronization
 bgp router-id 150.1.7.7
 bgp confederation identifier 200
 bgp confederation peers 65005
 neighbor 163.1.57.5 remote-as 65005
```


Task 2.4 Breakdown

BGP confederation is used in large scale BGP implementations in order to reduce the amount of iBGP peering sessions required to transport Network Layer Reachability Information (NLRI) throughout the AS. By subdividing the autonomous system into smaller sub-autonomous systems, the amount of iBGP peering sessions can be dramatically reduced.

The first step in defining a confederation is to enable the BGP process by issuing the **router bgp [*sub-ASN*]** command. Note that the AS number specified when the process is initiated is the sub-AS number.

Next, the **bgp confederation identifier [*public-AS*]** command is issued in order to define the public autonomous system number. All routers within the confederation must specify this option. Next, for routers that are on the edge of the sub-autonomous system and are peering with other sub-autonomous systems, the **bgp confederation peers [*peer sub-ASs*]** command is used. Only the sub-ASs that are directly peered with need be listed in this statement. Lastly, BGP neighbors are established as usual. The sub-AS number should be referenced for any peers that are within the confederation.

Inter sub-AS peerings exhibit both EBGP and iBGP characteristics. The most notable difference between a normal iBGP peering relationship and an inter sub-AS peering is that iBGP mesh need not be maintained between sub-ASs. However, each sub-AS exhibits normal iBGP behavior within the sub-AS. This means that within each sub-AS there must either be full iBGP mesh or route-reflection. The majority of other attributes between inter sub-AS peers behave as normal iBGP peers. For example, next-hop, local-preference, and MED maintain their values as they are passed between sub-ASs. However, AS-Path information does not.

Confederation sub-AS path information is denoted within parentheses in the AS-path list. However when it comes to best path selection, all sub-ASs within the parentheses count as a single AS. This is required to avoid any internal sub-AS structure from affecting path selection for global internet routes.

The second part of the task deal with BGP traffic engineering. AS 100 is required to prefer routing to AS 54 via BB1 and using BB3 as a backup egress point. Recall the four common attributes used to affect the BGP best path selection, and how they are applied:

Attribute	Direction Applied	Traffic Flow Affected
Weight	Inbound	Outbound
Local-Preference	Inbound	Outbound
AS-Path	Outbound	Inbound
MED	Outbound	Inbound

As a general rule, weight and local-preference are used to affect how traffic leaves the autonomous system, while AS-Path and MED are used to affect how traffic enters the AS. The above task requires that all traffic leaving AS 100 going to prefixes originated in AS 54 exits to BB1, while traffic going to AS 54's customers exits to BB3. Therefore as prefixes are learned from AS 54, either the weight or local-preference attribute should be modified to obtain the desired effect. However, since the above task states that the configuration should be done only on R6, weight cannot be used. Weight is only local to the router, and is not passed on with iBGP updates.

🕮 Deep Dive

- Why changing local preference to affect outbound paths is better than locally manipulating AS_PATHs?

Task 2.4 Verification

Verify BGP peering:

```
Rack1R5#show ip bgp summary  | begin Neighbor
Neighbor        V AS   MsgRcvd MsgSent TblVer InQ OutQ Up/Down  State/PfxRcd
150.1.4.4       4 65004    8      10       14    0    0 00:04:07        3
163.1.35.3      4   300   11       9       14    0    0 00:04:25       10
163.1.57.7      4 65007    7      10       14    0    0 00:03:52        0
```

Verify BGP paths inside the confederation; for instance verify the paths that contain two confederation sub-ASs on SW1:

```
Rack1SW1#show ip bgp regexp \([0-9]+_[0-9]+\)
BGP table version is 14, local router ID is 150.1.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      192.10.1.254             0    100      0 (65005 65004) 254 ?
*> 220.20.3.0       192.10.1.254             0    100      0 (65005 65004) 254 ?
*> 222.22.2.0       192.10.1.254             0    100      0 (65005 65004) 254 ?
```

Lastly, verify that confederation announces prefixes, learned from AS254, to AS 300:

```
Rack1R3#show ip bgp regexp ^200
BGP table version is 14, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      163.1.35.5                             0 200 254 ?
*> 220.20.3.0       163.1.35.5                             0 200 254 ?
*> 222.22.2.0       163.1.35.5                             0 200 254 ?
```


Verify that R6 has selected the best paths for AS54 originated prefixes via BB1:

```
Rack1R6#show ip bgp regexp _54$
BGP table version is 24, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  28.119.16.0/24   204.12.1.254             0             0 54 i
* i                 204.12.1.254             0    100      0 54 i
*>                  54.1.7.254                    200      0 54 i
*  28.119.17.0/24   204.12.1.254             0             0 54 i
* i                 204.12.1.254             0    100      0 54 i
*>                  54.1.7.254                    200      0 54 i
*  114.0.0.0        204.12.1.254                           0 54 i
<output omitted>
```

Now verify R2's BGP table:

```
Rack1R2#show ip bgp regexp _54$
BGP table version is 16, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i28.119.16.0/24   54.1.7.254               0    200      0 54 i
*>                  204.12.1.254             0             0 54 i
* i28.119.17.0/24   54.1.7.254               0    200      0 54 i
*>                  204.12.1.254             0             0 54 i
<output omitted>
```

Note that R2 does not prefer the path via BB1 due to the fact that the next-hop of 54.1.7.254 is unreachable. Lastly verify that the best paths to AS54 customers are through BB3:

```
Rack1R2#show ip bgp regexp 54(_[0-9]+)+$
BGP table version is 16, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 112.0.0.0        204.12.1.254                           0 54 50 60 i
* i                 54.1.7.254               0    100      0 54 50 60 i
*> 113.0.0.0        204.12.1.254                           0 54 50 60 i
* i                 54.1.7.254               0    100      0 54 50 60 i
```

Task 2.5 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Next-Hop Processing – Next-Hop-Self
BGP > BGP Fast Fallover

R6:
```
router bgp 100
 neighbor 204.12.1.2 next-hop-self
 no bgp fast-external-fallover
 neighbor 54.1.7.254 timers 10 30
```

Task 2.5 Breakdown

Two conditions must be met before a route can be considered for BGP best-path selection. The synchronization rule must be met, and there must be a route to the next-hop IP address of the prefix in question.

When a BGP update is passed on to an EBGP neighbor, the next-hop value of the prefix is updated to the local address used to peer with that neighbor. However, when an update is passed on to an iBGP neighbor, the next-hop value is not updated. In certain cases this may cause a problem.

In the above scenario R6 is learning BGP prefixes from BB1. As this is an EBGP peering relationship, the next-hop IP address that R6 sees for these prefixes is the address it uses to peer with BB1, 54.1.7.254. When these routes are propagated on to R2, the next hop value is not updated, as R2 is an iBGP neighbor of R6. However, since R2 does not have an IGP route to the network 54.1.7.0/24, it cannot consider any prefixes learned from BB1 passed on via R6 for the BGP best path selection. In such a case there are two options.

The first option is to inject the transit network into the IGP domain of the BGP network. This will allow all iBGP speaking neighbors to do an IGP lookup on the next-hop values of any prefixes learned from the external system. This method may be inefficient for networks with an exorbitant amount of EBGP neighbors, as these transit networks take up unnecessary space in the IGP table. This is due to the fact that traffic is never sent to a transit network, only past it.

The second option is to modify the next-hop value as the prefix is advertised into the iBGP domain. This is accomplished by issuing the **neighbor [*address*] next-hop-self** BGP process subcommand. This command overrides the default next-hop processing behavior, and modifies the next-hop value to the local address when an EBGP learned prefix is advertised to an iBGP neighbor. This way all internal BGP speaking routers only need to maintain IGP reachability information about the internal network.

Specifically in this scenario's case, when R6 uses the **neighbor 204.12.1.2 next-hop-self** command, updates learned from BB1 have the next-hop value adjusted to 204.12.1.6 when advertised on to R2, as this is the address that R2 is peering with. Since this network is advertised into the IGP (RIP) domain, R1 can do an IGP lookup on all BGP routes learned from AS 54.
By default, when a directly connected interface goes down, any EBGP peers which are directly connected on that interface have their BGP session reset. This in turn will cause the router to withdrawn any BGP prefixes learned from that neighbor from all other BGP peers. This behavior may be undesired in certain cases, as in the case that this task describes. To disable this behavior and wait for the dead time to expire before declaring a directly connected EBGP down, use the **no bgp fast-external-fallover** bgp subcommand.

Additionally, the BGP timers have been adjusted for this neighbor by using the **neighbor [address] timers [hello] [dead]** command.

🕮 Deep Dive

- Is it safe to change a next-hop on iBGP route reflectors?
- Why isn't fast fallover enabled for iBGP sessions?

Task 2.5 Verification

Check to see that next-hop problem at R6 has been resolved:

```
Rack1R2#sh ip bgp regexp _54$
BGP table version is 26, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   204.12.1.6               0    200      0 54 i
*                   204.12.1.254             0             0 54 i
*>i28.119.17.0/24   204.12.1.6               0    200      0 54 i
*                   204.12.1.254             0             0 54 i
*>i114.0.0.0        204.12.1.6               0    200      0 54 i
*                   204.12.1.254                           0 54 i
```

Verify the BGP timers configuration:

```
Rack1R6#show ip bgp neighbors 54.1.7.254
BGP neighbor is 54.1.7.254,  remote AS 54, external link
  BGP version 4, remote router ID 212.18.3.1
  BGP state = Established, up for 00:02:13
  Last read 00:00:03, last write 00:00:03, hold time is 30, keepalive
interval is 10 seconds
  Configured hold time is 30,keepalive interval is 10 seconds  Minimum
holdtime from neighbor is 0 seconds
<output omitted>
```


Just to be 100% sure we'll do some debugging (note the timestamps):

```
Rack1R6#debug ip bgp keepalives

09:34:38.610: BGP: 204.12.1.254 sending KEEPALIVE (io)
09:34:38.614: BGP: 204.12.1.254 received KEEPALIVE, length (excl.
header) 0
09:34:39.610: BGP: 54.1.7.254 sending KEEPALIVE (io)
09:34:39.654: BGP: 54.1.7.254 received KEEPALIVE, length (excl. header)
0
09:34:40.610: BGP: 204.12.1.2 sending KEEPALIVE (io)
09:34:40.610: BGP: 204.12.1.2 received KEEPALIVE, length (excl. header)
0
09:34:49.610: BGP: 54.1.7.254 sending KEEPALIVE (io)
09:34:49.654: BGP: 54.1.7.254 received KEEPALIVE, length (excl. header)
0
```

To verify that fast fallover is disabled try shutting down Serial 0/0/0 interface on R6:

```
Rack1R6(config)#int Serial 0/0/0
Rack1R6(config-if)#shutdown
08:32:29.676: %LINK-5-CHANGED: Interface Serial0/0/0, changed state to
administratively down
08:32:29.680: %LINK-3-UPDOWN: Interface Virtual-Access1, changed state
to down
08:32:30.676: %LINEPROTO-5-UPDOWN: Line protocol on Interface
Serial0/0/0, changed state to down
08:32:30.680: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-
Access1, changed state to down
….
08:32:49.156: %BGP-5-ADJCHANGE: neighbor 54.1.7.254 Down BGP
Notification sent
Apr 23 08:32:49.156: %BGP-3-NOTIFICATION: sent to neighbor 54.1.7.254
4/0 (hold time expired) 0 bytes
```

Note that the peering session was not shut down immediately as with fast fallover enabled.

Task 2.6 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Aggregation
BGP > BGP Aggregation – Summary-Only
BGP > BGP Aggregation – Unsuppress Map
BGP > BGP Communities – No-Export

R1:

```
router bgp 100
 aggregate-address 28.119.16.0 255.255.254.0 summary-only
 aggregate-address 112.0.0.0 248.0.0.0 summary-only
 neighbor 163.1.13.3 send-community
 neighbor 163.1.13.3 unsuppress-map UNSUPPRESS->R3
 neighbor 163.1.18.8 send-community
 neighbor 163.1.18.8 unsuppress-map UNSUPPRESS->SW2
!
ip prefix-list UNSUPPRESS->R3 seq 5 permit 28.119.17.0/24
ip prefix-list UNSUPPRESS->R3 seq 10 permit 116.0.0.0/8
ip prefix-list UNSUPPRESS->R3 seq 15 permit 117.0.0.0/8
ip prefix-list UNSUPPRESS->R3 seq 20 permit 118.0.0.0/8
ip prefix-list UNSUPPRESS->R3 seq 25 permit 119.0.0.0/8
!
ip prefix-list UNSUPPRESS->SW2 seq 5 permit 28.119.16.0/24
ip prefix-list UNSUPPRESS->SW2 seq 10 permit 112.0.0.0/8
ip prefix-list UNSUPPRESS->SW2 seq 15 permit 113.0.0.0/8
ip prefix-list UNSUPPRESS->SW2 seq 20 permit 114.0.0.0/8
ip prefix-list UNSUPPRESS->SW2 seq 25 permit 115.0.0.0/8
!
route-map UNSUPPRESS->SW2 permit 10
 match ip address prefix-list UNSUPPRESS->SW2
 set community no-export
!
route-map UNSUPPRESS->R3 permit 10
 match ip address prefix-list UNSUPPRESS->R3
 set community no-export
```


Task 2.6 Breakdown

There are two sections in this task. The first one requires R1 to optimally aggregate all BGP prefixes learned from AS 54. By default when configuring aggregation, the aggregate plus all subnets of the aggregate are advertised. By adding the summary-only keyword, only the aggregate is advertised. There are two major subnets learned from AS 54 and we aggregate them in optimal fashion.

The solution then uses a combination of selective prefix unsuppressing and community manipulation in order to accomplish complex traffic engineering. When a router does a lookup in the IP routing table, it always chooses the route with the longest match to the destination. For example, suppose a packet is received with a destination IP address of 1.2.3.4, and the following routes are in the IP routing table:

```
0.0.0.0/0
1.0.0.0/8
1.2.0.0/16
1.2.3.0/24
```

Regardless of metric, administrative distance, or protocol, packets destined for 1.2.3.4 will always use the route 1.2.3.0/24. This is due to the fact that this is the longest match for the prefix in the routing table. By leaking specific longer matches to AS 300, AS 100 is able to affect the return traffic flow. This is due to the fact that routers in AS 300 have to choice but to use the longest match in the IP routing table.

As aggregation with the **summary-only** keyword has already been configured on R1, AS 300 only has the summary route regarding the prefixes learned from AS 54. Therefore, in order to force AS 300 to make routing decisions based on longer matches in the routing table, the prefixes mentioned in the task must be selectively unsuppressed on a per neighbor basis.

Next, the task states that devices beyond AS should not have this specific reachability information. This has been accomplished by setting the community value to no-export as the prefixes are unsuppressed. In this manner AS 100 is able to force AS 300 to route a specific way, while at the same time ensuring upstream neighbors have the minimum amount of reachability information necessary.

🕮 Deep Dive

- What other solutions could AS 300 and AS 100 use to minimize routing table sizes in AS 100 and keep AS 300 fully informed of the prefixes behind AS 54?

Pitfall

The above task can lead to problems with the order of operations of the BGP engine. For example, if the community no-export was set using a separate route-map than the unsuppress-map, it would not have been applied. This is due to the fact that the route-map is processed *before* the unsuppress-map. In that case, the route-map would be setting the community on prefixes that do not exist in the output engine. The subnets do not exist in the output engine until they are unsuppressed in the unsuppress-map. For this reason it is best always to consolidate all attribute setting in a single route-map. Do not mix and match the application of route-maps, unsuppress-maps, attribute-maps, distribute-lists, prefix-lists, or filter-lists to the same neighbor in the same direction.

Task 2.6 Verification

Verify that only the summary prefixes are sent to AS300 from R1:

```
Rack1R1#show ip bgp neighbors 163.1.18.8 advertised-routes  | begin Net
   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/23   0.0.0.0                              32768 i
*> 112.0.0.0/5      0.0.0.0                              32768 i
```

Verify that the specific prefixes are unsuppressed when sending updates to the respective neighbors:

Prefixes sent to SW2:

```
Rack1R1#show ip bgp neighbors 163.1.18.8 advertised-routes  | begin Net
   Network          Next Hop            Metric LocPrf Weight Path
s>i28.119.16.0/24   204.12.1.6               0    200      0 54 i
*> 28.119.16.0/23   0.0.0.0                             32768 i
s>i112.0.0.0        204.12.1.6               0    200      0 54 50 60 i
*> 112.0.0.0/5      0.0.0.0                             32768 i
s>i113.0.0.0        204.12.1.6               0    200      0 54 50 60 i
s>i114.0.0.0        204.12.1.6               0    200      0 54 i
s>i115.0.0.0        204.12.1.6               0    200      0 54 i
```


Next, prefixes sent to R3:

```
Rack1R1#show ip bgp neighbors 163.1.13.3 advertised-routes | begin Net
   Network          Next Hop        Metric LocPrf Weight Path
*> 28.119.16.0/23   0.0.0.0                        32768 i
s>i28.119.17.0/24   204.12.1.6           0    200      0 54 i
*> 112.0.0.0/5      0.0.0.0                        32768 i
s>i116.0.0.0        204.12.1.6           0    200      0 54 i
s>i117.0.0.0        204.12.1.6           0    200      0 54 i
s>i118.0.0.0        204.12.1.6           0    200      0 54 i
s>i119.0.0.0        204.12.1.6           0    200      0 54 i
*> 205.90.31.0      163.1.18.8                       0 300 200 254 ?
*> 220.20.3.0       163.1.18.8                       0 300 200 254 ?
*> 222.22.2.0       163.1.18.8                       0 300 200 254 ?
```

Finally verify that R3 advertises only the summary prefixes to AS200:

```
Rack1R3#show ip bgp neighbors 163.1.35.5 advertised-routes | begin Net
   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/23   163.1.13.1               0             0 100 i
*> 112.0.0.0/5      163.1.13.1               0             0 100 i
*> 205.90.31.0      163.1.35.5                             0 200 254 ?
*> 220.20.3.0       163.1.35.5                             0 200 254 ?
*> 222.22.2.0       163.1.35.5                             0 200 254 ?

Rack1R3#show ip bgp 28.119.17.0
BGP routing table entry for 28.119.17.0/24, version 31
Paths: (1 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
  Advertised to update-groups:
     1
  100 54
    163.1.13.1 from 163.1.13.1 (150.1.1.1)
      Origin IGP, localpref 100, valid, external, best
      Community: no-export
```


Lastly, verify full BGP connectivity. Run the following TCL script on every BGP enabled router:

```
foreach i {
205.90.31.1
220.20.3.1
222.22.2.1
28.119.16.1
28.119.16.1
28.119.17.1
112.0.0.1
113.0.0.1
114.0.0.1
115.0.0.1
116.0.0.1
117.0.0.1
118.0.0.1
119.0.0.1
} {puts [exec "ping $i"] }
```

Note that pings to AS254 networks from R6 using the source IP address of interface Fa0/1 were unsuccessful. This is due to the fact that this IP subnet is not announced to AS254.

Task 3.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IPV6 > Link Local Addressing
IPv6 > Unique Local Addressing
IPv6 > OSPFv3

If you read the scenarios briefly, you will notice that only three devices are involved in IPv6 configuration, R3, R4 and R5 and there is only one routing protocol, OSPFv3. Therefore, making a separate IPv6 diagram is most likely unnecessary.

R3:

```
ipv6 unicast-routing
!
interface FastEthernet0/0
 ipv6 address FEC0:CC1E:1:38::/64 eui-64
!
interface Serial1/0
 ipv6 address FEC0:CC1E:1:35::3/64
 ipv6 address fe80::3 link-local
 frame-relay map ipv6 FEC0:CC1E:1:35::5 305 broadcast
!
ipv6 router ospf 1
!
interface FastEthernet0/0
 ipv6 ospf 1 area 100
!
! The default OSPFv3 network type for FR is non-broadcast
!
interface Serial1/0
 ipv6 ospf 1 area 100
 frame-relay map ipv6 FE80::5 305 broadcast
 ipv6 ospf network point-to-point
```


R4:

```
ipv6 unicast-routing
!
interface FastEthernet0/1
 ipv6 address FEC0:CC1E:1:4::/64 eui-64
!
interface Serial0/0/0
 ipv6 address FEC0:CC1E:1:54::4/64
 ipv6 address fe80::4 link-local
 frame-relay map ipv6 FEC0:CC1E:1:54::5 405 broadcast
!
interface Serial0/1/0
 ipv6 address FEC0:CC1E:1:45::4/64
!
! Create a set of IPv6 Prefixes
!
interface Loopback100
 ipv6 address 2001:220:20:3::1/64
 ipv6 ospf 1 area 100

!
interface Loopback101
 ipv6 address 2001:222:22:2::1/64
 ipv6 ospf 1 area 100
!
interface Loopback103
 ipv6 address 2001:205:90:31::1/64
 ipv6 ospf 1 area 100
!
! Enable OSPFv3 between R4 and R5
!
interface Serial0/0/0
 ipv6 ospf 1 area 100
 ipv6 ospf network point-to-point
 frame-relay map ipv6 FE80::5 405
!
interface FastEthernet0/1
 ipv6 ospf 1 area 100
!
interface Serial0/1/0
 ipv6 ospf 1 area 100
!
ipv6 router ospf 1
!
! Traffic Filtering using IPv6 Access-Lists
!
interface Serial0/0/0
 ipv6 traffic-filter DENY_FROM_VLAN38 in
!
interface Serial0/1/0
 ipv6 traffic-filter DENY_FROM_VLAN38 in
!
ipv6 access-list DENY_FROM_VLAN38
 deny ipv6 FEC0:CC1E:1:38::/64 FEC0:CC1E:1:4::/64
 permit ipv6 any any
```


R5:

```
ipv6 unicast-routing
!
interface Serial0/1/0
 ipv6 address FEC0:CC1E:1:45::5/64
!
interface Serial0/0/0.54 point-to-point
 ipv6 address FEC0:CC1E:1:54::5/64
 ipv6 address fe80::5 link-local
!
interface Serial0/0/0.35 point-to-point
 ipv6 address FEC0:CC1E:1:35::5/64
 ipv6 address fe80::5 link-local
!
! Change cost to affect traffic routing preference
!
interface Serial0/0/0.35 point-to-point
 ipv6 ospf 1 area 100
!
interface Serial0/0/0.54 point-to-point
 ipv6 ospf 1 area 100
 ipv6 ospf cost 1000
!
interface Serial0/1/0
 ipv6 ospf 1 area 100

ipv6 router ospf 1
```


Task 3.1 Breakdown

The scenario requires basic IPv6 addressing configuration in addition to setting up a single-area OSPFv3. Simple traffic engineering is implemented using OSPFv3 metric manipulation.

Additionally, data-plane traffic filtering needs to be configured by means of IPv6 traffic filters (access-lists). Access-list filtering in IPv6 is configured with the **`ipv6 access-list [name]`** command, where *name* is the access-list identifier. Note that there is no separation between standard access-lists and extended access-lists in IPv6. To apply the access-list to the interface the more intuitive **`ipv6 traffic-filter`** command is used. Keep in mind that just like IPv4 access-lists, IPv6 access-lists end in implicit deny.

🕮 Deep Dive

- Is it possible to implement OSPFv3 traffic engineering in a single area on per-prefix basis?
- What is the main architectural difference between OSPFv2 and OSPFv3?

Task 3.1 Verification

Verify layer 3 to layer 2 mapping on the Frame-Relay interfaces:

```
Rack1R4#show frame-relay map
Serial0/0/0 (up): ipv6 FEC0:CC1E:1:54::5 dlci 405(0x195,0x6450),
static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ipv6 FE80::5 dlci 405(0x195,0x6450), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0 (up): ip 163.1.54.5 dlci 405(0x195,0x6450), static,
              broadcast,
              CISCO, status defined, active
```

Verify connectivity:

```
Rack1R4#ping FEC0:CC1E:1:54::5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FEC0:CC1E:1:54::5, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/59/60 ms
```

Verify the OSPFv3 configuration:

```
Rack1R5#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
150.1.4.4         1   FULL/  -        00:00:39    6               Serial0/1/0
150.1.4.4         1   FULL/  -        00:00:37    5               Serial0/0/0.54
150.1.3.3         1   FULL/  -        00:00:38    6               Serial0/0/0.35
```


Verify the IPv6 OSPFv3 routes:

```
Rack1R5#show ipv6 route ospf
IPv6 Routing Table - Default - 12 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:205:90:31::1/128 [110/64]
     via FE80::215:62FF:FE41:FD9C, Serial0/1/0
O   2001:220:20:3::1/128 [110/64]
     via FE80::215:62FF:FE41:FD9C, Serial0/1/0
O   2001:222:22:2::1/128 [110/64]
     via FE80::215:62FF:FE41:FD9C, Serial0/1/0
O   FEC0:CC1E:1:4::/64 [110/65]
     via FE80::215:62FF:FE41:FD9C, Serial0/1/0
O   FEC0:CC1E:1:38::/64 [110/65]
     via FE80::3, Serial0/0/0.35
```

Verify that packets to the selected IPv6 prefixes flow via the R4-R5 serial link:

```
Rack1R3#traceroute 2001:222:22:2::1

Type escape sequence to abort.
Tracing the route to 2001:222:22:2::1

  1 FEC0:CC1E:1:35::5 44 msec 56 msec 48 msec
  2 FEC0:CC1E:1:45::4 68 msec 80 msec 68 msec
```

To verify the IPv6 traffic filter ping VLAN4 using the source address of VLAN38. First determine VLAN4 interface IPv6 address:

```
Rack1R4#show ipv6 interface brief fastEthernet 0/1
FastEthernet0/1            [up/up]
    FE80::215:62FF:FE41:FD9D
    FEC0:CC1E:1:4:215:62FF:FE41:FD9D
```


Now ping:

```
Rack1R3#ping FEC0:CC1E:1:4:215:62FF:FE41:FD9D source fastEthernet 0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FEC0:CC1E:1:4:215:62FF:FE41:FD9D,
timeout is 2 seconds:
Packet sent with a source address of FEC0:CC1E:1:38:211:20FF:FE93:E560
AAAAA
Success rate is 0 percent (0/5
```

Ping using another source IPv6 address:

```
Rack1R3#ping FEC0:CC1E:1:4:215:62FF:FE41:FD9D

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FEC0:CC1E:1:4:215:62FF:FE41:FD9D,
timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 88/93/100
ms
```

Verify the access-list hits on R4:

```
Rack1R4#show ipv6 access-list
IPv6 access list DENY_FROM_VLAN38
    deny ipv6 FEC0:CC1E:1:38::/64 any (5 matches) sequence 10
    permit ipv6 any any (38 matches) sequence 20
```

Verify that packets sourced from VLAN 38 but destined to other subnets than VLAN 4 are allowed:

```
Rack1R4#show ipv6 interface brief serial 0/0/0
Serial0/0/0                [up/up]
    FE80::4
    FEC0:CC1E:1:54::4

Rack1R3#ping ipv6 FEC0:CC1E:1:54::4 source fastEthernet 0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FEC0:CC1E:1:54::4, timeout is 2
seconds:
Packet sent with a source address of FEC0:CC1E:1:38:211:20FF:FE93:E560
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 100/109/117
ms
```


Task 5.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Multicast > PIM Dense Mode
Multicast > Multicast RPF Failure

The task is simple enough to require a multicast topology diagram. We only have two routers involved in multicast routing, using PIM Dense mode.

R4:

```
ip multicast-routing
!
interface FastEthernet0/1
 ip pim dense-mode
!
interface Serial0/1/0
 ip pim dense-mode
!
ip mroute 0.0.0.0 0.0.0.0 Serial0/1/0
```

R5:

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


Task 5.1 Breakdown

Another option to using a static mroute would be to manipulate the IGP routing protocols so that R4 prefers to route across the Serial over the Frame Relay to reach R5. This will ensure that the RPF check is successful when a static mroute is not used. Notice that a real lab exam may prohibit the use of static multicast routes and you may have to resort to IGP metric manipulation or using M-BGP.

Task 5.1 Verification

To verify the multicast configuration join VLAN4 interface to group 239.4.4.4. Next, ping 239.4.4.4 from R5 using VLAN5 as the source:

```
Rack1R4(config)#interface fastEthernet 0/1
Rack1R4(config-if)#ip igmp join-group 239.4.4.4

Rack1R5#ping
Protocol [ip]:
Target IP address: 239.4.4.4
Repeat count [1]: 10
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Interface [All]: Serial0/1/0
Time to live [255]:
Source address: 163.1.5.5
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 10, 100-byte ICMP Echos to 239.4.4.4, timeout is 2 seconds:
Packet sent with a source address of 163.1.5.5

Reply to request 0 from 163.1.45.4, 28 ms
Reply to request 1 from 163.1.45.4, 28 ms
Reply to request 2 from 163.1.45.4, 28 ms
Reply to request 3 from 163.1.45.4, 28 ms
Reply to request 4 from 163.1.45.4, 28 ms
```


Verify multicast routing table on R4:

```
Rack1R4#show ip mroute
IP Multicast Routing Table
<snip>

(*, 239.4.4.4), 00:01:30/stopped, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/1/0, Forward/Dense, 00:01:30/00:00:00
    FastEthernet0/1, Forward/Dense, 00:01:30/00:00:00
(163.1.5.5, 239.4.4.4), 00:00:29/00:02:55, flags: LT
  Incoming interface: Serial0/1/0, RPF nbr 163.1.45.5, Mroute
  Outgoing interface list:
    FastEthernet0/1, Forward/Dense, 00:00:29/00:00:00


Rack1R4(config)#interface fastEthernet 0/1
Rack1R4(config-if)#no ip igmp join-group 239.4.4.4
```


Task 6.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Filtering Fragmented Packets

You may also want to read the following INE blog post for more information on fragmentation issues:

http://blog.ine.com/2008/11/05/dealing-with-fragmented-traffic/

R4:

```
interface FastEthernet0/1
 ip access-group NO_FRAGMENTS out
!
ip access-list extended NO_FRAGMENTS
 deny   ip any any fragments
 permit ip any any
```

🕮 Further Reading

Access Control Lists and IP Fragments:

http://cisco.biz/en/US/tech/tk827/tk369/technologies_white_paper09186a00800949b8.shtml

RFC 1858: Security Consideration for IP Fragment Filtering

Task 6.1 Breakdown

The solution directly reflects the task requirements – it blocks all non-initial fragments, that is packets with Fragment Offset larger than zero. Normal packets and initial fragments are still allowed. Dropping non-initial fragments is common practice when TCP/UDP port information must be validated. For non initial fragments, a packet filter will allow the fragment as long as it matches the L3 information in the filter – that is the source/destination IP addresses. Stateless packet filters cannot check port information in non-initial fragments.

🕮 Deep Dive

- Is it possible to fragment an IP fragment?

Task 6.1 Verification

Temporarily relocate the access-list to R4's Frame-Relay interface. Next issue the "debug ip packet detail" command and ping R4 from R5 using a packet size that will require fragmentation:

```
Rack1R5#ping 150.1.4.4 size 1600
Type escape sequence to abort.
Sending 5, 1600-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
..

Rack1R4#
 IP: recv fragment from 163.1.54.5 offset 0 bytes
 IP: s=163.1.54.5 (Serial0/0/0), d=150.1.4.4, len 120, access denied
     IP Fragment, Ident = 254, fragment offset = 1480
 IP: s=163.1.45.5 (Serial0/1/0), d=224.0.0.13, len 54, rcvd 0,
proto=103
 IP: tableid=0, s=163.1.54.5 (Serial0/0/0), d=150.1.4.4 (Loopback0),
routed via RIB
  IP: s=163.1.54.5 (Serial0/0/0), d=150.1.4.4, len 1500, rcvd 4
     IP Fragment, Ident = 255, fragment offset = 0
     ICMP type=8, code=0
 IP: recv fragment from 163.1.54.5 offset 0 bytes
 IP: s=163.1.54.5 (Serial0/0/0), d=150.1.4.4, len 120, access denied
     IP Fragment, Ident = 255, fragment offset = 1480
```

Note that non-initial fragments are denied. Next, try pinging with packets small enough not to require fragmentation:

```
Rack1R5#ping 150.1.4.4 size 1000

Type escape sequence to abort.
Sending 5, 1000-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 508/508/508
ms
```


Task 6.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > Advanced HTTP Classification with NBAR

R4:

```
ip cef
!
class-map match-all ROOT_EXPLOIT
  match protocol http url "*root.exe*"
!
policy-map SET_DSCP_CS1
 class ROOT_EXPLOIT
  drop
!
interface FastEthernet0/0
 service-policy input SET_DSCP_CS1
```

Task 6.2 Breakdown

As of IOS 12.2(13)T, the **drop** command is a new option in the policy-map. As the command implies, any matching traffic is simply dropped. Prior to this option, traffic could be unconditionally dropped using the **police** keyword with both the conform and exceed actions set to drop.

🕮 Further Reading

Using Network-Based Application Recognition and ACLs for Blocking the "Code Red" Worm

http://www.cisco.com/en/US/products/hw/routers/ps359/products_tech_note09186a00800fc176.shtml

Task 6.2 Verification

To verify the configuration telnet to R6's Loopback on port 80:

```
FRS-BB2>telnet 150.1.6.6 80
Trying 150.1.6.6, 80 ... Open
GET /root.exe HTTP/1.1

<Hit ENTER a few times>
```

Your output should hang if the configuration is working.
Next verify the policy-map on R4:

```
Rack1R4#show policy-map interface Fa0/0
 FastEthernet0/0

  Service-policy input: SET_DSCP_CS1

    Class-map: ROOT_EXPLOIT (match-all)
      6 packets, 410 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol http url "*root.exe*"
      drop
<output omitted>
```

Task 6.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Control Plane Policing

```
R1:
!
! Classify ARP traffic
!
class-map ARP
 match protocol arp
!
ip access-list extended ICMP
 permit icmp any any
!
! Classify ICMP based on ACL
!
class-map ICMP
 match access-group name ICMP
!
! Traffic to be dropped
!
ip access-list extended TELNET
 permit tcp host 163.1.5.5 any eq telnet
!
class-map TELNET
 match access-group name TELNET
!
! Routing packets typically have IPP of 6
!
class-map ROUTING
 match ip precedence 6
!
policy-map CPP_INPUT
 class ARP
  police rate 100 pps
 class TELNET
  drop
 class ROUTING
  police rate 50 pps
!
policy-map CPP_OUTPUT
 class ICMP
  police rate 10 pps
!
control-plane
 service-policy input CPP_INPUT
 service-policy output CPP_OUTPUT
```


Task 6.3 Breakdown

Control plane is the simplest form of control plane protection. Typically, you classify traffic for CPP using access-lists and/or DSCP/IP precedence matching. NBAR classification is not guaranteed to work with CPP at all, with one important exception: **`match protocol arp`** command, which is implemented differently from a typical stateful NBAR classification.

🕮 Deep Dive

- Why rate-limiting ARP messages is an important security measure?

Task 6.3 Verification

Verify control-plane policy map:

```
Rack1R1#ping 163.1.18.8

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 163.1.18.8, timeout is 2 seconds:
!!.!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 1/1/4 ms

Rack1R1#show policy-map control-plane
 Control Plane

  Service-policy input: CPP_INPUT

    Class-map: ARP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol arp
      police:
          rate 100 pps, burst 24 packets
        conformed 0 packets; actions:
          transmit
        exceeded 0 packets; actions:
          drop
        conformed 0 pps, exceed 0 pps

    Class-map: TELNET (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name TELNET
      drop

    Class-map: ROUTING (match-all)
      17 packets, 1682 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: ip precedence 6
      police:
          rate 50 pps, burst 12 packets
        conformed 17 packets; actions:
          transmit
        exceeded 0 packets; actions:
          drop
        conformed 1 pps, exceed 0 pps

    Class-map: class-default (match-any)
      9 packets, 1291 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```



```
  Service-policy output: CPP_OUTPUT

    Class-map: ICMP (match-all)
      10 packets, 1140 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name ICMP
      police:
          rate 10 pps, burst 2 packets
        conformed 8 packets; actions:
          transmit
        exceeded 2 packets; actions:
          drop
        conformed 0 pps, exceed 0 pps

    Class-map: class-default (match-any)
      32 packets, 2835 bytes
      5 minute offered rate 1000 bps, drop rate 0 bps
      Match: any
```

Check that R5 cannot telnet to R1 off it's VLAN5 interface:

```
Rack1R5#telnet 150.1.1.1 /source-interface fastEthernet 0/0
Trying 150.1.1.1 ...
% Connection timed out; remote host not responding

Rack1R5#telnet 150.1.1.1
Trying 150.1.1.1 ... Open


User Access Verification

Password:

Rack24R1#show policy-map control-plane
 Control Plane

  Service-policy input: CPP_INPUT

    Class-map: ARP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol arp
      police:
          rate 100 pps, burst 24 packets
        conformed 0 packets; actions:
          transmit
        exceeded 0 packets; actions:
          drop
        conformed 0 pps, exceed 0 pps

    Class-map: TELNET (match-all)
      4 packets, 192 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name TELNET
      drop
```

Task 7.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

System Management > System Message Logging
System Management > Syslog Logging

R1 - SW2:
```
logging 163.1.5.100
logging 163.1.6.100
service timestamps log datetime msec
```

R2, R4, and R6:
```
logging facility local3
```

R1, R3, and R5:
```
logging facility local4
```

SW1 and SW2:
```
logging facility local5
```

Task 7.1 Breakdown

The only unusual requirement in this task was changing the timestamp format. This is not configured via the “logging” command setting like it would be natural but rather using the “service timestamps” command.

🕮 Deep Dive

- What is another syslog feature to improve syslog message integrity and prevent tampering other than timestamps?

Task 7.1 Verification

Verify basic syslog configuration:

```
Rack1SW2#show logging
Syslog logging: enabled (0 messages dropped, 1 messages rate-limited, 0
flushes, 0 overruns, xml disabled, filtering disabled)
<output omitted>

    File logging: disabled
    Trap logging: level informational, 99 message lines logged
        Logging to 163.1.5.100, 1 message lines logged, xml disabled,
              filtering disabled
        Logging to 163.1.6.100, 1 message lines logged, xml disabled,
```

Verify that milliseconds are included into timestamps:

```
Rack1SW2#show logging  | begin Log Buffer
Log Buffer (4096 bytes):

*Mar  1 08:44:51.441: %SYS-5-CONFIG_I: Configured from console by
console
```

Task 7.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

System Management > NTP

R1 - SW2:

```
ntp source Loopback0
ntp server 204.12.1.254
```

Task 7.2 Verification

Validate NTP status:

```
Rack1SW2#show ntp status
Clock is synchronized, stratum 5, reference is 204.12.1.254
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is
2**18
reference time is AF82175D.3E67403F (07:21:01.243 UTC Fri Apr 23 1993)
clock offset is 1.1874 msec, root delay is 7.68 msec
root dispersion is 1.28 msec, peer dispersion is 0.08 msec

Rack1SW2#show ntp associations

      address  ref clock   st  when  poll reach  delay  offset    disp
*~204.12.1.254 127.127.7.1  4    11    64  377     7.7    1.19     0.1
 * master (synced), # master (unsynced), + selected, - candidate, ~
configured
```

Task 7.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IP Services > DHCP Server

R6:

```
ip dhcp excluded-address 163.1.6.0 163.1.6.127
ip dhcp excluded-address 163.1.6.130
ip dhcp excluded-address 163.1.6.251 163.1.6.255
!
ip dhcp pool VLAN6_POOL
   network 163.1.6.0 255.255.255.0
   default-router 163.1.6.6
   domain-name INE.com
```


Task 7.3 Breakdown

In SOHO environments, the DHCP server functionality of IOS can significantly reduce the administrative overhead of maintaining static IP addressing or dedicating another machine to run DHCP. To enable the DHCP server process, first create a DHCP pool by issuing the **ip dhcp pool [name]** global configuration command. This will bring you to the DHCP server mode.

Once inside the DHCP server mode, specify the address pool by issuing the **network** command. The default gateway is then specified with the **default-router** option, DNS servers with the **dns-server** command, and domain-name with the **domain-name** command.

In order to control which portion of the specified network is assigned, exclude address space that should not be leased out by issuing the **ip dhcp excluded-address [start] [end]** global configuration command.

🕮 Deep Dive

- What is the main difference between DHCP and BOOTP?
- Where is BOOTP still being used in IOS?
- What automated configuration service relies on DHCP in Cisco IOS?

Task 7.3 Verification

Check the global pool settings:

```
Rack1R6#show ip dhcp pool

Pool VLAN6_POOL :
 Utilization mark (high/low)    : 100 / 0
 Subnet size (first/next)       : 0 / 0
 Total addresses                : 254
 Leased addresses               : 0
 Pending event                  : none
 1 subnet is currently in the pool :
 Current index        IP address range                    Leased
addresses
 163.1.6.1            163.1.6.1        - 163.1.6.254       0
```

To verify address allocation, temporarily enable Vlan6 SVI on SW1:

```
Rack1SW1#debug dhcp
Rack1SW1(config)#interface vlan 6
Rack1SW1(config-if)#ip address dhcp

 DHCP: DHCP client process started: 10
 RAC: Starting DHCP discover on Vlan6
 DHCP: Try 1 to acquire address for Vlan6
 DHCP: allocate request
 DHCP: new entry. add to queue
 DHCP: SDiscover attempt # 1 for entry:
 DHCP: SDiscover: sending 298 byte length DHCP packet
 DHCP: SDiscover 298 bytes
            B'cast on Vlan6 interface from 0.0.0.0
 DHCP: Received a BOOTREP pkt
 DHCP: offer received from 163.1.6.6
 DHCP: SRequest attempt # 1 for entry:
 DHCP: SRequest- Server ID option: 163.1.6.6
 DHCP: SRequest- Requested IP addr option: 163.1.6.128
 DHCP: SRequest placed lease len option: 86400
 DHCP: SRequest: 316 bytes
 DHCP: SRequest: 316 bytes
            B'cast on Vlan6 interface from 0.0.0.0
 DHCP: Received a BOOTREP pkt
Interface Vlan6 assigned DHCP address 163.1.6.128, mask 255.255.255.0


 DHCP Client Pooling: ***Allocated IP address: 163.1.6.128
 Allocated IP address = 163.1.6.128  255.255.255.0
```


Verify the DHCP bindings on R6:

```
Rack1R6#show ip dhcp binding
Bindings from all pools not associated with VRF:
IP address          Client-ID/              Lease expiration
Type
                    Hardware address/
                    User name
163.1.6.128         0063.6973.636f.2d30.    Apr 24 1993 07:26 AM
Automatic
                    3030.662e.3866.6232.
                    2e65.3830.302d.566c.
                    36
```

Task 7.4 Solution

🕮 Before You Begin

This topic is not a part of the active CCIE R&S exam blueprint, but it is still worth knowing the basics of IP mobility services.

```
R6:
!
! Enable Mobile ARP responses and control allowed IP addresses
!
interface FastEthernet0/0
 ip mobile arp access-group 2
!
! Redistribute mobile routes into RIP
!
router rip
 redistribute mobile metric 1
!
access-list 2 permit 163.1.5.25
```


Task 7.4 Breakdown

Not to be confused with Mobile IP, Local Area Mobility (LAM) offers a simple way for mobile hosts to roam around the network. When the **ip mobile arp** command is issued on the interface, the LAM process starts listening for ARP requests received on the interface that are from hosts which are not in the IP subnet of that interface. When these requests are received, the LAM processes knows that the ARP came from a mobile host. This hosts IP address is then installed in the IP routing table as a mobile host route. Additionally, ARP requests are sent to the host at a more frequent interval (minutes instead of hours) in order to ensure that they are still on the segment.

The **access-group** option tells the router which hosts to listen for ARP requests from. By default any host on the segment can be mobile. Note that LAM is not very scalable, as each mobile host requires a host route entry in the IP routing table and results in fast growth of IP routing tables.

🕮 Deep Dive

- Why IP address mobility is inherently in-scalable?

Task 7.4 Verification

To verify LAM configure a new interface VL6 SVI on SW1.

```
Rack1SW1#conf t
Rack1SW1(config)#interface vlan 6 Rack1SW1(config-
if)#ip address 163.1.5.25 255.255.255.0
```

Generate ARP packets from SW1:

```
Rack1SW1#ping 163.1.5.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 163.1.5.254, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)
```

Check mobile routes on R6:

```
Rack1R6#show ip route mobile
     163.1.0.0/16 is variably subnetted, 15 subnets, 2 masks
M       163.1.5.25/32 [3/1] via 163.1.5.25, 00:00:51, FastEthernet0/0
```

Verify that the mobile route is redistributed:

```
Rack1R2#show ip route 163.1.5.25
Routing entry for 163.1.5.25/32
  Known via "rip", distance 120, metric 1
  Redistributing via rip, ospf 1
  Advertised by ospf 1 subnets
  Last update from 204.12.1.6 on FastEthernet0/0, 00:00:14 ago
  Routing Descriptor Blocks:
  * 204.12.1.6, from 204.12.1.6, 00:00:14 ago, via FastEthernet0/0
      Route metric is 1, traffic share count is 1
```

Finally do a traceroute from the “mobile” host:

```
Rack1SW1#traceroute 163.1.5.5

Type escape sequence to abort.
Tracing the route to 163.1.5.5

  1  *
    163.1.6.6 0 msec 0 msec
  2 204.12.1.2 0 msec 0 msec 0 msec
  3 163.1.12.1 4 msec 4 msec 4 msec
  4 163.1.15.5 68 msec *  64 msec
```


Task 7.5 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IP Services > Netflow Random Sampling
IP Services > Netflow Aggregation Cache

```
R6:
!
! Define Flow Sampler
!
flow-sampler-map ONE_OUT_OF_THREE
 mode random one-out-of 3
!
! Create a policy-map for flow sampling
!
policy-map NETFLOW
 class class-default
  netflow-sampler ONE_OUT_OF_THREE
!
! Apply netflow sampler
!
interface FastEthernet0/1
 service-policy input NETFLOW
 service-policy output NETFLOW
!
! Define aggregation cache
!
ip flow-aggregation cache source-prefix
 cache entries 2048
 export destination 163.1.6.100 9999
 mask source minimum 16
 enabled
!
ip flow-export version 9
```


Task 7.5 Breakdown

The task has two significant requirements – Netflow cache aggregation and random sampling. The first is configured by creating a new cache and configuring an export destination. The second is accomplished by creating a random netflow sampler and applying it via interface-level service-policies.

🕮 Deep Dive

- Why random netflow sampling is so important?

Task 7.5 Verification

Check the netflow policy map applied to interfaces:

```
Rack1R6#show policy-map interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy input: NETFLOW

    Class-map: class-default (match-any)
      198 packets, 36872 bytes
      5 minute offered rate 1000 bps, drop rate 0 bps
      Match: any
      netflow-sampler: ONE_OUT_OF_THREE

  Service-policy output: NETFLOW

    Class-map: class-default (match-any)
      182 packets, 22200 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      netflow-sampler: ONE_OUT_OF_THREE
```

Issue a ping from R2 with 150 iterations; note that the number of packets accounted on R6 is approximately 3 times less than the number of packets actually send and received.

```
Rack1R2#ping 212.18.2.1 repeat 150

Type escape sequence to abort.
Sending 150, 100-byte ICMP Echos to 212.18.2.1, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!
Success rate is 100 percent (150/150), round-trip min/avg/max =
28/31/49 ms
```


Check the Netflow cache:

```
Rack1R6#show ip cache flow
IP packet size distribution (15024 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416
448  480
   .000 .015 .288 .409 .121 .146 .000 .016 .000 .000 .000 .000 .000
.000 .000

    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .000 .000 .000 .001 .000 .000 .000 .000 .000 .000 .000

IP Flow Switching Cache, 278544 bytes
  2 active, 4094 inactive, 193 added
  4859 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes
  Inactive flows timeout in 15 seconds
IP Sub Flow Cache, 34056 bytes
  2 active, 1022 inactive, 189 added, 189 added to flow
  0 alloc failures, 0 force free
  1 chunk, 10 chunks added
  last clearing of statistics 01:13:09
Protocol      Total  Flows  Packets Bytes Packets Active(Sec) Idle(Sec)
--------      Flows   /Sec    /Flow  /Pkt     /Sec     /Flow     /Flow
TCP-BGP          26    0.0        1    50      0.0       0.0      15.3
UDP-other        98    0.0      120   128      2.6      10.2      14.9
ICMP             47    0.0       63    99      0.6       3.9      15.5
IP-other         20    0.0        7    59      0.0      34.5      14.4
Total:          191    0.0       78   122      3.4       9.8      15.0

SrcIf       SrcIPaddress    DstIf       DstIPaddress    Pr SrcP DstP  Pkts
Fa0/1       204.12.1.2      Vi2         212.18.2.1      01 0000 0800    51
Fa0/1       0.0.0.0         Null        255.255.255.255 11 0044 0043     1
Vi2         212.18.2.1      Fa0/1*      204.12.1.2      01 0000 0000    49
```

Verify export configuration:

```
Rack1R6#show ip flow export
Flow export v9 is disabled for main cache
  Version 9 flow records
  Cache for source-prefix aggregation v8
  VRF ID : Default
    Destination(1)  163.1.6.100 (9999)
    Minimum source mask is configured to /16
  260 flows exported in 124 udp datagrams
  0 flows failed due to lack of export packet
  0 export packets were sent up to process level
  0 export packets were dropped due to no fib
  0 export packets were dropped due to adjacency issues
  0 export packets were dropped due to fragmentation failures
  0 export packets were dropped due to encapsulation fixup failures
```


Check the netflow aggregation cache:

```
Rack1R6#show ip cache flow aggregation source-prefix

IP Flow Switching Cache, 139272 bytes
  3 active, 2045 inactive, 126 added
  2590 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes
  Inactive flows timeout in 15 seconds
IP Sub Flow Cache, 17416 bytes
  3 active, 509 inactive, 126 added, 126 added to flow
  0 alloc failures, 0 force free
  1 chunk, 1 chunk added

Minimum source mask is configured to /16

Src If         Src Prefix      Msk  AS    Flows  Pkts B/Pk  Active
Fa0/1          0.0.0.0         /16  0     1      3  604     29.9
Vi2            212.18.0.0      /16  0     1     53  100      4.6
Fa0/1          204.12.0.0      /16  0     1     48  100      4.6
```

Task 7.6 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

System Management > KRON Command Scheduler

R4:

```
kron occurrence SAVE_DAILY_CONFIG at 6:00 recurring
 policy-list SAVE_RUNNING_CONFIG
!
kron policy-list SAVE_RUNNING_CONFIG
 cli show running-config | redirect tftp://163.1.4.100/r4-config
```


Task 7.6 Breakdown

Note that since the kron scheduler does not support interactive commands, using the command **copy running-config tftp://163.1.4.100/r4-config**" will not work. You may work around this limitation by using EEM policies too.

Task 7.6 Verification

Check the current kron schedule:

```
Rack1R4#show kron schedule
Kron Occurrence Schedule
SAVE_DAILY_CONFIG inactive, will run again in 0 days 10:42:47 at 6 :00
on
```


Task 7.7 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

System Management > Remote Shell

```
R1:
!
! Remote username and local source interface
!
ip rcmd remote-username RCP
ip rcmd source-interface Loopback0

R6:
ip rcmd rcp-enable
ip rcmd rsh-enable
!
ip rcmd remote-host Rack1R6 150.1.1.1 Rack1R1 enable
ip rcmd remote-host RCP 150.1.1.1 Rack1R1 enable
```

Task 7.7 Breakdown

The task directly points toward the use of RCP/RSH commands in the routers. This requires configuring security associations in the target host, mapping the remote username/IP address to the local privileges. The syntax for defining privilege mapping is as following:

ip rcmp remote-host <local-name> <remote-IP> <remote-name> <privilege>

By default, the local name equal to the local router's hostname, unless the remote command explicitly specifies a name like in our case for the "rcp" command.

Task 7.7 Verification

Enable RCMD debugging in R6:

```
Rack1R6#debug ip tcp rcmd
RCMD transactions debugging is on
Rack1R6#

Rack1R1#rsh 150.1.6.6 /user Rack1R6 show run interface Serial 0/0

Building configuration...

Current configuration : 110 bytes
!
interface Serial0/0/0
 no ip address
 encapsulation frame-relay
 frame-relay interface-dlci 201 ppp Virtual-Template1
 no frame-relay inverse-arp
 service-policy input NETFLOW
 service-policy output NETFLOW
end

Rack1R6#
RCMD: [514 <- 150.1.1.1:988] recv \0
RCMD: [514 <- 150.1.1.1:988] recv Rack1R1\0Rack1R6\0show run interface
Serial 0/0/0\0
RCMD: [514 -> 150.1.1.1:988] send <OK>
```

Transfer a file from R6 to R1 using RCP

```
Rack1R6#show flash:

System flash directory:
File  Length   Name/status
<snip>
  2   1941     saved-config
  3   1036     r6i
<snip>
```



```
Rack1R1#copy rcp://150.1.6.6/saved-config null:
Source username [RCP]?
Accessing rcp://RCP@150.1.6.6/saved-config...!
1941 bytes copied in 0.056 secs (34661 bytes/sec)

Rack1R6#
RCMD: [514 <- 150.1.1.1:974] recv Rack1R1\0RCP\0rcp -f saved-config\0
RCMD: [514 -> 150.1.1.1:974] send <OK>
RCMD: [514 <- 150.1.1.1:974] recv <OK>
RCP: [514 -> 150.1.1.1:974] send C0644 1941 saved-config\n
RCMD: [514 <- 150.1.1.1:974] recv <OK>
RCP: [514 -> 150.1.1.1:974] send 1941 bytes
RCMD: [514 -> 150.1.1.1:974] send <OK>
RCMD: [514 <- 150.1.1.1:974] recv <BAD, Write failed>
RCMD: [514 <- 150.1.1.1:967] recv \0
RCMD: [514 <- 150.1.1.1:967] recv Rack1R1\0RCP\0rcp -f saved-config\0
RCMD: [514 -> 150.1.1.1:967] send <OK>
RCMD: [514 <- 150.1.1.1:967] recv <OK>
RCP: [514 -> 150.1.1.1:967] send C0644 1941 saved-config\n
RCMD: [514 <- 150.1.1.1:967] recv <OK>
RCP: [514 -> 150.1.1.1:967] send 1941 bytes
RCMD: [514 -> 150.1.1.1:967] send <OK>
RCMD: [514 <- 150.1.1.1:967] recv <OK>
```


Task 8.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC Classification and Marking

R4:

```
!
! Classify VoIP Signaling and RTP traffic using an ACL
!
ip access-list extended VOIP
 permit tcp any any eq 1720
 permit udp any any range 16384 32767
!
class-map match-all VOIP
 match access-group name VOIP
!
! Mark Traffic on VoIP interfaces
!
policy-map MARK_VOIP
 class VOIP
  set dscp cs5
 class class-default
  set dscp cs1
!
! Policy for non-VoIP interface
!
policy-map SET_DSCP_CS1
 class class-default
  set dscp cs1
!
! VoIP Interface
!
interface FastEthernet0/1
 service-policy input MARK_VOIP
!
! Non-VoIP Interface
!
interface FastEthernet0/0
 service-policy input SET_DSCP_CS1
```



```
R5:
ip access-list extended VOIP
 permit tcp any any eq 1720
 permit udp any any range 16384 32767
!
class-map match-all VOIP
 match access-group name VOIP
!
policy-map MARK_VOIP
 class VOIP
  set dscp cs5
 class class-default
  set dscp cs1
!
policy-map SET_DSCP_CS1
 class class-default
  set dscp cs1
!
interface FastEthernet0/0
 service-policy input MARK_VOIP
!
interface FastEthernet0/1
 service-policy input SET_DSCP_CS1
!
interface Serial0/0/0.35 point-to-point
 service-policy input SET_DSCP_CS1
```

Task 8.1 Breakdown

The task requires classifying and marking VoIP traffic. The classification criteria are clearly outlined in the beginning of the QoS section: marking TCP 1720 and UDP 16384 32767. Although these port ranges are far from being optimal match, especially for RTP traffic, this is what we have in the task. Based on this information, we use access-list for traffic classification. We could have used NBAR, but the explicit mentioning of the port information prompts toward using the access-lists.

We also create a special policy-map for non-VoIP interfaces. Why we could have used the same policy on those, it makes sense to create a new one, which achieves the same result and offers better “security”.

Task 8.1 Verification

Verify the policy-map configuration (note the QoS set operations):

```
Rack1R4#show policy-map interface
 FastEthernet0/0

  Service-policy input: SET_DSCP_CS1

    Class-map: class-default (match-any)
      1 packets, 118 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      QoS Set
        dscp cs1
          Packets marked 1
 FastEthernet0/1

  Service-policy input: MARK_VOIP

    Class-map: VOIP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name VOIP
      QoS Set
        dscp cs5
          Packets marked 0

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      QoS Set
        dscp cs1
          Packets marked 0
```

Check access-list used for classification purposes:

```
Rack1R4#show ip access-list VOIP
Extended IP access list VOIP
    10 permit tcp any any eq 1720
    20 permit udp any any range 16384 32767
```


Task 8.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > Legacy Frame-Relay Traffic Shaping
QoS > Legacy FRTS with Per-VC Fragmentation

R3:
```
interface Serial1/0
 frame-relay traffic-shaping
 frame-relay interface-dlci 305
  class DLCI_305
!
map-class frame-relay DLCI_305
 frame-relay cir 768000
```

R4:
```
interface Serial0/0/0
 frame-relay traffic-shaping
 frame-relay interface-dlci 405
  class DLCI_405
!
map-class frame-relay DLCI_405
 frame-relay cir 768000
 frame-relay bc 7680
 frame-relay fragment 960
```

R5:
```
interface Serial0/0/0
 frame-relay traffic-shaping
!
interface Serial0/0/0.35 point-to-point
 frame-relay interface-dlci 503
  class DLCI_503
!
interface Serial0/0/0.54 point-to-point
 frame-relay interface-dlci 504
  class DLCI_504
!
map-class frame-relay DLCI_504
 frame-relay cir 768000
 frame-relay bc 7680
 frame-relay fragment 960
!
map-class frame-relay DLCI_503
 frame-relay cir 768000
```


Task 8.2 Breakdown

Frame Relay fragmentation allows for minimal delay when sending small real time packets across a Frame Relay circuit, such as VoIP. As previously discussed, a router always sends traffic out an interface at the serialization, or clocking, of that interface. Typically this results in periods of traffic bursts, followed by periods of no activity. Traffic shaping attempts to smooth these peaks and valleys out by pacing the output of packets on the interface. However, as the transmit ring (hardware queue) of an interface is always first in first out (FIFO), small real time packets can experience unacceptable delay when they are stuck behind large data packets, even when priority queueing is enabled.

By reducing the maximum packet size that can be sent out the Frame Relay circuit, fragmentation reduces the worst case delay that a real time packet must endure. Additionally, by enabling fragmentation we activate interface-level priority queue and interleaving, which ensures voice packets are correctly interleaved with the fragments of large data packets. Typically the fragmentation size is set based on the port physical rate, but this task specifies and explicit fragment size.

In real world configurations, you normally enable fragmentation at per-interface level, not just per-VC. The reason being is that unfragmented packets on another VC will block VoIP packets on the fragmenting VC. However, the scenario explicitly requires configuring fragmentation only between R4 and R5.

Task 8.2 Verification

Verify shaping & fragmentation configuration:

```
Rack1R5#show frame-relay pvc 504

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

DLCI = 504, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE (EEK UP), INTERFACE
= Serial0/0/0.54

<output omitted>

  Queueing strategy: weighted fair
  Current fair queue configuration:
   Discard     Dynamic      Reserved
   threshold   queue count  queue count
    64          16           0
  Output queue size 0/max total 600/drops 0
  fragment type end-to-end fragment size 960
  cir 768000    bc   7680      be 0         limit 960    interval 10
  mincir 384000    byte increment 960   BECN response no  IF_CONG no
  frags 107       bytes 10122     frags delayed 0       bytes delayed 0
  shaping inactive
  traffic shaping drops 0
```


Task 8.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC Classification and Marking
QoS > Using MQC with Legacy FRTS

R4:

```
!
! Mark all traffic but no CS5 with the DE bit
!
class-map match-all DSCP_CS5
 match  dscp cs5
!
policy-map FR_QOS
 class DSCP_CS5
!
! Set DE flag for all traffic but CS5
!
 class class-default
  set fr-de
!
map-class frame-relay DLCI_405
 service-policy output FR_QOS
```

R5:

```
class-map match-all DSCP_CS5
 match  dscp cs5
!
policy-map FR_QOS
 class DSCP_CS5
 class class-default
  set fr-de
!
map-class frame-relay DLCI_504
 service-policy output FR_QOS
```


Task 8.3 Breakdown

The Frame Relay discard-eligible bit tells the Frame Relay switches in the provider cloud which frames to drop first when congestion occurs. By manually setting the DE bit on non essential traffic, real time traffic such as VoIP is less likely to be dropped in the event of congestion.

Be careful with setting the DE bit in class-default when applying policy to a physical interface. This will affect LMI traffic and may break connectivity.

🕮 Deep Dive

- What other traffic you may want to exclude from DE marking other than VoIP?

Task 8.3 Verification

Verify DE-marking configuration:

```
Rack1R5#show frame-relay pvc 504

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

DLCI = 504, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE (EEK UP), INTERFACE
= Serial0/0/0.54

<output omitted>

 Serial0/0/0.54: DLCI 504 -

  Service-policy output: FR_QOS

    Class-map: DSCP_CS5 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps
      Match:  dscp cs5 (40)

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      QoS Set
        fr-de
          Packets marked 1
  <output omitted>
```


Task 8.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC Classification and Marking
QoS > MQC LLQ and Remaining Bandwidth Reservation

R4:

```
!
! Apply LLQ to previously defined policy-maps
!
policy-map FastEthernet_QOS
 class DSCP_CS5
  priority 256
!
! FR PVC Policy (already applied)
!
policy-map FR_QOS
 class DSCP_CS5
  priority 256
!
interface FastEthernet0/1
 service-policy output FastEthernet_QOS
```

R5:

```
policy-map FastEthernet_QOS
 class DSCP_CS5
  priority 256
!
policy-map FR_QOS
 class DSCP_CS5
  priority 256
!
interface FastEthernet0/0
 service-policy output FastEthernet_QOS
```


Task 8.4 Breakdown

The **priority 256** command ensures that all VoIP traffic up to 256Kbps in the output queue is dequeued first. This prioritization ensures the minimum amount of delay for priority packets as they sit in the output queue. The LLQ scheduler only triggers when there is congestion on the interface, i.e. when the TX-ring is full.

🕮 Deep Dive

- Is is possible to define LLQ without a bandwidth value?

Task 8.4 Verification

Verify LLQ configuration:

```
Rack1R5#show frame-relay pvc 504

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

DLCI = 504, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE (EEK UP), INTERFACE
= Serial0/0/0.54

<output omitted>

  service policy FR_QOS
 Serial0/0/0.54: DLCI 504 -

  Service-policy output: FR_QOS

    Class-map: DSCP_CS5 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:  dscp cs5 (40)
      Queueing
        Strict Priority
        Output Queue: Conversation 40
        Bandwidth 256 (kbps) Burst 6400 (Bytes)
        (pkts matched/bytes matched) 0/0
        (total drops/bytes drops) 0/0

    Class-map: class-default (match-any)
      172 packets, 16103 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      QoS Set
        fr-de
          Packets marked 172
<output omitted>
```



```
Rack1R5#show policy-map interface fastEthernet 0/0 output
 FastEthernet0/0

  Service-policy output: FastEthernet_QOS

    queue stats for all priority classes:

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

    Class-map: DSCP_CS5 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:  dscp cs5 (40)
      Priority: 256 kbps, burst bytes 6400, b/w exceed drops: 0


    Class-map: class-default (match-any)
      52 packets, 4157 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 51/4557
```


VOL1 Scenario Reference

Bridging and Switching > 802.1q Tunneling
Bridging and Switching > Layer 2 Etherchannel
Bridging and Switching > Storm Control
Bridging and Switching > Voice VLAN
Bridging and Switching > Private VLANs
Frame Relay > Frame Relay End-to-End Keepalives
RIPv2 > RIPv2 Filtering with Prefix-List
RIPv2 > RIPv2 Manual Summarization
RIPv2 > RIPv2 Filtering with Standard Access-List
OSPF > Repairing Discontiguous OSPF Areas with Virtual Link
OSPF > OSPF Network Type Non-Broadcast
OSPF > OSPF Filtering with Distribute Lists
OSPF > OSPF Internal Summarization
BGP > Establishing iBGP Peering
BGP > Establishing eBGP Peering
BGP > Multihop eBGP Peering
BGP > iBGP Confederations
BGP > BGP Bestpath Selection – Local Preference
BGP > BGP Next-Hop Processing – Next-Hop-Self
BGP > BGP Fast Fallover
BGP > BGP Aggregation
BGP > BGP Aggregation – Summary-Only
BGP > BGP Aggregation – Unsuppress Map
BGP > BGP Communities – No-Export
IPV6 > Link Local Addressing
IPv6 > Unique Local Addressing
IPv6 > OSPFv3

Multicast > PIM Dense Mode
Multicast > Multicast RPF Failure
Security > Filtering Fragmented Packets
QoS > Advanced HTTP Classification with NBAR
Security > Control Plane Policing
System Management > System Message Logging
System Management > Syslog Logging
System Management > NTP
IP Services > DHCP Server
IP Services > Netflow Random Sampling
IP Services > Netflow Aggregation Cache
System Management > KRON Command Schedule
System Management > Remote Shell
QoS > MQC Classification and Marking
QoS > Legacy Frame-Relay Traffic Shaping
QoS > Legacy FRTS with Per-VC Fragmentation
QoS > MQC Classification and Marking
QoS > Using MQC with Legacy FRTS
QoS > MQC Classification and Marking
QoS > MQC LLQ and Remaining Bandwidth Reservation

Deep Dive Answers

- Why do you need VTP Transparent Mode with Private VLANs?
    - Private VLANs mapping information is not passed using VTPv1/VTPv2. Therefore, configuring Private VLANs is not supported when switch is configured for VTP. There is support for extended attributes in VTPv3 and this supports Private VLANs configuration propagation.
- What other Layer 2 down detection mechanisms are available with Frame-Relay?
    - The main, local detection mechanism is LMI signaling. LMI messages have the special A (active) bit set for every PVC entry. It is possible to signal this bit through a FR network, propagating PVC status end-to-end but many implementations do not support this feature.
- What other way you can use in RIP to filter based on route source other than gateway filtering?
    - You may use extended access-lists. In this case, the first section of the entry (net/mask) specifies the filtered network and the second specifies the allowed advertising gateways. This is one of the early “hacks” that allowed for gateway-based filtering.
- How can routing loops form in OSPF transit area if you allow summarizing into it?
    - A transit area has multiple ABRs connected to area 0. If you let summarize routes in the area, only a single ABR may be mistakenly configured for summarization. The transit area will then select the other ABR which will in turn attempt to transit the same area again. This results in permanent routing loops. While this problem could be solved with consistent summarization, OSPF designers considered it being serious enough to automatically disable summarization into transit area.
- Why OSPF does not summarize inter-area routes and only applies summarization to intra-area prefixes?
    - OSPF only summarizes intra-area routes to avoid routing inconsistency. By OSPF design, you should summarize INTO area 0 and does not apply summarization to already summarized routes. The problem is that you may inject inconsistent routing information into non-backbone area and force the ABR to select incorrect routing path transit through a non-backbone area. To avoid such complications and possible loops, OSPF poses such restriction.

- How is control-plane routing loop different from data-plane loop?
    - Control plane routing loop is formed when routing information originating in the domain is re-injected back in the domain, e.g. by means of redistribution. Such information could not be verified to be loop free, unless it has some attributes to trace the control-plane path. For example, BGP tracks the AS path to detect potential routing loops. Data-plane routing loops occur when routers have inconsistent forwarding information. For example router R1 may user R2 as next hop for prefix X, while R2 may think R1 is the next hop for the same prefix. This could be a result of re-convergence, where one router has updated its routing tables while the other has not. Another problem causing such behavior could be inconsistent summarization, e.g. summarizing without a Null0 route.
- Why changing local preference to affect outbound paths is better than locally manipulating AS_PATHs?
    - Local Preference changes affect only the local AS and not any other systems upstream or downstream. Changing the AS_PATH attribute will affect the downstream AS's as well. The same does not apply to changing the MED value, but playing with MEDs may result in routing loops due to non-deterministic path selection.
- Is it safe to change a next-hop on iBGP route reflectors?
    - Only if you need to bring the reflector in the forwarding path, which normally signalize an suboptimal design. Changing next-hop in route-reflector environment could be dangerous, as iBGP peers do not posses the same amount of routing information. Modifying traffic forwarding may result in permanent data-plane routing loops.
- Why isn't fast fallover enabled for iBGP sessions?
    - Internal BGP session normally source off logical interfaces, e.g. Loopbacks and use IGP mechanics to route over a failure. eBGP sessions are typically created over physical links and benefit from fast down detection, triggering BGP re-convergence.
- What other solutions could AS 300 and AS 100 use to minimize routing table sizes in AS 100 and keep AS 300 fully informed of the prefixes behind AS 54?
    - In the task, summarization has been performed in AS 100. Normally, you want to summarize as close to the point that requires that as possible. In this particular scenario, it would be beneficial to make AS 300 send default route to AS 200 and maintain all information learned from AS 100. Outbound route filters could be used if AS 300 wants to control route filtering in AS 100 peers.

- Is it possible to implement OSPFv3 traffic engineering in a single area on per-prefix basis?
    - No, it is only possible to affect traffic flows by modifying OSPF link costs, which affects the whole traffic matrix. You may perform limited per-prefix traffic engineering by splitting network into multiple areas and performing summarization.
- What is the main architectural difference between OSPFv2 and OSPFv3?
    - OSPFv3 decoupled addressing (reachability) information from topological, routing information. There is no topological LSA that carries address information anymore. For example, link addresses have been removed from LSA type 1 and moved into a separate LSA. This change potentially makes OSPFv3 independent of address family used, similar to IS-IS. Additionally, any change in addressing is no longer considered a topological event and does not cause SPF re-computation.
- Is it possible to fragment an IP fragment?
    - Yes, this is possible if the fragment is large enough to allow for sub-fragmentation. The payload is split and the fragment offset field is modified in new fragments. The same fragment ID is retained in the new fragments, allowing them to be properly coalesced with other fragments.
- Why rate-limiting ARP messages is an important security measure?
    - ARP messages form the normal broadcast background on a typical Ethernet segment. All these packets are processes by router's CPU to build the adjacency table. In case of Layer 2 loop and broadcast storm or intentional ARP flood the router CPU could be hogged down and the router may become unmanageable.
- What is another syslog feature to improve syslog message integrity and prevent tampering other than timestamps?
    - In addition to using timestamp, syslog messages may have sequence numbers assigned to them, therefore allowing for proper order tracking. Removing a single message from a syslog file becomes complicated, as it requires properly renumbering all following messages.
- What is the main difference between DHCP and BOOTP?
    - BOOTP supports two main functions – obtaining and IP address and boot image name. It does not support flexible end node configuration like DHCP does, but it support IP forwarding and relay agents (unlike RARP).

- Where is BOOTP still being used in IOS?
  - Auto-Install over Frame-Relay is one good example where BOOTP is used for address configuration and image download.
- What automated configuration service relies on DHCP in Cisco IOS?
  - AutoInstall over Ethernet links support full-blown DHCP configuration, including DNS server and default-gateway auto-configuration..
- Why IP address mobility is inherently in-scalable?
  - IP addressing plan was not designed to be mobile. IP scalability is based on address hierarchies, which means a host may not easily change its location without changing an IP address. Preserving IP addresses during move results in proliferation of long unsummarizable prefixes.
- Why random netflow sampling is so important?
  - Random sampling allows for greatly reducing the amount of flow information processed and exported to the collectors. At the same time, it preserves traffic proportions among flows and allows for efficient capacity planning. Of course, random sampling could not be used for billing systems.
- What other traffic you may want to exclude from DE marking other than VoIP?
  - Normally, you want to be sure that routing control and VoIP traffic is never marked as DE eligible. While best way to ensure this is sending at CIR and properly servicing the queues, the DE-hack could be used when the router is sending traffic at PIR rate, hoping to affect marking decisions in the SP core.
- Is it possible to define LLQ without a bandwidth value?
  - It is possible in HQF implementation. A queue could be defined as simple priority queue, which allows it for getting practically all bandwidth resources when needed. CBWFQ mandates the use of policing value with the LLQ settings.

Lab 8 Solutions

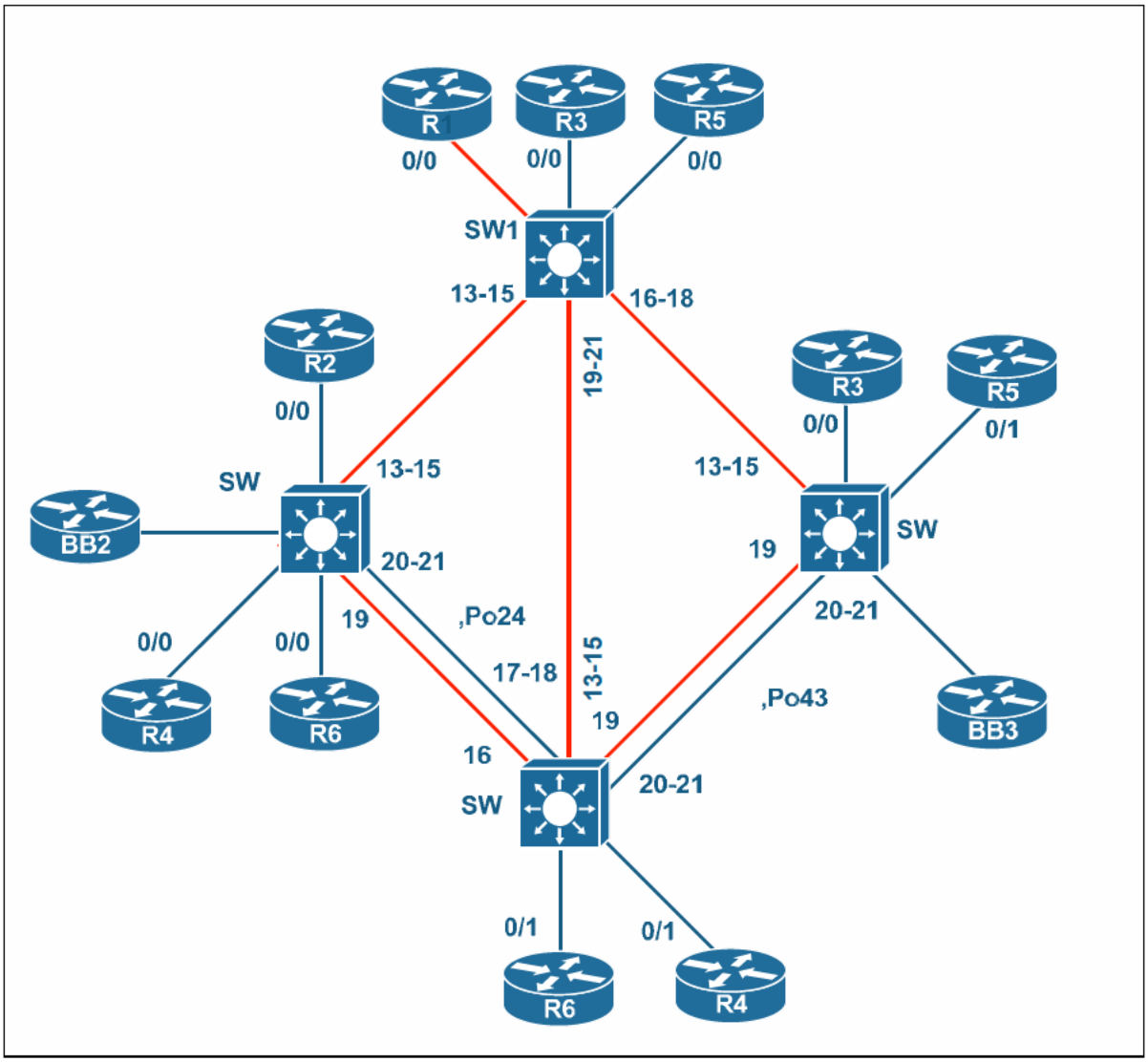
Task 1.1 and 1.4 Solution

🕮 Before You Begin

These tasks assume preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

```
Bridging and Switching > MST Root Bridge Selection
Bridging and Switching > MSTP and Rapid Spanning Tree
```

Prior to starting with Layer 2 configurations, we create a Layer 2 topology diagram as show below. The diagram is desirable as further tasks require layer 2 traffic engineering, which requires knowledge of layer 2 connectivity. The diagram displays trunk links in red color to distinguish them from other links. Notice that there are some port-channels pre-configured already for you.


R
R3
R5
0/0
0/0
0/0
SW1
13-15
16-18
19-21
R2
0/0
SW
13-15
BB2
20-21
19
,Po24
0/0
0/0
R4
R6
17-18
13-15
19
16
SW
20-21
0/1
0/1
R6
R4
R3
R5
0/0
0/1
13-15
SW
19
20-21
,Po43
BB3

SW1, SW2, SW3, and SW4:

```
!
! Configure all switches for MST and map VLANs to instances
!
spanning-tree mode mst
!
spanning-tree mst configuration
instance 1 vlan 3-7
instance 2 vlan 13-45
instance 3 vlan 52-67
instance 4 vlan 1,1001
```

SW1:

```
!
! Make SW1 root for instance #1
!
spanning-tree mst 1 root primary
```

SW2:

```
!
! Make SW1 root for instance #2
!
spanning-tree mst 2 root primary
```

SW3:

```
!
! Make SW1 root for instance #3
!
spanning-tree mst 3 root primary
```

SW4:

```
!
! Make SW1 root for instance #4
!
spanning-tree mst 4 root primary
```


Task 1.1 and 1.4 Breakdown

Carefully reading through the section you should notice that Task 1.1 and Task 1.4 should be completed together. The first task specifies the root bridge for VLAN groups, the second one defines the VLAN grouping and instance mapping. From the task wording it is obvious that both MSTP should be used as the flavor of spanning-tree. There are no advanced requirements like multi-region configurations, or interoperations. The scenarios require creating four MSTP instances in a single region.

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

Pitfall

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

Deep Dive

- Can you load-balance VLANs on the links connecting different MST regions?

Tasks 1.1 and 1.4 Verification

Verify MISTP roots for every instance:

```
Rack1SW1#show spanning-tree mst 1

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

Rack1SW2#show spanning-tree mst 2

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

Rack1SW3#show spanning-tree mst 3

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

Rack1SW4#show spanning-tree mst 4

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


Task 1.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Bridging and Switching > 802.1q Tunneling
Bridging and Switching > 802.1q Native VLAN

```
SW1, SW2, SW3, and SW4:
!
! We configured all switches for MTU change to account for
! different bridging paths for VLAN1 in case of primary path failure
!
system mtu 1504
system mtu routing 1500
vlan dot1q tag native
<reload>

SW2:
interface FastEthernet0/2
 switchport mode dot1q-tunnel

SW4:
interface FastEthernet0/6
 switchport mode dot1q-tunnel
```

☞ Quick Note
Although not needed on SW3 for this task, SW3 will be running OSPF with SW4 later in the lab. By ensuring that the MTUs match, it will alleviate any potential problems. Tagging native VLAN is needed to hide the innermost VLAN header for VLAN26, which is not configured in any switch. Notice that the 3550s (SW3, SW4) don't support the **system mtu routing** command.

Task 1.2 Breakdown

The requirement to provide connectivity between devices connected to different switches. This normally requires a separate VLAN, and both R2 and R6 are said to be configured for tagged subinterface in VLAN 26. However, the other requirement says that VLAN26 should not exist anywhere in the switches. The only option left is QinQ tunneling, but the task prohibts adding any new metro VLANs in the switches.

The solution is relying on any of the existing VLANs for tunneling. Since non of the connections use VLAN1, we may use it as a metro tag. However, this is the default native VLAN on all ports, and therefore frames in VLAN 1 are not tagged. To overcome this limitation, we enable VLAN1 tagging on all ports and configure the tunneling ports connected to R2 and R6 in the default access VLAN 1.

Task 1.2 Verification

Make sure VLAN 26 is not in the switches

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

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

Check the tunnel settings

```
Rack1SW2#show dot1q-tunnel interface fa0/2

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

Rack1SW4#show dot1q-tunnel interface fa0/6

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

Check connectivity

```
Rack1R2#ping 174.1.26.6

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


Task 1.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Bridging and Switching > MST Load Balancing with Port Priority

SW1:

```
!
! SW1 is the root bridge for MSTI 1
! Therefore we can change port priorities
! to affect downstream port blocking selection
!

!
! Links to SW2.
!
interface FastEthernet0/14
 spanning-tree mst 1 port-priority 32
!
interface FastEthernet0/15
 spanning-tree mst 1 port-priority 16

!
! Links to SW3
!
interface FastEthernet0/17
 spanning-tree mst 1 port-priority 32
!
interface FastEthernet0/18
 spanning-tree mst 1 port-priority 16

!
! Links to SW4
!
interface FastEthernet0/20
 spanning-tree mst 1 port-priority 32
!
interface FastEthernet0/21
 spanning-tree mst 1 port-priority 16
```


Task 1.3 Breakdown

This task requires prior knowledge of the active links connecting SW1 to other bridge. We've done that discovery earlier when configuring the first scenario in Layer 2 Technologies. Next, we need to keep in mind that VLANs 3-7 map to MSTI instance 1, which has SW1 as the root bridge.

Based on this information and the fact that we are only allowed to modify SW1 in this scenario, our only option to affect path selection by the downstream bridges is playing with bridge priority on SW1. We configure the priority values so that lower numbered links have higher priority and therefore less preferred in the role of the root ports for MSTI instance 1. We only change the priorities on the "higher" numbered ports, as the "lowest" number port will use the default priority of 128 and always lose to other ports.

Task 1.3 Verification

Walk through all downstream bridges and check the root ports for MSTI instance #1:

```
Rack1SW2#show spanning-tree mst 1

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

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

Rack1SW3#show spanning-tree mst 1

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

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

Rack1SW4#show spanning-tree mst 1

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

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


Task 1.4 Solution

Note

Solution and verification provided in **Task 1.1**

Task 1.5 Solution

Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Frame-Relay > PPP over Frame-Relay
QoS > MLP Link Fragmentation and Interleaving
QoS > MLPPP LFI over Frame-Relay

R2:

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


R3:

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


Task 1.5 Breakdown

Since Frame Relay does not natively support features such as authentication, link quality monitoring, and reliable transmission, it is sometimes advantageous to run PPP over Frame Relay in order to enable these features. A feature of PPP that can be used over Frame Relay as well is PPP multilink. The main purpose of this feature is enabling VoIP fragmentation scheme, compatible with other L2 mediums, mainly ATM. Another added feature is the ability to bond multiple physical or logical Frame-Relay links together into a single link. This configuration is aptly named Multilink PPP over Frame Relay (MLPoFR).

Note

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

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

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

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

Note

The IOS CLI may complain on the lack of MQC LLQ configurations and Frame-Relay traffic shaping, as these are assumed to be part of MLPPP configuration to provide effective VoIP fragmentation and interleaving. However, for the purpose of providing end-to-end connectivity QoS configuration could be skipped and the solution would still be functional.

🕮 Deep Dive

- What could be your option for providing interleaving and fragmentation over DSL lines?

Task 1.5 Verification

Check the multilink bundles in the system

```
Rack1R2#show ppp multilink

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

Test end-to-end connectivity:

```
Rack1R2#ping 174.1.23.3

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


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

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

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

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

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

Rack1R3(config-if)#do show frame pvc 302

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

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

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


Task 2.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

OSPF > OSPF Cleartext Authentication

R2:

```
!
! Authenticate adjacency to R6
!
interface FastEthernet0/0.26
 ip ospf authentication
 ip ospf authentication-key CISCO
!
! Enable OSPF and advertise networks
!
router ospf 1
 router-id 150.1.2.2
 network 150.1.2.2 0.0.0.0 area 0
 network 174.1.23.2 0.0.0.0 area 0
 network 174.1.26.2 0.0.0.0 area 0
```

R3:

```
router ospf 1
 router-id 150.1.3.3
 network 150.1.3.3 0.0.0.0 area 0
 network 174.1.23.3 0.0.0.0 area 0
```

R6:

```
!
! Authenticate adjacency to R2
!
interface FastEthernet0/1.26
 ip ospf authentication
 ip ospf authentication-key CISCO
!
router ospf 1
 router-id 150.1.6.6
 network 174.1.26.6 0.0.0.0 area 0
```


Task 2.1 Breakdown

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

🕮 Deep Dive

- How would authentication differ if OSPF was authenticate LSA contents, not packets?

Task 2.1 Verification

Verify the OSPF neighbors:

```
Rack1R6#show ip ospf neighbor

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

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

```
Rack1R6#show ip ospf interface fa0/1.26
FastEthernet0/1.26 is up, line protocol is up
  Internet Address 174.1.26.6/24, Area 0
  Process ID 1, Router ID 150.1.6.6, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State BDR, Priority 1
  Designated Router (ID) 150.1.2.2, Interface address 174.1.26.2
  Backup Designated router (ID) 150.1.6.6, Interface address 174.1.26.6
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:07
  Supports Link-local Signaling (LLS)
  Index 1/1, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.2.2  (Backup Designated Router)
  Suppress hello for 0 neighbor(s)
  Simple password authentication enabled
```


Task 2.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > OSPF Internal Summarization

R6:

```
!
! Switches have been configured for QinQ
! To avoid OSPF adjacency failures it is worth
! making OSPF ignore the neighbor MTU setting
!
interface FastEthernet0/0
 ip ospf mtu-ignore
!
router ospf 1
 network 150.1.6.6 0.0.0.0 area 67
 network 174.1.67.6 0.0.0.0 area 67
 area 67 range 150.1.6.0 255.255.254.0
```

SW1:

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


Task 2.2 Breakdown

Firstly, the **ip ospf mtu-ignore** command is not really necessary in this task, as SW1 has been already configured for the routing MTU of 1500. However, keep in mind that this command will also resolve the OSPF adjacency issue due to MTU mismatch.

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

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

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

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

🕮 Deep Dive

- In absence of MPLS, what other major drawback of summarization could you point out, not mentioning suboptimal routing?

Task 2.2 Verification

Verify OSPF adjacency:

```
Rack1SW1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.6.6         1   FULL/BDR        00:00:38    174.1.67.6      Vlan67
```

Verify the summary route generation:

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

Task 2.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

RIPv2 > RIPv2 Authentication

```
R5:
!
! Configure RIPv2 authentication
!
key chain RIP
 key 1
  key-string CISCO
!
interface FastEthernet0/1.52
 ip rip authentication mode md5
 ip rip authentication key-chain RIP
!
router rip
 version 2
 network 192.10.1.0
```

Task 2.3 Breakdown

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

🕮 Deep Dive

- Do key indexes have to match for RIPv2 authentication to succeed?

Task 2.3 Verification

Verify the RIP configuration:

```
Rack1R5#show ip protocols | begin rip
Routing Protocol is "rip"
  Sending updates every 30 seconds, next due in 12 seconds
  Invalid after 180 seconds, hold down 180, flushed after 240
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Redistributing: rip
  Default version control: send version 2, receive version 2
    Interface             Send  Recv  Triggered RIP  Key-chain
    FastEthernet0/1.52       2     2                    RIP
  Automatic network summarization is in effect
  Maximum path: 4
  Routing for Networks:
    192.10.1.0
  Routing Information Sources:
    Gateway         Distance      Last Update
    192.10.1.254         120      00:00:20
  Distance: (default is 120)
```

Verify the incoming RIP updates:

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

Rack1R5#show ip route rip
R  222.22.2.0/24 [120/7] via 192.10.1.254, 00:00:07, FastEthernet0/1.52
R  220.20.3.0/24 [120/7] via 192.10.1.254, 00:00:07, FastEthernet0/1.52
R  205.90.31.0/24 [120/7] via 192.10.1.254, 00:00:07,FastEthernet0/1.52
```

Verify the key-chain configuration:

```
Rack1R5#show key chain RIP
Key-chain RIP:
    key 1 -- text "CISCO"
        accept lifetime (always valid) - (always valid) [valid now]
        send lifetime (always valid) - (always valid) [valid now]
```


Task 2.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

EIGRP > EIGRP Split Horizon
OSPF > OSPF Stub Areas

R1:

```
!
! The following command is needed
! to allow for EIGRP route propagation
! between the spokes
!
interface Serial0/0
 no ip split-horizon eigrp 1024
```

R1:

```
!
! Redistribute OSPF again as R1 will ignore EIGRP external
! information until OSPF link to SW1 is broken
!
router eigrp 1024
 redistribute ospf 1 metric 10000 10 255 1 1500
```

R3:

```
!
! Select R3 as the point of redistribution.
!
router ospf 1
 redistribute eigrp 1024 subnets
!
router eigrp 1024
 redistribute ospf 1 metric 10000 10 255 1 1500
```

R5:

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


Task 2.4 Breakdown

The redistribution scenario is not complicated, but requires some considerations. EIGRP touches OSPF on two routers, R1 and R3. R1 is only attached to OSPF Area 38, which is a stub area, as configured initially. Therefore, it makes little sense redistributing EIGRP routes into OSPF on R1. At the same time, it may seem not worth redistributing OSPF routers on R1 into EIGRP – you may perform the same procedure on R3, and obtain more detailed routing information. However, if you redistribute OSPF into EIGRP on R3, R1 will be able to reach Area 38 routes by the virtue of the fact that EIGRP external distance is higher than OSPF's. On the negative side, this will also prevent EIGRP learned external prefixes from propagating further into EIGRP domain – EIGRP only propagates prefixes if they are in the routing table. In order to fix this, we need to redistribute OSPF into EIGRP once again on R1.

Based on the above facts, we select R3 as the point of mutual redistribution and additionally inject OSPF prefixes into EIGRP on R1. In addition to configuring this, we also disable split horizon on R1, which was preventing EIGRP route exchange between R4 and R5 in case of the backdoor (VLAN45) link failure. While it's not strictly necessary in this particular, it is still worth doing to provide extra redundancy in EIGRP domain. Besides, you will see that other scenarios rely on multipath in the EIGRP cloud which requires split horizon to be disabled.

🕮 Deep Dive

- What is third-party next-hop?
- Does RIPv2 support third-party next-hop?

Task 2.4 Verification

Verify that RIP prefixes are being redistributed:

```
Rack1R1#show ip route eigrp | i D EX
D EX 222.22.2.0/24 [170/2172416] via 174.1.145.5, 00:00:56, Serial0/0
D EX 220.20.3.0/24 [170/2172416] via 174.1.145.5, 00:00:56, Serial0/0
D EX    174.1.23.2/32 [170/2172416] via 174.1.13.3, 00:00:15, Serial0/1
D EX 192.10.1.0/24 [170/2172416] via 174.1.145.5, 00:00:56, Serial0/0
D EX 205.90.31.0/24 [170/2172416] via 174.1.145.5, 00:00:56, Serial0/0
```

Verify that split-horizon is disabled for EIGRP on R1's Serial0/0 interface:

```
Rack1R4#show ip eigrp topology 192.10.1.0
IP-EIGRP (AS 1024): Topology entry for 192.10.1.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is
261120
  Routing Descriptor Blocks:
  174.1.45.5 (FastEthernet0/1), from 174.1.45.5, Send flag is 0x0
      Composite metric is (261120/258560), Route is External
      Vector metric:
        Minimum bandwidth is 10000 Kbit
        Total delay is 200 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
      External data:
        Originating router is 150.1.5.5
        AS number of route is 0
        External protocol is RIP, external metric is 0
        Administrator tag is 0 (0x00000000)
  174.1.145.1 (Serial0/0/0), from 174.1.145.1, Send flag is 0x0
      Composite metric is (2684416/2172416), Route is External
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 40100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
      External data:
        Originating router is 150.1.5.5
        AS number of route is 0
        External protocol is RIP, external metric is 0
        Administrator tag is 0 (0x00000000)
```


Verify that mutual redistribution between OSPF and EIGRP on R3 is successful:

```
Rack1R4#show ip route 174.1.26.0
Routing entry for 174.1.26.0/24
  Known via "eigrp 1024", distance 170, metric 2172416, type external
  Redistributing via eigrp 1024
  Last update from 174.1.145.1 on Serial0/0/0, 00:17:06 ago
  Routing Descriptor Blocks:
  * 174.1.145.1, from 174.1.145.1, 00:17:06 ago, via Serial0/0/0
      Route metric is 2172416, traffic share count is 1
      Total delay is 20100 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1

Rack1R2#show ip route 192.10.1.0
Routing entry for 192.10.1.0/24
  Known via "ospf 1", distance 110, metric 20, type extern 2, forward
metric 1
  Last update from 174.1.23.3 on Multilink1, 00:17:31 ago
  Routing Descriptor Blocks:
  * 174.1.23.3, from 150.1.3.3, 00:17:31 ago, via Multilink1
      Route metric is 20, traffic share count is 1
```

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

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

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

Task 2.5 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

EIGRP > Unequal Cost Load Balancing

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

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

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


Task 2.5 Breakdown

This is the task that requires EIGRP split horizon to be disabled on R1, so that R4 may learn alternative routes to the networks behind BB2. EIGRP is the only protocol that supports unequal cost load balancing. Once this feature is enabled, the router distributes traffic proportionately to the ratios of the metrics that are associated with different paths. In order to achieve the 4:1 ratio for traffic, we need to modify metrics so the metric ratio becomes 4:1. That is, on R4, the path through the Ethernet link should have a metric which is four times better than the path through the Frame-Relay connection. Since by default, EIGRP takes into consideration delay and bandwidth we can modify any combinations of these in order to achieve the 4:1 ratio. It is simpler to modify the delay values as they are additive and do not affect other features, such as QoS settings. Remember to adjust the EIGRP variance setting to make sure the secondary path falls under the range of allowed metrics.

Now for the calculations part. Suppose the path going via Ethernet interface has the metric value of M1 and the metric across the Frame-Relay connection is M2. We need to make the following hold true:

4*M1 = M2

Or in other words (we omit the scaling factor of 256)

4* [10^7/BW1 + Delay1] = [10^7/BW2 + Delay2]

Let's find the bandwidth values for any prefix learned from R5. There should be two paths and two bandwidth values:

```
Rack1R4#show ip eigrp topology 192.10.1.0
<snip>
  Routing Descriptor Blocks:
  174.1.45.5 (FastEthernet0/1), from 174.1.45.5, Send flag is 0x0
      Composite metric is (261120/258560), Route is External
      Vector metric:
        Minimum bandwidth is 10000 Kbit
<snip>
  174.1.145.1 (Serial0/0/0), from 174.1.145.1, Send flag is 0x0
      Composite metric is (2684416/2172416), Route is External
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 40100 microseconds
<snip>
```

Based on these BW values, we need to find out what the delay values should be to keep the proportion::

4* [10000000/10000 + Delay1] = [1000000/1544 + Delay2]
4*[1000 + Delay1] = [6477 + Delay2]
4000 + 4*Delay1 = 6477 + Delay2
Delay2 = 4*Delay1 – 2477

Let Delay1 equals to 6000 (arbitrary value) and Delay2 value would be 21523, or rounded 21500.

🕮 Deep Dive

- Can one implement unequal cost load balancing in OSPF?

Task 2.5 Verification

Verify the traffic share counters for any prefix behind R5:

```
Routing entry for 192.10.1.0/24
  Known via "eigrp 1024", distance 170, metric 1794560, type external
  Redistributing via eigrp 1024
  Last update from 174.1.145.1 on Serial0/0/0, 00:01:38 ago
  Routing Descriptor Blocks:
    174.1.145.1, from 174.1.145.1, 00:01:38 ago, via Serial0/0/0
      Route metric is 7164672, traffic share count is 1
     Total delay is 215110 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
  * 174.1.45.5, from 174.1.45.5, 00:01:38 ago, via FastEthernet0/1
      Route metric is 1794560, traffic share count is 4
     Total delay is 60100 microseconds, minimum bandwidth is 10000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
```

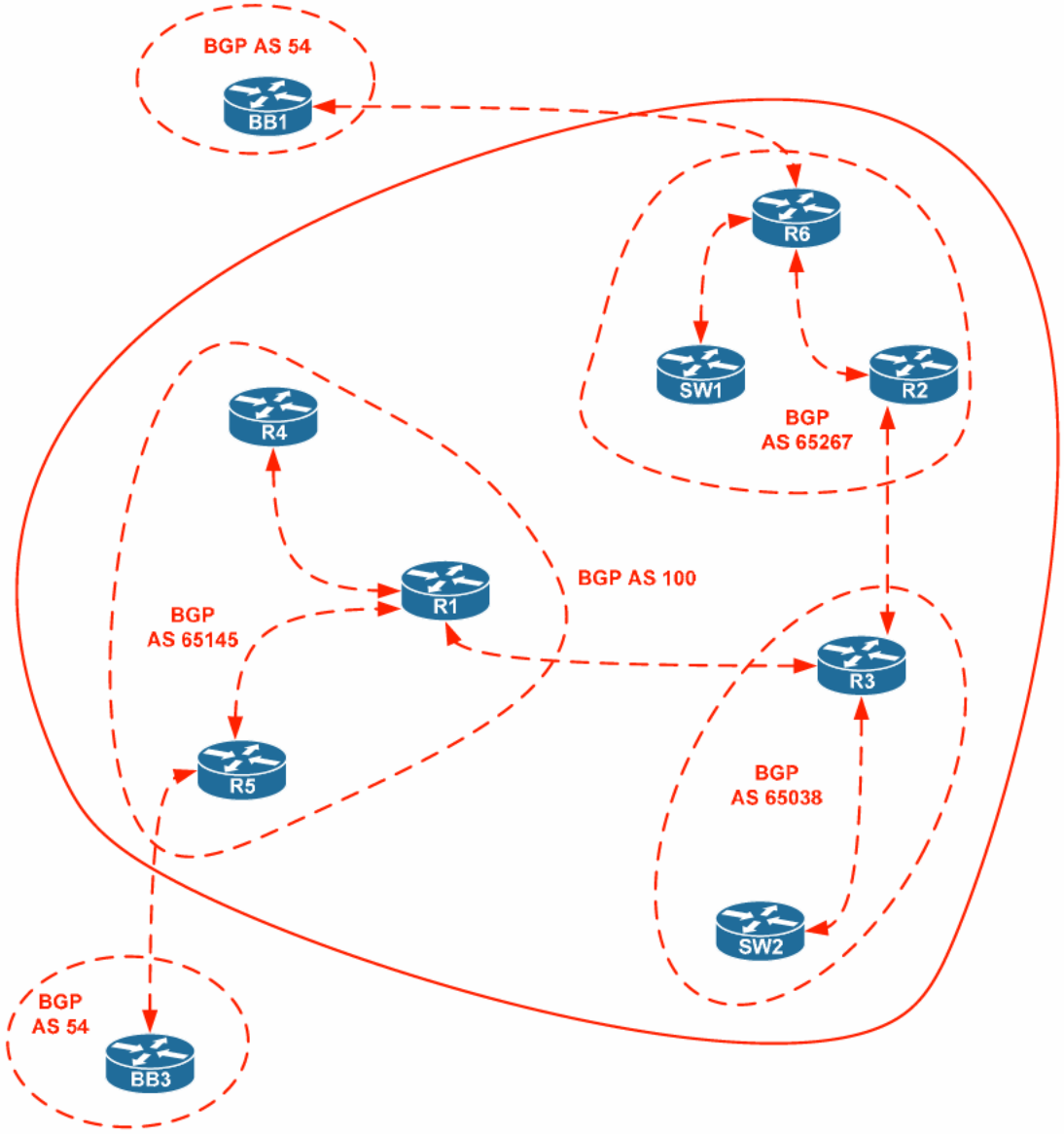

Task 2.6 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

```
BGP > BGP Filtering with Prefix-Lists
BGP > iBGP Confederations
```

The below is the BGP peering diagram made up for this scenario. Notice the confederation AS#100 encompassing other sub-systems.



R5:

```
!
! Filter the aggregate on IBGP peering sessions
!
router bgp 65145
 redistribute static
 neighbor 150.1.4.4 route-map TO_R4 out
 neighbor 150.1.1.1 route-map TO_R1 out
!
ip route 174.1.0.0 255.255.0.0 Null0
!
ip prefix-list AGGREGATE seq 5 permit 174.1.0.0/16
!
! We use separate route maps to allow for flexible policy per peer
!
route-map TO_R4 deny 10
 match ip address prefix-list AGGREGATE
!
route-map TO_R4 permit 1000
!
route-map TO_R1 deny 10
 match ip address prefix-list AGGREGATE
!
route-map TO_R1 permit 1000
```

R6:

```
router bgp 65267
 redistribute static
 neighbor 174.1.26.2 route-map TO_R2 out
 neighbor 174.1.67.7 route-map TO_SW1 out
!
ip route 174.1.0.0 255.255.0.0 Null0
!
ip prefix-list AGGREGATE seq 5 permit 174.1.0.0/16
!
route-map TO_R2 deny 10
 match ip address prefix-list AGGREGATE
!
route-map TO_R2 permit 1000
!
route-map TO_SW1 deny 10
 match ip address prefix-list AGGREGATE
!
route-map TO_SW1 permit 1000
```


Task 2.6 Breakdown

There is a number of ways to aggregate prefixes in BGP. The most preferred and stable one is using static routes and network statement – it allows explicit control over advertised information. Another way is using static routes and redistribution, where redistribution is normally controlled by tag-based route-map. This approach allows for flexibly adding new routes without changing BGP configuration. In our case, we prefer simple uncontrolled redistribution, which not normally recommended in real world scenarios.

We are asked to block all iBGP peers from receiving the summary prefix. While filtering could be achieved in multiple ways, using per-peer route-map allows for maximum flexibility and easy policy changes. Therefore we prefer this way and match the aggregate prefix with a prefix-list. Notice that most of the configurations is similar between R5 and R6 so you may simply adjust R5's configuration and route-map names and then apply to R6.

🕮 Deep Dive

- What is easier to migrate to from iBGP full mesh: Route Reflectors or Confederations?

Task 2.6 Verification

Verify the prefixes advertised to the backbone routers:

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

Total number of prefixes 1

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

Total number of prefixes 1
```

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

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

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


Task 2.7 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Next-Hop Processing – Next-Hop Self
BGP > BGP Next-Hop Processing – Manual Modification

R5:

```
!
! peer-address changes the next-hop to peering address
!
route-map TO_R4 permit 1000
 set ip next-hop peer-address
!
route-map TO_R1 permit 1000
 set ip next-hop peer-address
```

R6:

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

☠ Pitfall

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

Task 2.7 Solution

Typically, advertising a customer link into IGP is not recommended, as it will propagate instabilities into the routing protocol. A more common approach is to change the next-hop address to the iBGP peering address of the router connecting to the customer. This, however, is not an optimal choice in "managed" environments, where ISP manages the customer CE devices. In such solutions, advertising the peering link subnet into BGP is normally the best solution, as it provides management access to the CE device yet helps isolating IGP from the link instabilities.

In our case, changing the next-hop to the speaker's iBGP peering address is requested in the scenario. However, it is explicitly prohibited to use the command **`neighbor X.X.X.X next-hop-self`**. Therefore, our only option is using a route-map and the command **`set ip next-hop peer-address`**. How is this approach better? Firstly, it allows for changing the next hop selectively, which may be important in some traffic engineering scenarios. Secondly, using the peer-address option allows the policy to be reusable among different peers, as it does not hard-code the next-hop IP address, like it would be using the command **`set ip next-hop <IP>`**. Notice that peer-address here means the "peering address" not the "peer's address".

🕮 Deep Dive

- When changing a next-hop for VPNv4 update, what effect that will have?

Task 2.7 Verification

Verify that the next-hop is changed:

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

Check the route reachability from internal peers:

```
Rack1R2#show ip bgp 119.0.0.0
BGP routing table entry for 119.0.0.0/8, version 17
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
     2
  54
    174.1.26.6 from 174.1.26.6 (150.1.6.6)
      Origin IGP, metric 0, localpref 100, valid, confed-internal, best

Rack1R4#show ip bgp 119.0.0.0
BGP routing table entry for 119.0.0.0/8, version 16
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Flag: 0x900
  Not advertised to any peer
  (65038 65267) 54
    174.1.26.6 (metric 7164416) from 150.1.1.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, confed-internal
  54
    150.1.5.5 (metric 1689600) from 150.1.5.5 (150.1.5.5)
      Origin IGP, metric 0, localpref 100, valid, confed-internal, best
```


Task 2.8 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Bestpath Selection – MED
BGP > iBGP Confederations

R3:
```
router bgp 65038
 network 174.1.3.0 mask 255.255.255.0
```

R4:
```
router bgp 65145
 network 174.1.4.0 mask 255.255.255.0
```

R5:
```
router bgp 65145
 neighbor 204.12.1.254 route-map MED out
 neighbor 204.12.1.254 send-community
!
ip prefix VLAN_4 permit 174.1.4.0/24
ip prefix VLANS_3_&_7 permit 174.1.3.0/24
ip prefix VLANS_3_&_7 permit 174.1.7.0/24
!
! bump metric up for VLAN4 when advertising to BB3
!
route-map MED permit 10
 match ip address prefix-list VLAN_4
 set metric 20
 set community no-export
!
route-map MED permit 20
 match ip address prefix-list VLANS_3_&_7
 set metric 10
 set community no-export
!
route-map MED permit 30
```


R6:

```
router bgp 65267
 neighbor 54.1.2.254 route-map MED out
 neighbor 54.1.2.254 send-community
!
ip prefix VLAN_4 permit 174.1.4.0/24
ip prefix VLANS_3_&_7 permit 174.1.3.0/24
ip prefix VLANS_3_&_7 permit 174.1.7.0/24
!
! Bump metric up for VLANs 3/7 when advertising to BB1
!
route-map MED permit 10
 match ip address prefix-list VLAN_4
 set metric 10
 set community no-export
!
route-map MED permit 20
 match ip address prefix-list VLANS_3_&_7
 set metric 20
 set community no-export
!
route-map MED permit 30
```

SW1:

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


Task 2.8 Breakdown

In order to affect inbound traffic flow, either AS-Path or MED should be modified. MED is only compared (by default) between prefixes learned from the same autonomous system. Since AS 100 is dual homed to AS 54, this is an appropriate case to modify the multi-exit discriminator value. Keep in mind that by default routes with no metric are preferred over routes having metric attribute. In our solution we increase VLAN4's prefix metric when advertising it to BB3 and increase VLANs 3,7 metric when advertising these prefixes to BB1, thus enforcing the desired inbound routing pattern.

The task additionally requires the detailed routing information to be contained within AS 54. This is achieved in the usual way, by using default community signaling and attaching the no-export community to all advertised prefixes.

Notice that in real world, using MED to affect inbound traffic may not work, as some ISPs clean MED inbound on their peering links to enforce "hot potato" routing based on their internal IGP metrics and prevent possible BGP oscillations caused by MED attribute processing in RR/Confederation scenarios.

🕮 Deep Dive

- What condition should be met in order to **`bgp always-compare-med`** option to have constructive use?

Task 2.8 Verification

Verify that communities are enabled on peering links:

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

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

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

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

  100
    54.1.2.6 from 54.1.2.6 (150.1.6.6)
      Origin IGP, metric 20, localpref 100, valid, external
      Community: no-export

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

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

Task 2.9 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Communities – Local AS

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

Task 2.9 Breakdown

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

🕮 Deep Dive

- What are the most common operations implemented with BGP community signaling?

Task 2.9 Verification

Verify that the summary prefix on R5:

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

Check that R3 does not receive the summary prefix:

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


Task 3.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IPv6 > OSPFv3
IPv6 > OSPFv3 over NBMA

R1:
```
interface FastEthernet0/0.1001
 ipv6 ospf 1 area 0
!
interface Serial0/0
 ipv6 ospf neighbor FE80::5
 ipv6 ospf 1 area 0
```

R5:
```
interface Serial0/0/0
 ipv6 ospf priority 0
 ipv6 ospf 1 area 0
```

Task 3.1 Verification

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

```
Rack1R1#show ipv6 ospf neighbor

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

Check OSPF routes at R5:

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

Task 3.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IPv6 > IPv6 OSPFv3

```
R5:
ipv6 route ::/0 Null0
!
ipv6 router ospf 1
 default-information originate
```

Task 3.2 Breakdown

The task implicitly talks about default routing, but prohibits the use of unconditional default route in OSPFv3. Thus, the only solution is creating a static default IPv6 route and advertising conditional default route into OSPFv3.

Task 3.2 Verification

Verify the default route in the routing table of R1:

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


Task 3.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IPv6 > IPv6 Redistribution

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

Task 3.3 Breakdown

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

Task 3.3 Verification

Check for the route to VLAN1001 on R4:

```
Rack1R4#show ipv6 route rip
IPv6 Routing Table - Default - 6 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   FEC0:CC1E:1:1::/64 [120/2]
     via FE80::213:1AFF:FE68:B04C, FastEthernet0/1
```

Validate connectivity:

```
Rack1R4#ping ipv6 FEC0:CC1E:1:1::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FEC0:CC1E:1:1::1, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/64 ms
```

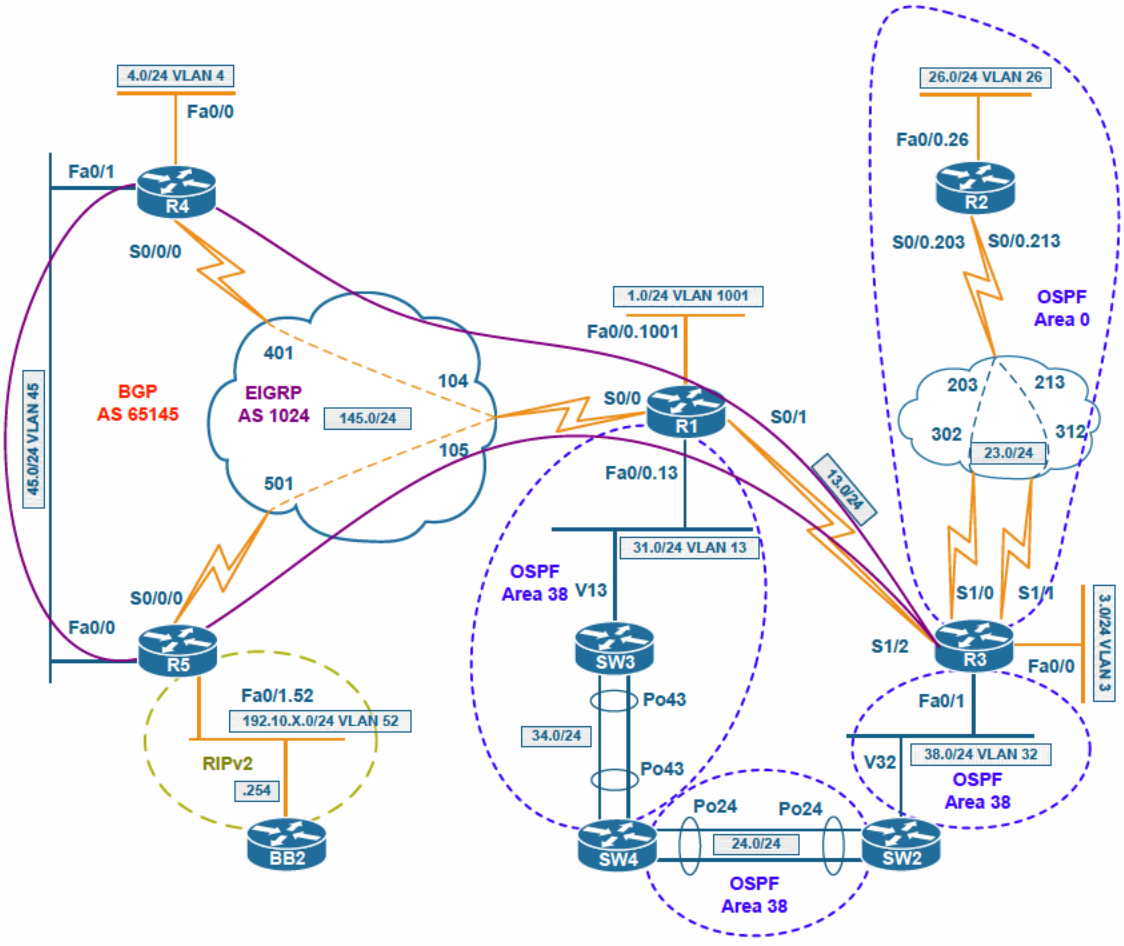

Task 5.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

```
Multicast > PIM Sparse-Dense Mode
Multicast > Auto-RP
Multicast > Auto-RP – Multiple Candidate RPs
Multicast > Auto-RP – Filtering Candidate RPs
Multicast > RPF Failure
```

Multicast topology spans multiple IGP protocols, so it is worth making a Multicast diagram for this scenario. Multicast enabled links are highlighted below:



R1:

```
interface Loopback0
 ip pim sparse-dense-mode
!
ip pim send-rp-announce Loopback0 scope 16 group-list EVEN interval 5
!
! Select groups with even first octet
!
ip access-list standard EVEN
 permit 224.0.0.0 0.255.255.255
 permit 226.0.0.0 0.255.255.255
 permit 228.0.0.0 0.255.255.255
 permit 230.0.0.0 0.255.255.255
 permit 232.0.0.0 0.255.255.255
 permit 234.0.0.0 0.255.255.255
 permit 236.0.0.0 0.255.255.255
 permit 238.0.0.0 0.255.255.255
!
! This fixes the RPF failure on R1
!
router eigrp 1024
 distance eigrp 90 109
```

R2:

```
interface Loopback0
 ip pim sparse-dense-mode
!
ip pim send-rp-announce Loopback0 scope 16 group-list ODD interval 5

!
! Select groups with odd first octet
!
ip access-list standard ODD
 permit 225.0.0.0 0.255.255.255
 permit 227.0.0.0 0.255.255.255
 permit 229.0.0.0 0.255.255.255
 permit 231.0.0.0 0.255.255.255
 permit 233.0.0.0 0.255.255.255
 permit 235.0.0.0 0.255.255.255
 permit 237.0.0.0 0.255.255.255
 permit 239.0.0.0 0.255.255.255
```



```
R3:
!
! Fix the prefix mask in OSPF to /24
! This is needed to fix the RPF problem on R1
!
interface Loopback0
 ip pim sparse-dense-mode
 ip ospf network point-to-point
!
ip pim send-rp-discovery Loopback0 scope 16
!
ip pim rp-announce-filter rp-list R2_RP group-list R2_GROUPS
ip pim rp-announce-filter rp-list R1_RP group-list R1_GROUPS
!
ip access-list standard R1_GROUPS
 permit 224.0.0.0 0.255.255.255
 permit 226.0.0.0 0.255.255.255
 permit 228.0.0.0 0.255.255.255
 permit 230.0.0.0 0.255.255.255
 permit 232.0.0.0 0.255.255.255
 permit 234.0.0.0 0.255.255.255
 permit 236.0.0.0 0.255.255.255
 permit 238.0.0.0 0.255.255.255
!
ip access-list standard R1_RP
 permit 150.1.1.1
!
ip access-list standard R2_GROUPS
 permit 225.0.0.0 0.255.255.255
 permit 227.0.0.0 0.255.255.255
 permit 229.0.0.0 0.255.255.255
 permit 231.0.0.0 0.255.255.255
 permit 233.0.0.0 0.255.255.255
 permit 235.0.0.0 0.255.255.255
 permit 237.0.0.0 0.255.255.255
 permit 239.0.0.0 0.255.255.255
!
ip access-list standard R2_RP
 permit 150.1.2.2
```


Task 5.1 Breakdown

After discovering the topology, you will notice that multicast domain spreads across two different IGPs. Very often this means RPF problems, due to inconsistent route selection among the domains. Additionally, if you don't have multicast enabled over all links in IGP topology, you almost inevitably face RPF issues as some sources may be reachable over non-multicast enabled links.

In this task, R1 and R2 send dense mode multicast which is supposed to reach R3. R3 in turn is the MA, advertising the mapping to all routers in the multicast domain. While there are no problems with R1 and R2 multicast reaching R3, the multicast flow from R3 will break at R1, as this router prefers all OSPF paths over OSPF, and not over EIGRP link to R3. There are two main ways to fix this problem: either create a static multicast route, or fix IGP path selection so that R3 prefers reaching OSPF routes over the connection to R3. The second option also means suboptimal routing from EIGRP domain to the prefixes in Area 38, which may lead to routing loops. Therefore, if you opt to the second option, make sure you still maintain full reachability.

In the solution, we decrease EIGRP's AD on R1 to 109, making EIGRP external prefixes preferred over OSPF paths. In addition to this, we have to apply the command ip ospf network point-to-point to R3's Loopback0 interface. In fact, it's always a good idea to use this command on OSPF loopbacks when you have redistribution configured. The issue is that R1 would still prefer reaching R3's Loopback0 prefix via OSPF, as it sees it with the mask /32 while EIGRP would have it with the mask /24.

Would this change preserve full reachability? It should, as now R1 will route for all OSPF Area 38 prefixes to R3, and R3 will never route back, as it has OSPF prefixes preferred over EIGRP's and there are no additional links between R1 and R3. You may still want to run a ping script to ensure everything is fine.

The explicit requirements in the task are purely technical. You need to configure R1 and R2 to advertise themselves as RPs for only selected ranges of the multicast groups. Since the ranges are non-summarizable, you need to create detailed access-lists matching the selected groups. One list matches the "odd" groups; another one matches the "even" groups. Remember, you may only use permit statements to select groups for sparse mode, as deny statement would mean the group is to be flooded in dense mode. After we're done with group to RP advertisements, we need to configure RP advertisement filters on R3, to ensure consistent mapping. There are no overlapping groups between the RPs and hence no contention, so this filtering is purely for security reasons. You re-create the same access-lists you've been using for the RP announcements and bind them to the RPs, thus fulfilling the requirement.

Task 5.1 Verification

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

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

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


Task 5.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Multicast > PIM Sparse Mode

R1, R2, R3, R4 and R5:

```
!
! Configure all multicast routers as requested
!
ip pim spt-threshold 128
```

Task 5.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Multicast > PIM NBMA Mode

R1:

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

R4:

```
interface Fa0/0
ip igmp join-group 226.0.0.4
```


Task 5.3 Verification

To verify, ping 226.0.0.4 from a host emulated in VLAN52. Configure SW1 with a temporary interface in VLAN52 and enable it sending multicast out of this interface:

```
SW1:
ip multicast-routing distributed
!
interface Vlan 52
 ip address 192.10.1.7 255.255.255.0
 ip pim dense-mode

Rack14SW1#ping 226.0.0.4 repeat 1000

Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 226.0.0.4, timeout is 2 seconds:

Reply to request 0 from 174.1.145.4, 126 ms
Reply to request 1 from 174.1.145.4, 118 ms
Reply to request 1 from 174.1.145.4, 135 ms
Reply to request 2 from 174.1.145.4, 134 ms
```


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

```
Rack14R1#show ip mroute 226.0.0.4
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 226.0.0.4), 00:35:24/stopped, RP 150.1.1.1, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0, 174.1.145.4, Forward/Sparse-Dense, 00:34:24/00:02:33

(192.10.1.7, 226.0.0.4), 00:00:31/00:03:09, flags: T
  Incoming interface: Serial0/0, RPF nbr 174.1.145.5
  Outgoing interface list:
    Serial0/0, 174.1.145.4, Forward/Sparse-Dense, 00:00:31/00:02:58
```


Task 5.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Multicast > Multicast Helper Map

R1:

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

R2:

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


Task 5.4 Breakdown

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

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

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

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

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

The syntax for broadcast to multicast conversion is:

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

The syntax for multicast to broadcast conversion is:

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

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

Note

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

Finally, in order to transmit the multicast to broadcast conversion, the target interface must support directed broadcast transmissions. To enable the transmission of these directed broadcast packets issue the **`ip directed-broadcast`** interface level command. Notice that by default the translated packet will have a destination IP address of 255.255.255.255. If you want to change it to a subnet broadcast, apply the command **`ip broadcast-address`** to the egress interface.

Deep Dive

- What options could you use to propagate Layer 2 broadcast among different segments?

Task 5.4 Verification

Option 1:

Create an IP SLA monitor on R6:

R6:
```
ip sla 2
 udp-echo 174.1.26.255 3434 source-port 3434 control disable
 timeout 1000
 threshold 1000
 frequency 1
ip sla schedule 2 life forever start-time now
```

Note

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

Option 2:

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

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

R6:
```
ip domain-lookup

Rack1R6#ping r1

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


Verification for both options:

Debug multicast packets on R2:

```
Rack1R2#debug ip mpacket

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

Verify the directed broadcast packets on R1:

```
Rack1R1#debug ip packet detail 100

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


Task 6.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Traffic Filtering with Extended Access-Lists
IP Services > Proxy ARP
System Management > CDP

R5:

```
no ip source-route
!
interface FastEthernet0/1.52
 ip access-group APPEASE_MANAGER in
 no ip proxy-arp
 no cdp enable
!
interface FastEthernet0/1.53
 ip access-group APPEASE_MANAGER in
 no ip proxy-arp
 no cdp enable
!
ip access-list extended APPEASE_MANAGER
 deny   tcp any 174.1.0.0 0.0.255.255 eq www
 deny   tcp any 174.1.0.0 0.0.255.255 eq telnet
 !
 deny   tcp any 150.1.0.0 0.0.255.255 eq www
 deny   tcp any 150.1.0.0 0.0.255.255 eq telnet
 deny   icmp any any echo
 permit ip any any
```


Task 6.1 Breakdown

The task requests you to perform the well-known “hardening” procedure for the router. This normally means disabling all unneeded or potentially risky services, which may expose a security vulnerability or leak information to a potential attacker. Examples of such services are source-routing option, CDP on public interfaces, proxy ARP (allows for hosts on different subnets to use the router), ICMP unreachables and so on. In addition to hardening the router, there is an explicit request for filtering some types of traffic. Based on the service names and destinations it is easy to come up with a packet filtering access-list.

Notice that the access-list follows the logic “block some services, permit everything else”, which is not a very secure approach, but causes less potential service disruption when implemented in the first place.

🕮 Deep Dive

- Why IP source routing could be dangerous?
- What benefits of source routing can you point out?

Task 6.1 Verification

Verify that CDP is disabled on required interfaces:

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

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

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

Verify that source-routing is disabled:

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


Finally, verify the access list:

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

Task 6.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC Classification and Marking

```
R5:
 class-map match-all FROM_ETHERNET
  match input-interface FastEthernet0/1
!
 policy-map SAME_INTERFACE
  class FROM_ETHERNET
   drop
!
interface FastEthernet0/1.52
 service-policy output SAME_INTERFACE
!
interface FastEthernet0/1.53
 service-policy output SAME_INTERFACE
```


Task 6.2 Breakdown

The above configuration stops traffic from transiting between two interfaces. By matching the input interface for traffic and dropping it, traffic cannot come in VLAN 52 interface and go out of VLAN 53, or vice versa. Notice that you cannot match input subinterface with the MQC, but the logic here still works as it drops packet coming in and then out of the same interface.

Similar filtering could be implemented using route-maps, which allow matching output interface or subinterface and therefore are more granular.

🕮 Deep Dive

- Is there any other ways to disallow two subinterfaces from communicating to each other?

Task 6.2 Verification

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

```
R5:
router rip
 redistribute connected
```

Next do a ping from BB2 to 204.12.1.254:

```
BB2>ping 204.12.1.254

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

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

```
Rack14R5#show policy-map interface fastEthernet 0/1.53
 FastEthernet0/1.53

  Service-policy output: SAME_INTERFACE

    Class-map: FROM_ETHERNET (match-all)
      4 packets, 472 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: input-interface FastEthernet0/1
      drop

    Class-map: class-default (match-any)
      1 packets, 78 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

R5:
router rip
 no redistribute connected
```

Task 6.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC Classification and Marking

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

Task 6.3 Breakdown

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

Task 6.3 Verification

Verify the access-lists:

```
Rack14R5#show ip access-lists
Extended IP access list SMTP_SERVER
    10 permit tcp host 192.10.1.100 eq smtp 204.12.1.0 0.0.0.255
    20 permit tcp 204.12.1.0 0.0.0.255 host 192.10.1.100 eq smtp
```

Check the updated class-map:

```
Rack14R5#show class-map
 Class Map match-any class-default (id 0)
   Match any

 Class Map match-all FROM_ETHERNET (id 1)
   Match input-interface FastEthernet0/1
   Match not access-group name SMTP_SERVER
```


Task 7.1 Solutions

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IP Services > Proxy ARP

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

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

Task 7.1 Breakdown

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

🕮 Deep Dive

- Can you break a large IP network into segments using Proxy ARP?

Task 7.1 Verification

Verify that proxy ARP is disabled:

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

Task 7.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IP Services > WCCPv1 Web Cache

```
R4:
!
! Enable WCCP cache services
!
ip wccp web-cache

!
! Enable redirection outound
!
interface Serial0/0/0
 ip wccp web-cache redirect out
```

Task 7.2 Breakdown

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

Task 7.2 Verification

Check WCCP status on the interface:

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

Task 7.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IP Services > IP SLA

R6:

```
ip sla 1
 icmp-echo 115.0.0.1
 request-data-size 1250
 timeout 25

 threshold 20
 frequency 30
!
ip sla schedule 1 start-time now life forever
```

Task 7.3 Breakdown

Make sure you are able to properly interpret the scenario requirements as follows:

- Since R6 needs to account for ICMP packets that have a delay higher than 25ms, this translates to the **`timeout`** value above 25.
- Send frequency and packet sizes are easy to deduce from the task requirements.
- In order to validate the SLA enforcing 20ms latency the **`threshold`** value needs to be 20.

Each IP SLA has what is called an operation return-code value. There are multiple possible codes for it, but most common are "**`OK`**" and "**`Over-Threshold`**". When using IP SLA along with object tracking, two properties of IP SLA operation could be tracked: state and reachability. The table below summarizes shows how the tracking methods and the codes returned result in tracking states::

Tracking	Return Code	Track State
State	OK (all other return codes)	Up Down
Reachability	OK or OverThreshold (all other return codes)	Up Down

We are performing state tracking in this scenario and want the object to go down once the SLA threshold is crossed. Therefore, setting the operation threshold to 20ms will ensure that the tracking state is down to signal the breach of SLA.

Task 7.3 Verification

Check IP SLA monitor configuration

```
Rack1R6#show ip sla configuration 1
IP SLAs, Infrastructure Engine-II.
Entry number: 1
Owner:
Tag:
Type of operation to perform: icmp-echo
Target address/Source address: 115.0.0.1/0.0.0.0
Type Of Service parameter: 0x0
Request size (ARR data portion): 1250
Operation timeout (milliseconds): 25
Verify data: No
Vrf Name:
Schedule:
   Operation frequency (seconds): 30  (not considered if randomly
scheduled)
   Next Scheduled Start Time: Start Time already passed
   Group Scheduled : FALSE
   Randomly Scheduled : FALSE
   Life (seconds): Forever
   Entry Ageout (seconds): never
   Recurring (Starting Everyday): FALSE
   Status of entry (SNMP RowStatus): Active
Threshold (milliseconds): 20 (not considered if react RTT is
configured)
Distribution Statistics:
   Number of statistic hours kept: 2
   Number of statistic distribution buckets kept: 1
   Statistic distribution interval (milliseconds): 20
History Statistics:
   Number of history Lives kept: 0
   Number of history Buckets kept: 15
   History Filter Type: None
Enhanced History:
```


Verify the SLA statistics:

```
Rack1R6#sho ip sla statistics 1
IPSLAs Latest Operation Statistics

IPSLA operation id: 1
Type of operation: icmp-echo
        Latest RTT: NoConnection/Busy/Timeout
Latest operation start time: *21:49:54.957 UTC Tue May 4 2010
Latest operation return code: Timeout
Number of successes: 0
Number of failures: 9
Operation time to live: Forever
```

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

Task 7.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IP Services > HSRP

R2:
```
interface FastEthernet0/0.26
 standby 1 ip 174.1.26.254
 standby 1 preempt
 standby version 2
```

R6:
```
track 1 ip sla 1
!
interface FastEthernet0/1.26
 standby 1 ip 174.1.26.254
 standby 1 priority 110
 standby 1 preempt
 standby 1 track 1 decrement 20
 standby version 2
```

Task 7.4 Breakdown

For gateway redundancy, there are three protocols that can be used: HSRP, VRRP and GLBP. Since there should be only one active forwarder/gateway at any point in time, GLBP is not an option. Since VRRP runs directly over IP, using protocol number 112, HSRP would fit, since it runs over UDP. However, by default HSRP uses the “all routers” multicast address of 224.0.0.2. There is exists HSRP version 2, which provides HSRP with some enhancements. Among them, the multicast address used is changed to 224.0.0.102 to resolve the conflict with CGMP Leave processing.

🕮 Deep Dive

- What rule do you need to observer when deploying HSRP in a switch-block where VLANs spread multiple switches?

Task 7.4 Verification

Verify the tracking configuration:

```
Rack1R6#show track
Track 1
  IP SLA 1 state
  State is Down
    1 change, last change 00:10:00
  Latest operation return code: Timeout
  Tracked by:
    HSRP FastEthernet0/1.26 1
```

Verify the HSRP groups:

```
Rack1R6#show standby
FastEthernet0/1.26 - Group 1 (version 2)
  State is Standby
    4 state changes, last state change 00:04:10
  Virtual IP address is 174.1.26.254
  Active virtual MAC address is 0000.0c9f.f001
    Local virtual MAC address is 0000.0c9f.f001 (v2 default)
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 0.192 secs
  Preemption enabled
  Active router is 174.1.26.2, priority 100 (expires in 10.176 sec)
    MAC address is 0011.20fd.6e00
  Standby router is local
  Priority 90 (configured 110)
    Track object 1 state Down decrement 20
  Group name is "hsrp-Fa0/1.26-1" (default)
```

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

Task 8.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC Class-Based Generic Traffic Shaping

R1:

```
policy-map SHAPE
class class-default
  shape average 128000
!
interface Serial0/0
 service-policy output SHAPE
```

R4, and R5:

```
policy-map SHAPE
class class-default
  shape average 128000
!
interface Serial0/0/0
 service-policy output SHAPE
```


Task 8.1 Breakdown

Frame-Relay SPs typically mark packets that exceed traffic contract with DE bit, if the traffic contract supports transmissions over peak rate. Even contracts with no peak rate may allow customers send over CIR, if the SP configures ingress policer accordingly. In this scenario we are explicitly required to configure shaping to a specified CIR value. Since Bc is not provided, we let the system pick default value, as it does not matter.

The requirement states that **`map-class frame-relay`** command should not be used. This means we are left with two options: using the legacy **`traffic-shape`** command or MQC generic traffic shaping. We resort to the last option in the solution.

🕮 Further Reading

The four flavors of Frame-Relay Traffic Shaping

http://blog.ine.com/2010/06/14/the-four-flavors-of-frame-relay-traffic-shaping/

Task 8.1 Verification

Check the policy map applied to the interface:

```
Rack1R1#show policy-map interface serial 0/0

 Serial0/0

  Service-policy output: SHAPE

    Class-map: class-default (match-any)
      1 packets, 13 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
        Target/Average   Byte   Sustain   Excess    Interval  Increment
          Rate           Limit  bits/int  bits/int  (ms)      (bytes)
        128000/128000    1984   7936      7936      62        992

     Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
     Active Depth                         Delayed   Delayed   Active
     -      0         0         0         0         0         no
```

Task 8.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC Class-Based GTS and CBWFQ

R1:

```
policy-map CBWFQ
 class class-default
  bandwidth percent 100
  queue-limit 10
!
policy-map SHAPE
 class class-default
  service-policy CBWFQ
```

Task 8.2 Breakdown

In order to tune the shaper's queue, you need to apply a nested service-policy. This nested configuration specifies the CBWFQ settings. Since there is only one class and one queue, we manually allocate all bandwidth to the class-default and tune the queue limit. Notice that by default, with HQF (Hierarchical Queueing Framework) and IOS 12.4(20)T the default shaper's queue is FIFO, not WFQ as it was in prior versions.

Task 8.2 Verification

Verify new queueing parameters:

```
Rack1R1#show policy-map interface serial 0/0

 Serial0/0

  Service-policy output: SHAPE

    Class-map: class-default (match-any)
      27 packets, 1063 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval
Increment
             Rate           Limit  bits/int  bits/int  (ms)
(bytes)
           128000/128000    1984   7936      7936      62        992

        Adapt  Queue     Packets   Bytes     Packets   Bytes
Shaping
        Active Depth                         Delayed   Delayed   Active
        -      0         18        946       0         0         no

      Service-policy : CBWFQ

        Class-map: class-default (match-any)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: any
          Queueing
            Output Queue: Conversation 25
            Bandwidth 100 (%)
            Bandwidth 128 (kbps)Max Threshold 10 (packets)
             (pkts matched/bytes matched) 0/0
         (depth/total drops/no-buffer drops) 0/0/0
```


Task 8.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC Classification and Marking
QoS > LLQ and Remaining Bandwidth Reservation

R1:

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

R3:

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


Task 8.3 Breakdown

The task wording clearly spells out the following: the classification criteria for traffic (UDP port 7070), type of traffic (latency sensitive) and finally the maximum allowed bit rate and bursts. All these parameters translate into the QoS policy implementing LLQ policer for the mentioned traffic. Classification is performed based on an access-list matching the protocol and port number.

Task 8.3 Verification

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

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

  Service-policy output: LLQ

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

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


Task 8.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC WRED

R4:
```
policy-map CBWFQ
  class class-default
   bandwidth percent 100
   random-detect
   random-detect precedence 5 60 90 5
!
policy-map SHAPE
 class class-default
  service-policy CBWFQ
```

Task 8.4 Breakdown

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

🕮 Deep Dive

- Does WRED have any effect on non-TCP traffic?

Task 8.4 Verification

Verify the WRED configuration:

```
Rack1R4#show policy-map interface serial 0/0/0

 Serial0/0/0

  Service-policy output: SHAPE

    Class-map: class-default (match-any)
      8507 packets, 407217 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 7/280
      shape (average) cir 128000, bc 512, be 512
      target shape rate 128000
        lower bound cir 0,  adapt to fecn 0

      Service-policy : CBWFQ

        Class-map: class-default (match-any)
          7 packets, 627 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: any
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 7/280
          bandwidth 100% (128 kbps)
            Exp-weight-constant: 9 (1/512)
            Mean queue depth: 0 packets
class    Transmitted    Random drop  Tail drop  Minimum    Maximum    Mark
         pkts/bytes     pkts/bytes   pkts/bytes  thresh     thresh     prob

0              3/39           0/0          0/0       20         40     1/10
1               0/0           0/0          0/0       22         40     1/10
2               0/0           0/0          0/0       24         40     1/10
3               0/0           0/0          0/0       26         40     1/10
4               0/0           0/0          0/0       28         40     1/10
5               0/0           0/0          0/0       60         90     1/5
6               4/241         0/0          0/0       2          40     1/10
7               0/0           0/0          0/0       34         40     1/10
```

VOL1 Scenario Reference

Bridging and Switching > MST Root Bridge Selection
Bridging and Switching > MSTP and Rapid Spanning Tree
Bridging and Switching > 802.1q Tunneling
Bridging and Switching > 802.1q Native VLAN
Bridging and Switching > MST Load Balancing with Port Priority
Frame-Relay > PPP over Frame-Relay
QoS > MLP Link Fragmentation and Interleaving
QoS > MLPPP LFI over Frame-Relay
OSPF > OSPF Cleartext Authentication
QoS > OSPF Internal Summarization
RIPv2 > RIPv2 Authentication
EIGRP > EIGRP Split Horizon
OSPF > OSPF Stub Areas
EIGRP > Unequal Cost Load Balancing
BGP > BGP Filtering with Prefix-Lists
BGP > iBGP Confederations
BGP > BGP Next-Hop Processing – Next-Hop Self
BGP > BGP Next-Hop Processing – Manual Modification
BGP > BGP Bestpath Selection – MED
BGP > iBGP Confederations
BGP > BGP Communities – Local AS
IPv6 > OSPFv3
IPv6 > OSPFv3 over NBMA
IPv6 > IPv6 OSPFv3
IPv6 > IPv6 Redistribution
Multicast > PIM Sparse-Dense Mode
Multicast > Auto-RP
Multicast > Auto-RP – Multiple Candidate RPs
Multicast > Auto-RP – Filtering Candidate RPs
Multicast > RPF Failure
Multicast > PIM Sparse Mode
Multicast > PIM NBMA Mode
Multicast > Multicast Helper Map

Security > Traffic Filtering with Extended Access-Lists
IP Services > Proxy ARP
System Management > CDP
QoS > MQC Classification and Marking
QoS > MQC Classification and Marking
IP Services > Proxy ARP
IP Services > WCCPv1 Web Cache
IP Services > IP SLA
IP Services > HSRP
QoS > MQC Class-Based Generic Traffic Shaping
QoS > MQC Class-Based GTS and CBWFQ
QoS > MQC Classification and Marking
QoS > LLQ and Remaining Bandwidth Reservation
QoS > MQC WRED

Deep Dive Answers

- Can you load-balance VLANs on the links connecting different MST regions?
    - It is not possible as there is only a single CIST instance connecting regions. All VLANs map to the same instance, and hence you cannot load-balance them across different link connecting two regions. If you want load-balancing, put the links in the same regions.
- How would authentication differ if OSPF was authenticating LSA contents, not packets?
    - OSPF implements adjacency authentication, i.e. it validates that it knows the other end of the adjacency. This implicitly authenticates the LSAs received on the "validated" link. Using LSA authentication alone removes the need to authenticate every link, by assuming there is a common authentication key to all routers in the area, used to sign the LSAs. This, however, make authentication less granular and more vulnerable, as there is a single failure element in the security mechanism now.
- In absence of MPLS, what other major drawback of summarization could you point out, not mentioning suboptimal routing?
    - As it's well known, MPLS does not work well with summarization, due to end-to-end LSP breaking. Although there are workaround to this problem, there are other issues. Suboptimal routing is one of them and the last one is loss of reachability information. Routing and Reachability are two different concepts: a node could be routable but not reachable. Aggregation hides reachability information as it suppresses information for individual nodes. Node reachability is important concept in many scenarios, such as BGP next-hop-tracking, as it greatly improves convergence. There are certain solutions to this problem, like IP SLA tracking or propagation reachability hash in a special LSA. But overall it is important to remember the three main problems that summarization create: inconsistent forwarding information, suboptimal routing and loss of reachability detection.
- Do key indexes have to match for RIPv2 authentication to succeed?
    - Yes, RIPv2 sends MD5 signature along with the key ID used to sign the payload. This allows the receiving side to select proper key for signature validation. This also allows having multiple keys to validate incoming updates, although only one key (with the lowest ID) is used to send the updates out of the interface.

- What is third-party next-hop?
    - Normally, distance-vector protocols use the IP address used to source the update as the next hop for the routes received in the update. The third-party next-hop explicitly encodes the next-hop IP address for routes found in the update. This allows the receiving side to pick up a better next-hop than the router sending the update. This is particularly useful in partial mesh scenarios, where hub is relaying updates between spokes, but a better direct path exists between the spokes. The other use is optimized redistribution, where redistributed protocol has better exit point on the same subnet which is translated into the third-party next-hop. It is important to remember that in distance-vector protocols the third-party next-hop always belongs to the same subnet as the routers sending and receiving the updates.
- Does RIPv2 support third-party next-hop?
    - Yes, RIPv2 supports this feature, though you cannot change it manually. Third-party next-hop could be set automatically in some redistribution scenarios to provide better routing.
- Can one implement unequal cost load balancing in OSPF?
    - It is not possible by default, as SPF does not create secondary loop-free paths. With some extension to SPF, it is possible to create loop-free alternatives (LFAs) where possible and theoretically use them for unequal cost load-balancing. However, normally the LFAs are one-hop and used for the purpose of fast-rerouting and not load-balancing. There is an alternative way of implementing unequal-cost load balancing in OSPF by means of splitting faster links into groups of slower links and letting OSPF utilize them for ECMP. However, this approach requires manual intervention and is cumbersome.
- What is easier to migrate to from iBGP full mesh: Route Reflectors or Confederations?
    - Route-Reflectors support non-disruptive migration and thus were favored by many ISPs. Use of confederations makes sense in the situations where you want to scale your IGP, as every confederation may have a separate, isolated IGP. Additionally, confederations make merging two different AS# much easier, as they require no renumbering.

- When changing a next-hop for VPNv4 update, what effect that will have?
    - VPNv4 update has forwarding information encoded via a VPN label and IPv4 next-hop address. The VPN label could only be interpreted in the context of the receiving side identified by the next-hop IPv4 address. Therefore, it is important that the IPV4 next-hop does not change, or the VPN forwarding information becomes useless. However, sometimes it is needed to change the forwarding path for the MPLS LSP to make it go through a VPNv4 relaying node, such as an ASBR in inter-AS VPN scenarios. The solution it to re-generate the VPNv4 label every time the IPv4 next-hop changes and create a local mapping from the new to the old VPN4 label. This allows for changing forwarding path and preserving VPNv4 information.
- What condition should be met in order to **`bgp always-compare-med`** option to have constructive use?
    - MED is supposed to be the IGP metric for the protocol used in the advertising system. Therefore, to make reasonable comparisons, the metrics used in the two different AS's should have the same meaning, e.g. come from the same type of IGP and follow the same assignment logic. This is rarely being true, but the disallowing MED comparison between AS's opens possibility for BGP route oscillations due to inconsistent route selection in RR/Confederation environments.
- What are the most common operations implemented with BGP community signaling?
    - The first common use is marking the type of route: e.g. customer, peer, transit. This allows for more flexible policy configuration in BGP environment. The second common use is letting communities signal best-path selection preference. E.g. certain communities could me "apply local preference of 1000 to tagged routes" and allow the downstream AS to affect inbound traffic flow using local signaling. The last "esoteric" use implies using communities to signal QoS or security treatment of specific prefixes: e.g. provide different classes of services in the upstream AS or blackhole certain prefixes prior to traffic reaching downstream AS.
- What options could you use to propagate Layer 2 broadcast among different segments?
    - Normally, this is a task for Layer 2 VPNs, either point-to-point or multipoint. Some legacy solutions could include the use of bridging over GRE tunnels (undocumented feature).or DLSw which allows connecting bridged segments. If IP only connectivity is required, it is possible to use Proxy ARP to glue the large IP subnet, provided that all routes are advertising just parts of it.

- Why IP source routing could be dangerous?
  - Firstly, most routers have to leverage slow forwarding path to process IP options in the packet header, which poses serious issue to CPU overutilization. Secondly, crafted packets could be used to bypass certain devices such as firewalls.
- What benefits of source routing can you point out?
  - Theoretically, source routing allows the "customer" selecting the path it prefers, thus giving end-user better control over the services. Additionally, source route might be a better choice in dynamic environment, such as mobile networks, as opposed to proactive routing distribution the network topology/routing information.
- Is there any other ways to disallow two subinterfaces from communicating to each other?
  - At layer 2, you may use private VLANs or protected ports that split the connected devices into separate broadcast sub-domains. At layer 3, you may separate interfaces into different VRFs and block route export between the two VRFs, while still letting every VRF access the global routing table.
- Can you break a large IP network into segments using Proxy ARP?
  - It is possible in some migration or high-availability scenarios. The idea is to configured the hosts with a shorter netmask, while routers with longer mask and proxy ARP. The hosts would try to ARP for their peers, while routers will intercept the messages and then route the packets to remote segments. The only problem could be the use of link-local signaling, such as TTL=1 packets, e.g. link local multicast.
- What rule do you need to observe when deploying HSRP in a switch-block where VLANs spread multiple switches?
  - In situations like this, multiple switches provide separate physical instances for the same gateway. To avoid suboptimal forwarding, you need to make sure HSRP primary gateway is the same switch as STP root for the same VLAN. Otherwise, traffic going to the HSRP primary gateway will have to cross the root bridge, which introduces additional forwarding hop, decreases performance and makes troubleshooting more complicated.
- Does WRED have any effect on non-TCP traffic?
  - No, WRED assumes traffic elasticity, and unless the protocol is designed to interpret packet drops as signal of congestion it will not works well with WRED/RED.

Lab 9 Solutions

Task 1.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Bridging and Switching > STP Root Guard
Bridging and Switching > STP Root Bridge Election
Bridging and Switching > 802.1q Tunneling
Bridging and Switching > Layer 3 Etherchannel
Bridging and Switching > Etherchannel over 802.1q Tunneling
Bridging and Switching > Tuning STP Convergence Timers

SW1:

```
system mtu 1504
!
interface range FastEthernet0/17, FastEthernet 0/20
 switchport access vlan 100
 switchport mode dot1q-tunnel
 l2protocol-tunnel cdp
 no cdp enable
!
interface range FastEthernet0/18, FastEthernet 0/21
 switchport access vlan 101
 switchport mode dot1q-tunnel
 l2protocol-tunnel cdp
 no cdp enable
```


SW2:

```
interface FastEthernet0/24
 spanning-tree guard root
!
spanning-tree vlan 68 root primary diameter 3 hello-time 1
```

SW3:

```
interface Port-channel1
 no switchport
 ip address 148.1.1.9 255.255.255.0
!
interface FastEthernet0/14
 no switchport
 no ip address
 channel-group 1 mode on
!
interface FastEthernet0/15
 no switchport
 no ip address
 channel-group 1 mode on
```

SW4:

```
interface Port-channel1
 no switchport
 ip address 148.1.1.10 255.255.255.0
!
interface FastEthernet0/14
 no switchport
 no ip address
 channel-group 1 mode on
!
interface FastEthernet0/15
 no switchport
 no ip address
 channel-group 1 mode on
```


Task 1.1 Breakdown

Spanning-tree root guard is typically used when a provider is leasing an FastEthernet line out to a customer. In the case that a switch in the customer's network is elected root, all traffic from the provider and its other customers must follow sub-optimal forwarding. Root guard can be used to prevent this case by disabling the port connected to the customer if a superior BPDU is received. The term *superior* BPDU implies that the cost to the root out that port is better than the current root port. Another use case is Layer 2 switchblock, where root guard is used on the distribution switches to protect against the acces-level devices erroneously or maliciously claiming themselves as root devices. To enable root guard, use the interface level command **`spanning-tree guard root`**.

In order to bundle links in Etherchannel over a 802.1q cloud make sure to use a separate tunneling VLAN for each of the two links in the Etherchannel. The VLANs only need to be present on SW1, since SW1 has the connections to both SW3 and SW4.

🕮 Deep Dive

- Why do you need a separate VLAN for every Etherchannel link tunneled using QinQ?
- Is it possible to disable MAC address learning on the VLANs used to tunnel P2P traffic?

Task 1.1 Verification

Verify that Root Guard is enabled; check the output of "**show span vlan 68**" on SW2:

```
Rack1SW2#show spanning-tree interface fa0/24 detail
 Port 26 (FastEthernet0/24) of VLAN0232 is forwarding
   Port path cost 100, Port priority 128, Port Identifier 128.26.
   Designated root has priority 33000, address 0015.63c8.8800
   Designated bridge has priority 33000, address 0016.9d31.8380
   Designated port id is 128.26, designated path cost 9
   Timers: message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is shared by default
   Root guard is enabled on the port
   BPDU: sent 2346, received 0

Rack1SW2#show span vlan 68

VLAN0068
  Spanning tree enabled protocol ieee
  Root ID     Priority    24644
              Address     001b.8f0c.2a00
              This bridge is the root
              Hello Time   1 sec  Max Age  7 sec  Forward Delay  5 sec

<output omitted>

Rack1SW3#show etherchannel summary | begin Group
Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
1      Po1(RU)          -        Fa0/14(P)   Fa0/15(P)
13     Po13(SU)         -        Fa0/16(P)   Fa0/17(P)   Fa0/18(P)
```

Verify the output of "**show cdp neighbors**" on SW3:

```
Rack1SW3#show cdp neighbors | i 0/14|0/15
Rack1SW4      Fas 0/15        141          R S I     WS-C3560- Fas 0/15
Rack1SW4      Fas 0/14        141          R S I     WS-C3560- Fas 0/14
```


Verify that only CDP tunneling is enabled; bote that once L2 tunneling is enabled on a port, BPDU Filter is automatically enabled at the port level; it does not show up at "**show run**" :

```
Rack1SW1#show l2protocol-tunnel summary
COS for Encapsulated Packets: 5
Drop Threshold for Encapsulated Packets: 0

Port        Protocol    Shutdown         Drop             Status
                        Threshold        Threshold
                        (cdp/stp/vtp)    (cdp/stp/vtp)
                        (pagp/lacp/udld) (pagp/lacp/udld)
---------- ---------- ---------------- ---------------- ----------
Fa0/17      cdp --- --- ----/----/----   ----/----/----   up
            ---- ---- ---- ----/----/----   ----/----/----
Fa0/18      cdp --- --- ----/----/----   ----/----/----   up
            ---- ---- ---- ----/----/----   ----/----/----
Fa0/20      cdp --- --- ----/----/----   ----/----/----   up
            ---- ---- ---- ----/----/----   ----/----/----
Fa0/21      cdp --- --- ----/----/----   ----/----/----   up
            ---- ---- ---- ----/----/----   ----/----/----

Rack1SW1#show spanning-tree interface fastEthernet 0/17 detail
 Port 19 (FastEthernet0/17) of VLAN0100 is forwarding
   Port path cost 19, Port priority 128, Port Identifier 128.19.
   Designated root has priority 32868, address 001f.6ce6.8700
   Designated bridge has priority 32868, address 001f.6ce6.8700
   Designated port id is 128.19, designated path cost 0
   Timers: message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   Link type is point-to-point by default
   Bpdu filter is enabled internally
   BPDU: sent 0, received 0

Rack1SW3#ping 148.1.1.10

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 148.1.1.10, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```


Task 1.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Frame-Relay > Inverse-ARP
Frame-Relay > Point-to-Point Subinterfaces
IP Routing > Backup Interface

R3:
```
interface Serial1/1
 ip address 148.1.35.3 255.255.255.0
 no frame-relay inverse-arp ip 311
 no frame-relay inverse-arp ip 312
 no frame-relay inverse-arp ip 314
```

R4:
```
interface Serial0/0/0.401 point-to-point
 backup delay 0 300
 backup interface Serial0/1/0
```

R5:
```
interface Serial0/0/0
 ip address 148.1.35.5 255.255.255.0
 no frame-relay inverse-arp ip 501
 no frame-relay inverse-arp ip 502
 no frame-relay inverse-arp ip 503
 no frame-relay inverse-arp ip 504
```


Task 1.2 Breakdown

There are two requirements in this task: configure Frame-Relay inverse ARP on physical interfaces and enable backup link feature in R4. The first requirement is made complicated by requesting that inverse-ARP should be running on some DLCIs, without the use of any subinterfaces. Since it is possible to enumerate all DLCIs using the **`show frame-relay pvc`** command, the solution explicitly disables Inverse-ARP all DLCIs but the ones that should use it. Notice that this does not account for the PVCs added in the future, but the scenario doe not require accounting for this.

The backup interface feature enables tracking of the point-to-point Frame-Relay subinterface on R4, which would go done once the respective DLCI is no longer learned via LMI. The main subinterface going down will trigger the use of the direct serial link connecting R4 and R5. The “return” delay is set to 30 seconds.

🕮 Deep Dive

- What other options you can use to implement backup routing as opposed to using backup interfaces?

Task 1.2 Verification

Check for active dynamic mappings

```
Rack1R3#show frame-relay map
Serial1/0.302 (up): point-to-point dlci, dlci 302(0x12E,0x48E0),
broadcast
          status defined, active
Serial1/1 (up): ip 148.1.35.5 dlci 315(0x13B,0x4CB0), dynamic,
              broadcast,
              CISCO, status defined, active

Rack1R5#show frame-relay map
Serial0/0/0 (up): ip 148.1.35.3 dlci 513(0x201,0x8010), dynamic,
              broadcast,, status defined, active

Rack1R5#ping 148.1.35.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 148.1.35.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms
```

Verify backup link configuration:

```
Rack1R4#show backup
Primary Interface   Secondary Interface   Status
-----------------   -------------------   ------
Serial0/0/0.401       Serial0/1/0         normal operation
```

Test the backup configuration:

```
Rack1R4(config)#interface s0/0/0.401
Rack1R4(config-subif)#do debug backup
Backup events debugging is on
Rack1R4(config-subif)#no frame-relay interface-dlci 401
BACKUP(Serial0/0/0.401): event = primary interface went down
BACKUP(Serial0/0/0.401): changed state to "waiting to backup"
BACKUP(Serial0/0/0.401): event = timer expired on primary
BACKUP(Serial0/0/0.401): secondary interface (Serial0/1) made active
BACKUP(Serial0/0/0.401): changed state to "backup mode"
%LINK-3-UPDOWN: Interface Serial0/1, changed state to up
BACKUP(Serial0/1): event = secondary interface came up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1, changed
state to up
BACKUP(Serial0/1): event = secondary interface came up
```



```
Rack1R4(config-subif)#do show backup
Primary Interface   Secondary Interface   Status
-----------------   -------------------   ------
Serial0/0/0.401       Serial0/1/0           backup mode

Rack1R4(config-subif)# frame-relay interface-dlci 401
BACKUP(Serial0/0/0.401): event = primary interface came up
BACKUP(Serial0/0/0.401): changed state to "waiting to revert"
Rack1R4(config-fr-dlci)#exit

Rack1R4(config-subif)#do show backup
Primary Interface   Secondary Interface   Status
-----------------   -------------------   ------
Serial0/0/0.401       Serial0/1/0           waiting to revert (290 more
seconds)
```


Task 2.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

OSPF > OSPF MD5 Authentication
OSPF > Miscellaneous OSPF Features

R1 and SW2:

```
router ospf 1
 ignore lsa mospf
```

R1:

```
service password-encryption
!
interface FastEthernet0/0
 ip ospf message-digest-key 7 md5 CISCO
!
router ospf 1
 area 168 authentication message-digest
```

SW2:

```
service password-encryption
!
interface Vlan18
 ip ospf message-digest-key 7 md5 CISCO
!
interface Vlan68
 ip ospf authentication null
!
router ospf 1
 area 168 authentication message-digest
```


Task 2.1 Breakdown

Cisco's OSPF implementation does not support type 6 LSA (multicast OSPF). By default, every time one of these LSAs is received, a syslog message is generated. To disable this behavior, issue the OSPF routing process subcommand **ignore lsa mospf**.

There are two ways to enable OSPF authentication, on a per area basis and on a per interface basis. As the above task states, that the **ip ospf authentication message-digest** command cannot be used, area authentication must be used. However, this task also states that the adjacency between R6 and SW2 must not be authenticated. Since R1, R6, and SW2 are all in the same area, this presents a problem. This task illustrates that there are actually three types of OSPF authentication, MD5, clear text, and NULL. By setting the OSPF authentication type to NULL on VLAN 68, SW2 has effectively disabled OSPF authentication on that interface.

Next, this task states that R1 and SW2 should use a pre-encrypted key with the number 7. This task is designed to illustrate the difference between key number and encryption type. The key number of an MD5 key is used as a seed or salt value in the MD5 hash algorithm. This seed is a number used to randomize the output of the hash algorithm, and decrease the effectiveness of a brute force attack on the MD5 algorithm. Key numbers must match on all devices authenticating on the segment.

The encryption type determines whether or not the password is stored in a clear-text or encrypted form in the router's configuration file. By issuing the **service password-encryption** global configuration command, all clear text passwords in the routers configuration are encrypted with type 7 encryption. Type 7 encryption uses a Cisco proprietary insecure reversible encryption algorithm, based on a Vigenere cipher. This encryption is simply used to shield a password from an over the shoulder user seeing the password in show commands or backups or configuration files

Task 2.1 Verification

Confirm that authentication has been enabled

```
Rack1SW2#show ip ospf interface vl18 | begin Message
  Message digest authentication enabled
    Youngest key id is 7

Rack1SW2#show running-config interface vl68 | begin Message
Rack1SW2#
```

Verify password encryption:

```
Rack1R1#show running-config interface fa0/0

interface FastEthernet0/0
 ip address 148.1.18.1 255.255.255.0
 ip ospf message-digest-key 7 md5 7 106D202A2638
 ip ospf priority 0
 duplex auto
 speed auto

Rack1SW2#show running-config interface vl18

interface Vlan18
 ip address 148.1.18.8 255.255.255.0
 ip ospf message-digest-key 7 md5 7 00273A352774
```


Task 2.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

EIGRP > EIGRP Convergence Timers
EIGRP > EIGRP MD5 Authentication
EIGRP > EIGRP Key-Chain Rotation

R3:
```
interface Serial1/1
 ip hello-interval eigrp 100 4
 ip hold-time eigrp 100 12
```

R5:
```
interface Serial0/0/0
 ip hello-interval eigrp 100 4
 ip hold-time eigrp 100 12
```

R5 and SW1:
```
key chain EIGRP
 key 1
   key-string CISCO2010
    accept-lifetime 00:00:00 Jan 1 1993 00:15:00 Jan 1 2011
    send-lifetime 00:00:00 Jan 1 1993 23:45:00 Dec 31 2010
 key 2
   key-string CISCO2011
    accept-lifetime 23:15:00 Dec 31 2010 infinite
    send-lifetime 23:45:00 Dec 31 2010 infinite
```

R5:
```
interface FastEthernet0/0
 ip authentication mode eigrp 100 md5
 ip authentication key-chain eigrp 100 EIGRP
```

SW1:
```
interface FastEthernet0/5
 ip authentication mode eigrp 100 md5
 ip authentication key-chain eigrp 100 EIGRP
```


Task 2.2 Breakdown

To adjust neighbor hello and dead intervals in EIGRP, use the interface level commands **ip hello-interval eigrp [AS] [hello_interval]** and **ip hold-time eigrp [AS] [hold_time]**. By default, the EIGRP hello interval is 60 seconds for low speed NBMA interfaces and 5 seconds for all other media. The hold-time defaults to three times these values.

EIGRP Key chain authentication allows for key lifetime and rotation based on time. This option allows for smooth transition between authentication keys throughout the entire network at the same time. The two options that dictate a key's timing are the **accept-lifetime** and the **send-lifetime**. As their names imply, the accept lifetime is the time period for which the specified key will be accepted from a neighbor as valid for authentication. The send-lifetime specifies during which time interval the key will be valid for transmission to a neighbor. The **infinite** option dictates that the specified key is valid from the start time on.

In order to perform smooth key rollover, the way EIGRP authentication works is:

- All sent packets are authenticated using the first available/active (based on time) key
- All received packets are authenticated against any available/active keys; if more than one key is active packets are matched against all keys.

This is why regularly, in EIGRP key transition phases, the accept-lifetime of the actual and the next key overlap.

> **Note**
>
> To ensure smooth key transition is a real network, NTP should be used in any practical time based key chain authentication implementations.

Task 2.2 Verification

Verify the EIGRP interface characteristics:

```
Rack1R5#show ip eigrp interfaces detail s0/0/0
IP-EIGRP interfaces for process 100

                Xmit Queue   Mean   Pacing Time   Multicast    Pending
Interface Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer   Routes
Se0/0/0    1        0/0        39       0/15         159           0
  Hello interval is 4 sec
  Next xmit serial <none>
  Un/reliable mcasts: 0/0  Un/reliable ucasts: 4/7
  Mcast exceptions: 0  CR packets: 0  ACKs suppressed: 1
  Retransmissions sent: 1  Out-of-sequence rcvd: 0
  Authentication mode is not set

Rack1R3#show ip eigrp interfaces detail s1/1
IP-EIGRP interfaces for process 100

                 Xmit Queue   Mean   Pacing Time   Multicast    Pending
Interface Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer   Routes
Se1/1        1        0/0       663       5/190       3454           0
  Hello interval is 4 sec
  Next xmit serial <none>
  Un/reliable mcasts: 0/0  Un/reliable ucasts: 4/18
  Mcast exceptions: 0  CR packets: 0  ACKs suppressed: 1
  Retransmissions sent: 11  Out-of-sequence rcvd: 0
  Authentication mode is not set
```

Verify EIGRP authentication:

```
Rack1SW1#show ip eigrp interfaces detail fa0/5
IP-EIGRP interfaces for process 100

                Xmit Queue   Mean   Pacing Time   Multicast    Pending
Interface Peers Un/Reliable  SRTT   Un/Reliable   Flow Timer   Routes
Fa0/5       1        0/0         4       0/10          50           0
  Next xmit serial <none>
  Un/reliable mcasts: 0/2  Un/reliable ucasts: 5/5
  Mcast exceptions: 0  CR packets: 0  ACKs suppressed: 0
  Retransmissions sent: 3  Out-of-sequence rcvd: 0
  Authentication mode is md5,  key-chain is "EIGRP"
```



```
Rack1SW1#show key chain EIGRP
Key-chain EIGRP:
    key 1 -- text "CISCO2005"
        accept lifetime (00:00:00 UTC Jan 1 1993) - (00:15:00 UTC Jan 1
2006) [valid now]
        send lifetime (00:00:00 UTC Jan 1 1993) - (23:45:00 UTC Dec 31
2005) [valid now]
    key 2 -- text "CISCO2006"
        accept lifetime (23:15:00 UTC Dec 31 2005) - (infinite)
        send lifetime (23:45:00 UTC Dec 31 2005) - (infinite)

Rack1SW1#show ip eigrp neighbors fastethernet0/5
IP-EIGRP neighbors for process 100
H   Address         Interface       Hold Uptime   SRTT   RTO  Q  Seq Type
                                   (sec)         (ms)       Cnt Num
0   148.1.57.5      Fa0/5             13 00:01:38    4   200  0  14
```

Task 2.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > VLAN Filters for IP Traffic

```
SW2:
vlan access-map RIP_FILTER 10
 action drop
 match ip address 100
!
vlan access-map RIP_FILTER 20
 action forward
vlan filter RIP_FILTER vlan-list 232
!
access-list 100 permit udp host 192.10.1.254 eq rip any eq rip
```


Task 2.3 Breakdown

Although SW2 is only switching at layer two for the above network segment, it can still filter traffic above layer 3 by using VLAN Access-Lists (VACLs). The above VACL matches all RIP traffic coming from BB2 and discards it. Make sure to add the explicit permit sequence so that all other traffic is forwarded unmodified. Alternatively, an access list could also be applied inbound on the port connecting to BB2.

Pitfall

Notice we used the RIP port value for both source and destination ports. This is because RIP updated need to be filtered, which are always sent with a source/destination pair of 520. Queries/requests may be sent with a source port different than 520, but this is not our case.

Task 2.3 Verification

Before filter has been applied:

```
Rack1R2(config)#access-list 100 permit udp any any eq 520
Rack1R2#debug interface fastEthernet 0/0
Condition 1 set
Rack1R2#debug ip packet detail 100
IP packet debugging is on (detailed) for access list 100
IP: s=192.10.1.254 (FastEthernet0/0), d=224.0.0.9, len 132, rcvd 2
    UDP src=520, dst=520
IP: s=192.10.1.2 (local), d=224.0.0.9 (FastEthernet0/0), len 172,
sending broad/multicast
     UDP src=520, dst=520
IP: s=192.10.1.254 (FastEthernet0/0), d=224.0.0.9, len 132, rcvd 2
```

After filter has been applied:

```
Rack1R2#debug ip packet detail 100
IP packet debugging is on (detailed) for access list 100

IP: s=192.10.1.2 (local), d=224.0.0.9 (FastEthernet0/0), len 132,
sending broad/multicast
    UDP src=520, dst=520
IP: s=192.10.1.3 (FastEthernet0/0), d=224.0.0.9, len 112, rcvd 2
    UDP src=520, dst=520
IP: s=192.10.1.2 (local), d=224.0.0.9 (FastEthernet0/0), len 132,
sending broad/multicast
     UDP src=520, dst=520
```


Task 2.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

RIPv2 > RIPv2 Filtering with Passive Interfaces
RIPv2 > RIPv2 Filtering with Administrative Distance

R3:
```
router rip
 no passive-interface FastEthernet0/0
```

SW1:
```
router rip
 passive-interface default
 no passive-interface Vlan7
 no passive-interface Vlan73
 network 148.1.0.0
```

SW3 and SW4:
```
router rip
 version 2
 network 148.1.0.0
 network 150.1.0.0
 distance 80
 distance 200 0.0.0.0 255.255.255.255 1
 no auto-summary
!
access-list 1 permit 150.1.0.0 0.0.255.255
```

☞ Quick Note
Any value lower than EIGRP's administrative distance could be used here

☞ Quick Note
Ensure EIGRP is used for any of the 150.1.0.0/16 subnets. Altering EIGRP administrative distance for these routes would also be a valid solution.

Task 2.4 Breakdown

The task requires migrating IGP protocol in SW3 and SW4. The general strategy looks as following – deploying the new protocol with a higher AD on the devices and then lowering the distance for the new protocol, letting it take the primary role.

The fact that R1 and SW1 are border devices connected to the RIPv2 domain allows for keeping the same distance for RIP on these two routers. We lower the AD for RIPv2 on SW3 and SW4, but keep the RIP distance selectively high for all Loopback prefixes. This fulfills the requirement for SW3 and SW4 reaching all 150.X.0.0/16 subnets via EIGRP, as this protocol is still running on both devices.

Task 2.4 Verification

Make sure to verify this section after completing redistribution. The networks for the loopbacks of R2, R3 and R4, for example, are only known via RIP initially, since those devices are not running EIGRP.

```
Rack1SW3#show ip route rip
R    204.12.1.0/24 [80/2] via 148.1.1.10, 00:00:07, Port-channel1
R    192.10.1.0/24 [80/1] via 148.1.3.3, 00:00:05, FastEthernet0/13
     148.1.0.0/24 is subnetted, 10 subnets
R       148.1.18.0 [80/3] via 148.1.3.3, 00:00:05, FastEthernet0/13
R       148.1.4.0 [80/4] via 148.1.3.3, 00:00:05, FastEthernet0/13
R       148.1.7.0 [80/1] via 148.1.1.10, 00:00:07, Port-channel1
R       148.1.0.0 [80/1] via 148.1.3.3, 00:00:05, FastEthernet0/13
R       148.1.57.0 [80/2] via 148.1.1.10, 00:00:07, Port-channel1
R       148.1.35.0 [80/1] via 148.1.3.3, 00:00:06, FastEthernet0/13
R       148.1.77.0 [80/2] via 148.1.1.10, 00:00:07, Port-channel1
     31.0.0.0/16 is subnetted, 4 subnets
R       31.3.0.0 [80/3] via 148.1.1.10, 00:00:07, Port-channel1
R       31.2.0.0 [80/3] via 148.1.1.10, 00:00:07, Port-channel1
R       31.1.0.0 [80/3] via 148.1.1.10, 00:00:07, Port-channel1
R       31.0.0.0 [80/3] via 148.1.1.10, 00:00:07, Port-channel1
     150.1.0.0/24 is subnetted, 7 subnets
R       150.1.4.0 [120/4] via 148.1.3.3, 00:02:31, FastEthernet0/13
R       150.1.3.0 [120/1] via 148.1.3.3, 00:02:32, FastEthernet0/13
R       150.1.2.0 [120/2] via 148.1.3.3, 00:02:32, FastEthernet0/13
     30.0.0.0/16 is subnetted, 4 subnets
R       30.2.0.0 [80/3] via 148.1.1.10, 00:00:08, Port-channel1
R       30.3.0.0 [80/3] via 148.1.1.10, 00:00:08, Port-channel1
R       30.0.0.0 [80/3] via 148.1.1.10, 00:00:08, Port-channel1
R       30.1.0.0 [80/3] via 148.1.1.10, 00:00:08, Port-channel1

Rack1SW3#show ip route eigrp
     148.1.0.0/24 is subnetted, 10 subnets
D       148.1.5.0 [90/35840] via 148.1.1.10, 00:28:53, Port-channel1
     150.1.0.0/24 is subnetted, 7 subnets
D       150.1.7.0 [90/158720] via 148.1.1.10, 00:28:53, Port-channel1
D       150.1.5.0 [90/161280] via 148.1.1.10, 00:28:53, Port-channel1
D       150.1.10.0 [90/143360] via 148.1.1.10, 05:50:49, Port-channel1
```



```
Rack1SW4#show ip route rip
R    204.12.1.0/24 [80/1] via 148.1.7.7, 00:00:08, FastEthernet0/21
R    192.10.1.0/24 [80/2] via 148.1.1.9, 00:00:02, Port-channel1
     148.1.0.0/24 is subnetted, 10 subnets
R       148.1.18.0 [80/4] via 148.1.1.9, 00:00:02, Port-channel1
R       148.1.4.0 [80/5] via 148.1.1.9, 00:00:02, Port-channel1
R       148.1.0.0 [80/2] via 148.1.1.9, 00:00:02, Port-channel1
R       148.1.3.0 [80/1] via 148.1.1.9, 00:00:02, Port-channel1
R       148.1.57.0 [80/1] via 148.1.7.7, 00:00:08, FastEthernet0/21
R       148.1.35.0 [80/2] via 148.1.1.9, 00:00:03, Port-channel1
R       148.1.77.0 [80/1] via 148.1.7.7, 00:00:09, FastEthernet0/21
     31.0.0.0/16 is subnetted, 4 subnets
R       31.3.0.0 [80/2] via 148.1.7.7, 00:00:09, FastEthernet0/21
R       31.2.0.0 [80/2] via 148.1.7.7, 00:00:09, FastEthernet0/21
R       31.1.0.0 [80/2] via 148.1.7.7, 00:00:09, FastEthernet0/21
R       31.0.0.0 [80/2] via 148.1.7.7, 00:00:09, FastEthernet0/21
     150.1.0.0/24 is subnetted, 7 subnets
R       150.1.4.0 [200/5] via 148.1.1.9, 00:00:03, Port-channel1
R       150.1.3.0 [200/2] via 148.1.1.9, 00:00:03, Port-channel1
R       150.1.2.0 [200/3] via 148.1.1.9, 00:00:03, Port-channel1
     30.0.0.0/16 is subnetted, 4 subnets
R       30.2.0.0 [80/2] via 148.1.7.7, 00:00:09, FastEthernet0/21
R       30.3.0.0 [80/2] via 148.1.7.7, 00:00:09, FastEthernet0/21
R       30.0.0.0 [80/2] via 148.1.7.7, 00:00:09, FastEthernet0/21
R       30.1.0.0 [80/2] via 148.1.7.7, 00:00:09, FastEthernet0/21

Rack1SW4#show ip route eigrp
     148.1.0.0/24 is subnetted, 10 subnets
D       148.1.5.0 [90/33280] via 148.1.7.7, 00:30:31, FastEthernet0/21
     150.1.0.0/24 is subnetted, 7 subnets
D       150.1.7.0 [90/156160] via 148.1.7.7, 00:30:31, FastEthernet0/21
D       150.1.5.0 [90/158720] via 148.1.7.7, 00:30:31, FastEthernet0/21
D       150.1.9.0 [90/143360] via 148.1.1.9, 00:30:29, Port-channel1

Rack1R3#show ip route | include 150.1.9|150.1.10|148.1.1.0
D       148.1.1.0 [90/30720] via 148.1.3.9, 00:31:00, FastEthernet0/0
D       150.1.10.0 [90/158720] via 148.1.3.9, 00:31:00, FastEthernet0/0
D       150.1.9.0 [90/156160] via 148.1.3.9, 00:31:00, FastEthernet0/0

Rack1SW1#show ip route | include 150.1.9|150.1.10|148.1.1.0
D       148.1.1.0 [90/15616] via 148.1.7.10, 01:34:27, Vlan7
D       150.1.10.0/24 [90/130816] via 148.1.7.10, 01:34:27, Vlan7
D       150.1.9.0/24 [90/143616] via 148.1.7.10, 00:29:10, Vlan7
```


Task 2.5 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

EIGRP > EIGRP Summarization
RIPv2 > RIPv2 Filtering with Administrative Distance

R1:
```
router ospf 1
 redistribute rip subnets
!
router rip
 redistribute ospf 1 metric 1
```

R3:
```
router eigrp 100
 redistribute rip metric 10000 1000 255 1 1500
!
router rip
 redistribute eigrp 100 metric 1
```

R5:
```
interface Serial0/1/0
 ip summary-address eigrp 100 0.0.0.0 0.0.0.0
!
router eigrp 100
 redistribute static metric 64 10 255 1 1500
!
ip route 150.1.4.0 255.255.255.0 148.1.45.4 5
```

SW1:
```
router eigrp 100
 redistribute rip metric 10000 1000 255 1 1500
!
router rip
!
! By using metric 15, BB3 cannot propagate redistributed prefixes
! any further as their metric reaches infinity.
!
 redistribute eigrp 100 metric 15
 distance 171 0.0.0.0 255.255.255.255 RIP_AD
!
ip access-list standard RIP_AD
 permit 148.1.68.0 0.0.0.255
 permit 148.1.8.0 0.0.0.255
 permit 148.1.0.0 0.0.0.255
 permit 148.1.6.0 0.0.0.255
 permit 148.1.4.0 0.0.0.255
 permit 148.1.18.0 0.0.0.255
 permit 192.10.1.0 0.0.0.255
```


Task 2.5 Breakdown

The **default-information [in | out]** statement in EIGRP does not generate a default route advertisement like other IGPs. Instead, it simply allows a default route which already exists to be received or propagated. EIGRP, like IGRP, uses the **ip default-network** statement to propagate default information. A default network must be a classful network dynamically learned that is not directly connected. However, a default-network propagates in all directions. In the above scenario, it is specified that R5 should generate a default route specifically to R4. Since a default-network advertisement cannot be filtered out without filtering the actual network, **ip default-network** cannot be used in this case.

A default route is the most generic IPv4 summary address there is, having a subnet mask of zero. Therefore, a default route can be generated by using an interface summary-address of 0.0.0.0. The above configuration example dictates so.

We are told that we need to make sure that R4's Serial and loopback interfaces still have reachability when R4's frame connection is down, and that we can add a static route. R5 is already advertising the Serial link into EIGRP when that interface is up, so the network to be concerned with is R4's loopback, which is advertised into RIP when the Frame connection is up. R5 will need to have connectivity for this network. Here, that is achieved by adding a static route to the loopback, with R4's serial address as the next hop. The static is then redistributed into EIGRP. Alternatively, we could have R4 add the loopback network to EIGRP, since the adjacency between R4 and R5 is only up when the serial link between those two is not in the backup state.

RIP defines an infinite (unreachable) metric as 16. Metric is incremented as a route advertisement exits an interface. The above task states that the devices on VLAN 73 should not be able to pass on RIP updates learned from SW1. By redistributing prefixes from the EIGRP domain into the RIP domain with a metric of 15, their metric will be infinite when BB3 or any other device tries to pass them on.

Task 2.5 Verification

First make sure that SW1 advertises prefixes with a metric of 15:

```
Rack1SW1#debug ip rip
RIP protocol debugging is on

 RIP: sending v1 update to 255.255.255.255 via Vlan73 (204.12.1.7)
 RIP: build update entries
       network 148.1.0.0 metric 15
       network 150.1.0.0 metric 15
       network 192.10.1.0 metric 15
 RIP: sending v2 update to 224.0.0.9 via Vlan73 (204.12.1.7)
 RIP: build update entries
       148.1.0.0/24 via 0.0.0.0, metric 15, tag 0
       148.1.3.0/24 via 0.0.0.0, metric 15, tag 0
       148.1.4.0/24 via 0.0.0.0, metric 15, tag 0
<output omitted>
```

Next, confirm full connectivity between internal routers. There are two cases here: first when R4 primary link is up and second when backup link is active. For the second case to settle, you need to wait for old RIP routes to expire on R3. You may want to speed up convergence and to do so just do a “clear ip route *” on every RIP speaking router. Next, use the following TCL script to test connectivity:


```
foreach i {
148.1.18.1
148.1.0.1
150.1.1.1
148.1.0.2
150.1.2.2
192.10.1.2
148.1.3.3
148.1.0.3
150.1.3.3
148.1.35.3
192.10.1.3
148.1.0.4
150.1.4.4
148.1.5.5
150.1.5.5
148.1.57.5
148.1.35.5
148.1.6.6
150.1.6.6
148.1.68.6
148.1.7.7
150.1.7.7
148.1.57.7
204.12.1.7
148.1.77.7
148.1.18.8
148.1.8.8
150.1.8.8
148.1.68.8
148.1.1.9
148.1.1.10
150.1.9.9
150.1.10.10
} {  ping $i  }
```

Note that the Frame Relay link on R6 as well as the backup link between R4 & R5, and VLAN4 is excluded from this connectivity test.

Verify on SW3 and SW4 that there are no more loopbacks known in RT via RIP now:

```
Rack1SW3#show ip route rip | i 150
Rack1SW3#

Rack1SW3#show ip route eigrp
     150.1.0.0/16 is variably subnetted, 10 subnets, 2 masks
D       150.1.7.0/24 [90/158720] via 148.1.1.10, 00:22:40, Port-
channel1
D       150.1.5.0/24 [90/161280] via 148.1.1.10, 00:22:40, Port-
channel1
D EX    150.1.4.0/24 [170/514560] via 148.1.3.3, 00:22:40,
FastEthernet0/13
D EX    150.1.3.0/24 [170/514560] via 148.1.3.3, 00:22:40,
FastEthernet0/13
D EX    150.1.2.0/24 [170/514560] via 148.1.3.3, 00:22:40,
FastEthernet0/13
D EX    150.1.1.0/24 [170/514560] via 148.1.3.3, 00:22:40,
FastEthernet0/13
D EX    150.1.8.8/32 [170/514560] via 148.1.3.3, 00:22:41,
FastEthernet0/13
D EX    150.1.6.6/32 [170/514560] via 148.1.3.3, 00:22:41,
FastEthernet0/13
D       150.1.10.0/24 [90/143360] via 148.1.1.10, 00:22:41, Port-
channel1

Rack1SW4#show ip route rip | i 150.
Rack1SW4#

Rack1SW4#show ip route eigrp
     150.1.0.0/16 is variably subnetted, 10 subnets, 2 masks
D       150.1.7.0/24 [90/156160] via 148.1.7.7, 00:23:11,
FastEthernet0/21
D       150.1.5.0/24 [90/158720] via 148.1.7.7, 00:23:11,
FastEthernet0/21
D EX    150.1.4.0/24 [170/517120] via 148.1.1.9, 00:23:11, Port-
channel1
D EX    150.1.3.0/24 [170/517120] via 148.1.1.9, 00:23:11, Port-
channel1
D EX    150.1.2.0/24 [170/517120] via 148.1.1.9, 00:23:11, Port-
channel1
D EX    150.1.1.0/24 [170/517120] via 148.1.1.9, 00:23:11, Port-
channel1
D EX    150.1.8.8/32 [170/517120] via 148.1.1.9, 00:23:11, Port-
channel1
D EX    150.1.6.6/32 [170/517120] via 148.1.1.9, 00:23:11, Port-
channel1
D       150.1.9.0/24 [90/143360] via 148.1.1.9, 00:23:11, Port-channel1
```


Task 2.6 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Auto-Summary
BGP > BGP Regular Expressions

R6:

```
router bgp 100
 neighbor 54.1.3.254 filter-list 25 in
!
ip as-path access-list 25 permit ^54(_[0-9]+)?$
```

R4:

```
router bgp 400
 redistribute connected route-map CONNECTED->BGP
 auto-summary
!
interface FastEthernet 0/1
 ip address 10.1.4.4 255.255.255.0
 no shutdown
!
route-map CONNECTED->BGP permit 10
 match interface FastEthernet 0/1
```


Task 2.6 Breakdown

A common view to take of the BGP table is the directly connected AS's customer routes. Taking this type of view conserves memory since the router does not have to store the entire BGP table.

Recall that the characters used in regular expressions:

Character	Meaning
^	Start of string
$	End of string
[]	Range of characters
-	Used to specify range (i.e. [0-9])
( )	Logical grouping
.	Any single character
*	Zero or more instances
+	One or more instance
?	Zero or one instance
_ (underscore)	Comma, open or close brace, open or close parentheses, start or end of string, or space

The goal of the regular expression used in the above task is to match routes originated in AS 54, or routes originated in AS 54's customer's networks. Therefore, the possible AS paths to match are either "54" or "54 X", where "X" is any single AS. First, let us match just routes originated in AS 54:

```
ip as-path access-list 25 permit ^54$
```

This means:

Character(s)	Meaning
^	Start of line
54	Exactly AS 54
$	End of line

Now, we need to check for the case "54 X" where X is any single AS:

```
ip as-path access-list 25 permit ^54_[0-9]+$
```

This means:

Character(s)	Meaning
^	Start of line
54	Exactly AS 54
_	Space
[0-9]	Any number 0 through 9
+	One or more instance of 0 to 9
$	End of line

Comparing the above two expressions, it is evident that the difference between them is the sequence "_[0-9]+" Therefore, these expressions can be combined by checking for zero or one instance (true or false) of this sequence. Zero or one instances is the character ? Remember that to match the question mark in the line the escape sequence CTRL-V must be issued first.

Notice that the **auto-summary** keyword in BGP only affects prefixes that were redistributed into BGP. When it is on, routes redistributed into the BGP domain are automatically summarized to the classful boundary. When auto-summary is disabled, subnets will retain their subnet mask information.

🕮 Deep Dive

- What regular expression would you use to accept BGP prefixes from your upstream customers, provided that the customers may use AS_PATH prepending?

Task 2.6 Verification

Check the BGP table contents before you applied the solution:

```
Rack1R6#show ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   54.1.3.254                             0 54 i
*> 28.119.17.0/24   54.1.3.254                             0 54 i
*> 112.0.0.0        54.1.3.254                0            0 54 50 60 i
*> 113.0.0.0        54.1.3.254                0            0 54 50 60 i
*> 114.0.0.0        54.1.3.254                0            0 54 i
*> 115.0.0.0        54.1.3.254                0            0 54 i
*> 116.0.0.0        54.1.3.254                0            0 54 i
*> 117.0.0.0        54.1.3.254                0            0 54 i
*> 118.0.0.0        54.1.3.254                0            0 54 i
*> 119.0.0.0        54.1.3.254                0            0 54 i
*>i205.90.31.0      148.1.0.2                 0    100     0 200 254 ?
*>i220.20.3.0       148.1.0.2                 0    100     0 200 254 ?
*>i222.22.2.0       148.1.0.2                 0    100     0 200 254 ?
```

Apply the regexp:

```
Rack1R6#show ip bgp quote-regexp ^54(_[0-9]+)?$ | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   54.1.3.254                             0 54 i
*> 28.119.17.0/24   54.1.3.254                             0 54 i
*> 114.0.0.0        54.1.3.254                0            0 54 i
*> 115.0.0.0        54.1.3.254                0            0 54 i
*> 116.0.0.0        54.1.3.254                0            0 54 i
*> 117.0.0.0        54.1.3.254                0            0 54 i
*> 118.0.0.0        54.1.3.254                0            0 54 i
*> 119.0.0.0        54.1.3.254                0            0 54 i
```

Modify regexp slightly for verification:

```
Rack1R6#show ip bgp quote-regexp ^200(_[0-9]+)?$ | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*>i205.90.31.0      148.1.0.2                 0    100     0 200 254 ?
*>i220.20.3.0       148.1.0.2                 0    100     0 200 254 ?
*>i222.22.2.0       148.1.0.2                 0    100     0 200 254 ?
```

Verify the prefix origination/summarization:

```
Rack1R4#show ip bgp
BGP table version is 9, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop      Metric LocPrf Weight Path
*> 10.0.0.0         0.0.0.0            0         32768 ?
*> 205.90.31.0      148.1.0.2                        0 100 200 254 ?
*> 220.20.3.0       148.1.0.2                        0 100 200 254 ?
*> 222.22.2.0       148.1.0.2                        0 100 200 254 ?
<output omitted>
```

Task 2.7 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Aggregation
BGP > BGP Aggregation – Suppress Map
BGP > BGP Communities – No-Export

R6:
```
router bgp 100
 network 54.1.3.0 mask 255.255.255.0
 network 148.1.6.0 mask 255.255.255.0
!
 aggregate-address 148.1.0.0 255.255.128.0 suppress-map SUPPRESS_MAP
!
route-map SUPPRESS_MAP permit 10
```

SW1:
```
interface Loopback1
 ip address 148.1.177.7 255.255.255.0
!
router bgp 65057
 neighbor 148.1.57.5 send-community
 network 148.1.177.0 mask 255.255.255.0 route-map SET_COMM
!
route-map SET_COMM permit 10
 set community no-export
```


Task 2.7 Breakdown

The **summary-only** keyword suppresses all subnets of an aggregate in order to prevent them from being advertised. The **suppress-map** is a route-map that can selectively suppress one or more subnets of the aggregate.

Typically, this configuration is used when the amount of prefixes that should be suppressed is outweighed by the amount that should not be suppressed. Unlike an unsuppress-map, the suppress-map is configured on the aggregate itself, and affects the local BGP table. The unsuppress-map is applied on a per neighbor basis.

The above configuration uses an unsuppress-map that does not match anything. This is effectively an explicit permit statement for the route-map, and will suppress all subnets of the aggregate.

Another way to solve this task would have been to simply filter the subnets from being advertised to BB1 by using an access-list or prefix-list.

The above task illustrates the usage of the **route-map** keyword on the **network** statement. This route-map can be used to modify attributes of the prefix as it is originated into the BGP domain. These attributes may include weight, local-preference, MED, and community.

Task 2.7 Verification

Verify the summary prefix generation. Confirm that the specific prefix has been suppressed.

```
Rack1R6#show ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*>i10.0.0.0         148.1.0.4                0    100      0 400 ?
*> 28.119.16.0/24   54.1.3.254                            0 54 i
*> 28.119.17.0/24   54.1.3.254                            0 54 i
*> 54.1.3.0/24      0.0.0.0                  0         32768 i
*> 114.0.0.0        54.1.3.254               0             0 54 i
*> 115.0.0.0        54.1.3.254               0             0 54 i
*> 116.0.0.0        54.1.3.254               0             0 54 i
*> 117.0.0.0        54.1.3.254               0             0 54 i
*> 118.0.0.0        54.1.3.254               0             0 54 i
*> 119.0.0.0        54.1.3.254               0             0 54 i
*> 148.1.0.0/17     0.0.0.0                              32768 i
s> 148.1.6.0/24     0.0.0.0                  0         32768 i
*>i205.90.31.0      148.1.0.2                0    100      0 200 254 ?
*>i220.20.3.0       148.1.0.2                0    100      0 200 254 ?
*>i222.22.2.0       148.1.0.2                0    100      0 200 254 ?

Rack1R6#show ip bgp neigh 54.1.3.254 advertised-routes
BGP table version is 17, 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
*>i10.0.0.0         148.1.0.4                0    100      0 400 ?
*> 54.1.3.0/24      0.0.0.0                  0         32768 i
*> 148.1.0.0/17     0.0.0.0                              32768 i
*>i205.90.31.0      148.1.0.2                0    100      0 200 254 ?
*>i220.20.3.0       148.1.0.2                0    100      0 200 254 ?
*>i222.22.2.0       148.1.0.2                0    100      0 200 254 ?

Total number of prefixes 6
```

Confirm that R5 does not advertise the prefix to any eBGP peer:

```
Rack1R5#show ip bgp 148.1.177.0
BGP routing table entry for 148.1.177.0/24, version 23
Paths: (1 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer, RIB-failure(17))
  Not advertised to any peer
  Local
    148.1.57.7 from 148.1.57.7 (150.1.7.7)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Community: no-export
```


Task 3.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IPv6 > Automatic 6to4 Tunnels
IPv6 > IPv6 Filtering

R3:
```
interface Tunnel3456
 ipv6 address 2002:9601:303:3456::3/64
 tunnel source Loopback0
 tunnel mode ipv6ip 6to4
!
ipv6 route 2002::/16 Tunnel3456
```

R4:
```
interface Tunnel3456
 ipv6 address 2002:9601:404:3456::4/64
 tunnel source Loopback0
 tunnel mode ipv6ip 6to4
!
ipv6 route 2002::/16 Tunnel3456
```

R5:
```
interface Tunnel3456
 ipv6 address 2002:9601:505:3456::5/64
 tunnel source Loopback0
 tunnel mode ipv6ip 6to4
!
ipv6 route 2002::/16 Tunnel3456
```

R6:
```
interface Tunnel3456
 ipv6 address 2002:9601:606:3456::6/64
 tunnel source Loopback0
 tunnel mode ipv6ip 6to4
!
ipv6 route 2002::/16 Tunnel3456
```

R6:
```
!
! VTY access-class filter
!
ipv6 access-list PC_IN_VLAN6
 permit ipv6 host 2002:9601:606:1:0209:6BFF:FE06:47EF any
!
line vty 0 4
 ipv6 access-class PC_IN_VLAN6 in
```


Task 3.1 Breakdown

The task has two requirements: the first one is setting IPv6 tunnels and static routing and the second one is about enabling IPv6 traffic filtering. The task wording points toward the use of 6to4 tunnels sourced off the Loopback0 interfaces. We use the static route for 2002::/16 to reach all 6to4 based subnets across the multipoint tunnels.

The second part of the scenario requires applying an access-class to VTY lines. There is a separate access-class available to IPv6 hosts and it uses named extended access-lists only.

🕮 Deep Dive

- Is dynamic routing possible over 6to4 tunnels?
- Do 6to4 tunnels have to use the fixed prefix 2002::/16?

Task 3.1 Verification

Verify the 6to4 tunneling by looking at the routing table.

```
Rack1R3#show ipv6 route static
IPv6 Routing Table - 9 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra,OI - OSPF inter,OE1 - OSPF ext 1,OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
S   2002::/16 [1/0]
     via ::, Tunnel3456
```

Test connectivity:

```
Rack1R3#ping 2002:9601:404:3456::4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2002:9601:404:3456::4, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 76/76/80 ms

Rack1R3#ping 2002:9601:505:3456::5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2002:9601:505:3456::5, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/70/72 ms

Rack1R3#ping 2002:9601:606:3456::6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2002:9601:606:3456::6, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/23/28 ms

Rack1R3#traceroute 2002:9601:606:3456::6

Type escape sequence to abort.
Tracing the route to 2002:9601:606:3456::6

  1 2002:9601:606:3456::6 20 msec 20 msec 24 msec
```


Telnet to R6 from any other IPv6-enabled host:

```
Rack1R3#telnet 2002:9601:606::6
Trying 2002:9601:606::6 ...
% Connection refused by remote host
```

Task 4.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

MPLS VPN > VRF Lite

R5:

```
ip vrf R5TEST
 rd 5:5

interface loopback50
 ip vrf forw R5TEST
 ip address 50.50.50.5 255.255.255.0

interface Loopback 51
 ip vrf forwarding R5TEST
 ip address 51.51.51.5 255.255.255.0
```

R4:

```
ip vrf R4TEST
 rd 4:4

interface Loopback 40
 ip vrf forwarding R4TEST
 ip address 40.40.40.4 255.255.255.0


interface Loopback 41
 ip vrf forwarding R4TEST
 ip address 41.41.41.4 255.255.255.0
```


Task 4.1 Breakdown

The task simply asks for creating VRF tables in R4 and R5 and assigning some interfaces to the new VRF.

🕮 Deep Dive

- How can you select traffic/routes to be added to a VRF?

Task 4.1 Verification

```
Rack1R4#show ip route vrf R4TEST connected
     40.0.0.0/24 is subnetted, 1 subnets
C       40.40.40.0 is directly connected, Loopback40
     41.0.0.0/24 is subnetted, 1 subnets
C       41.41.41.0 is directly connected, Loopback41
Rack1R4#

Rack1R5#show ip route vrf R5TEST connected
     51.0.0.0/24 is subnetted, 1 subnets
C       51.51.51.0 is directly connected, Loopback51
     50.0.0.0/24 is subnetted, 1 subnets
C       50.50.50.0 is directly connected, Loopback50
Rack1R5#
```


Task 4.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

MPLS VPN > VRF Lite
MPLS VPN > PE-CE Routing with EIGRP

R4:
```
interface tunnel 45
 tunnel source 150.1.4.4
 tunnel destination 150.1.5.5
 ip vrf forwarding R4TEST
 ip address 148.1.45.4 255.255.255.254
```

R5:
```
!
interface tunnel 45
 tunnel source 150.1.5.5
 tunnel destination 150.1.4.4
 ip vrf forwarding R5TEST
 ip address 148.1.45.5 255.255.255.254
```

R4:
```
router eigrp 99
 address-family ipv4 vrf R4TEST
 autonomous-system 45
 network 40.40.40.4 0.0.0.0
 network 41.41.41.4 0.0.0.0
 network 148.1.45.4 0.0.0.0
 no auto-summary
```

R5:
```
router eigrp 99
 address-family ipv4 vrf R5TEST
 autonomous-system 45
 network 50.50.50.5 0.0.0.0
 network 51.51.51.5 0.0.0.0
 network 148.1.45.5 0.0.0.0
 no auto-summary
```


Task 4.2 Breakdown

This task is a continuation of the previous one. A tunnel connecting R4 and R5 is deployed. All interfaces are isolated in the same VRF making it a VRF-lite solution. EIGRP is enabled in the new VRF to provide transparent routing on all VRF interfaces. MPLS tunneling and MP-BGP are not used in this scenario.

🕮 Deep Dive

- What is the main scalability problem with VRF-lite?

Task 4.2 Verification

```
Rack1R4#ping vrf R4TEST 148.1.45.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 148.1.45.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 172/172/176
ms

Rack1R4#show ip route vrf R4TEST eigrp
     51.0.0.0/24 is subnetted, 1 subnets
D       51.51.51.0 [90/27008000] via 148.1.45.5, 00:49:17, Tunnel45
     50.0.0.0/24 is subnetted, 1 subnets
D       50.50.50.0 [90/27008000] via 148.1.45.5, 00:49:17, Tunnel45

Rack1R5#show ip route vrf R5TEST eigrp
     40.0.0.0/24 is subnetted, 1 subnets
D       40.40.40.0 [90/27008000] via 148.1.45.4, 00:50:30, Tunnel45
     41.0.0.0/24 is subnetted, 1 subnets
D       41.41.41.0 [90/27008000] via 148.1.45.4, 00:50:30, Tunnel45

Rack1R5#ping vrf R5TEST 41.41.41.4 source 50.50.50.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 41.41.41.4, timeout is 2 seconds:
Packet sent with a source address of 50.50.50.5
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 172/173/176
ms
```


Task 5.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Multicast > PIM Dense Mode
Multicast > Multicast RPF Failure

R1:
```
interface Tunnel0
  ip unnumbered Loopback0
 ip pim dense-mode
 tunnel source Loopback0
 tunnel destination 150.1.3.3
!
ip mroute 0.0.0.0 0.0.0.0 Tunnel0
```

R3:
```
interface Tunnel0
  ip unnumbered Loopback0
 ip pim dense-mode
 tunnel source Loopback0
 tunnel destination 150.1.1.1
```

R6:
```
interface FastEthernet0/0
 ip igmp join-group 224.6.6.6
```


Task 5.1 Breakdown

Multicast traffic from R3 to R6 needs to traverse the frame-relay network where R2 is the hub and R1, R3 are spokes. R2 will receive multicast traffic on its frame-relay interface and needs to send it back out on the same interface, which is not possible. In order to overcome this problem, there are two options:

- Configure sub-interfaces on R2, one for its connection to R1 and one for its connection to R3. Alternatively, tunnels could be used.
- Configure PIM NBMA mode on R2's Frame-Relay interface. This option is not supported with PIM Dense mode.

Since we cannot create new sub-interfaces, we resort to the tunnel option. IP unnumbered has been configured on tunnels as the task does not specify the use of any additional subnets.

Task 5.1 Verification

Verify that R6 may now receives the multicast feed:

```
Rack1R3#ping 224.6.6.6 source fa0/0 repeat 5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 224.6.6.6, timeout is 2 seconds:

Reply to request 0 from 148.1.68.6, 176 ms
Reply to request 1 from 148.1.68.6, 156 ms
Reply to request 2 from 148.1.68.6, 156 ms
Reply to request 3 from 148.1.68.6, 156 ms
Reply to request 4 from 148.1.68.6, 156 ms
```

Verify the multicast routing tables:

```
Rack1R1#show ip mroute 224.6.6.6
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.6.6.6), 00:00:13/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Tunnel0, Forward/Dense, 00:00:13/00:00:00
    Serial0/0, Forward/Dense, 00:00:13/00:00:00
    FastEthernet0/0, Forward/Dense, 00:00:13/00:00:00

(148.1.3.3, 224.6.6.6), 00:00:13/00:02:52, flags: T
  Incoming interface: Tunnel0, RPF nbr 150.1.3.3, Mroute
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:00:14/00:00:00
    Serial0/0, Forward/Dense, 00:00:14/00:00:00
```


Task 5.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Multicast > Multicast RPF Failure

R2:
```
ip multicast rpf backoff 300 5000
```

Task 5.2 Breakdown

The task requires changing triggered RPF interval value. Normally you tune this timer in situations where you want to improve multicast convergence.

🕮 Deep Dive

- What other factors are important for fast multicast flow re-convergence after a link failure?

Task 5.2 Verification

Verify the backoff timers:

```
Rack1R2#show ip rpf events
Last 15 triggered multicast RPF check events

RPF backoff delay: 300 msec
RPF maximum delay: 5 sec

DATE/TIME             BACKOFF    PROTOCOL   EVENT         RPF CHANGES
Mar 1 09:15:40.833    500 msec   RIP        Route UP        0
Mar 1 09:15:18.829    500 msec   RIP        Route UP        0
Mar 1 05:31:12.802    500 msec   Connected  Route UP        0
<output omitted>
```


Task 5.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Multicast > IGMP Timers

R6:
```
interface FastEthernet0/1
  ip igmp query-interval 20
  ip igmp query-max-response-time 16
```

SW2:
```
interface Vlan 68
 ip igmp query-interval 20
```

Task 5.3 Breakdown

The designated IGMPv2/3 querier on a segment is elected based on IP address, the lowest IP address being preferred. Detection of the designated querier failure is based on the **ip igmp querier-timeout** value on all non-designated queriers on the segment. This value defaults to 120s and the task requires that this timer to be three times faster than by default which is 40s.

🕮 Deep Dive

- How do end nodes use the IGMP query max response time?

Task 5.3 Verification

Verify timers using **the show igmp interface** command.

```
Rack1R6#show ip igmp interface fastEthernet 0/1
FastEthernet0/1 is up, line protocol is up
<snip>
  IGMP query interval is 20 seconds
  IGMP querier timeout is 40 seconds
  IGMP max query response time is 4 seconds
<snip>
  IGMP querying router is 148.1.68.6 (this system)

Rack1SW1#show ip igmp interface Vlan 68
<snip>
  IGMP querier timeout is 60 seconds
  IGMP max query response time is 10 seconds
<snip>

Rack1SW2#show ip igmp interface vlan 68
<snip>
  IGMP query interval is 20 seconds
  IGMP querier timeout is 40 seconds
  IGMP max query response time is 10 seconds
<snip>
```


Task 6.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Traffic Filtering with Extended Access-Lists

R6:

```
interface Serial0/0/0
 ip access-group RFC1918 in
!
ip access-list extended RFC1918
 deny   ip 10.0.0.0 0.255.255.255 any
 deny   ip 172.16.0.0 0.15.255.255 any
 deny   ip 192.168.0.0 0.0.255.255 any
 permit ip any any
```

Task 6.1 Breakdown

RFC 1918 specifies that the address ranges 10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16 are reserved for private usage. As these addresses are not publicly routable on the Internet, there is no valid reason for traffic to be received on an outside interface which is sourced from a host in this range. Therefore, best security practice dictates that these addresses should be denied from entering the network.

Task 6.1 Verification

```
Rack1R6#show ip interface serial 0/0/0 | section Inbound
  Inbound  access list is RFC1918

Rack1R6#show ip access-lists RFC1918
Extended IP access list RFC1918
    10 deny ip 10.0.0.0 0.255.255.255 any
    20 deny ip 172.16.0.0 0.15.255.255 any
    30 deny ip 192.168.0.0 0.0.255.255 any
    40 permit ip any any (64 matches)
```

Task 6.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Traffic Filtering with Extended Access-Lists
Security > Traffic Filtering with Time-Based Access-Lists

R5:

```
ip access-list extended DENY_INTERNET_SURFING
 permit tcp 148.1.5.0 0.0.0.255 host 148.1.3.100 eq 80 443
 permit tcp 148.1.5.0 0.0.0.255 any eq 80 443 time-range NON_WORK_HOURS
 deny tcp 148.1.5.0 0.0.0.255 any eq 80 443
 permit ip 148.1.5.0 0.0.0.255 any
!
time-range NON_WORK_HOURS
 periodic weekend 0:00 to 23:59
 periodic weekdays 0:00 to 8:59
 periodic weekdays 17:00 to 23:59
!
interface FastEthernet0/1
 ip access-group DENY_INTERNET-SURFING in
```


Task 6.2 Breakdown

Newer IOS versions allow multiple non-consecutive ports to be matched on a single ACL line. Here, both HTTP and HTTPS are matched in the same line. A second ACL entry is needed to match the time-range and allow during the non-work hours.

When dealing with multiple possible paths, you may need to apply the policy to multiple interfaces.

🕮 Deep Dive

- Why time-based access-lists are not effective at high traffic-rates?

Task 6.2 Verification

Check that the time range is active:

```
Rack1R5#show time-range
time-range entry: NON_WORK_HOURS (active)
   periodic weekend 0:00 to 23:59
   periodic weekdays 0:00 to 8:59
   periodic weekdays 17:00 to 23:59
   used in: IP ACL entry
```

Confirm that access-list has been applied

```
Rack1R5#show ip interface fastEthernet 0/1 | section Inbound
  Inbound  access list is DENY_INTERNET-SURFING
```

Check that time-based entries are active in the ACL

```
Rack1R5#show ip access-lists  DENY_INTERNET_SURFING
Extended IP access list DENY_INTERNET_SURFING
    10 permit tcp 148.1.5.0 0.0.0.255 host 148.1.3.100 eq www 443
    20 permit tcp 148.1.5.0 0.0.0.255 any eq www 443 time-range
NON_WORK_HOURS (active)
    30 deny tcp 148.1.5.0 0.0.0.255 any eq www 443
    40 permit ip 148.1.5.0 0.0.0.255 any
```


Task 6.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Traffic Filtering with Extended Access-Lists
Security > Controlling Terminal Line Access

R2:

```
access-list 99 permit 150.1.0.0 0.0.255.255
!
access-list 100 permit ip any host 150.1.3.3
access-list 100 permit ip any host 192.10.1.3
access-list 100 permit ip any host 148.1.35.3
access-list 100 permit ip any host 148.1.0.3
access-list 100 permit ip any host 148.1.3.3

!
username TELNET password CISCO
username TELNET access-class 100
!
line vty 0 181
 access-class 99 in
 login local
```


Task 6.3 Breakdown

The task involves two requirements: blocking telnet access to R2 from any subnets other that 150.X.0.0/16 and limiting a user logged into R2 from telnetting outbound. The first requirement is fulfilled using VTY access class, and the second one is implemented using an access-list bound to the said user. Access-lists bound to local users are used to filter outgoing connections initiated from exec mode by the user. Notice that the default "line" authentication has been changed to "local" to let the user logging in identify itself.

Since we are to restrict use connecting from R2 on R2 itself, we have to list all the IP addresses belonging to R3. It normally makes more sense to control devices access at the device itself.

🕮 Deep Dive

- What other means can you use to control inbound telnet access to a router?

Task 6.3 Verification

Make sure you can only connect to R2 using a Loopback subnets as a source:

```
Rack1R5#telnet 150.1.2.2
Trying 150.1.2.2 ...
% Connection refused by remote host

Rack1R5#telnet 150.1.2.2 /source Loopback0
Trying 150.1.2.2 ... Open


User Access Verification

Username: TELNET
Password: CISCO
```

Try connecting from R2 to different destinations and ensure that you can only connect to R3:

```
Rack1R2>telnet 150.1.1.1
Trying 150.1.3.3 ...
% Connections to that host not permitted from this terminal

Rack1R2>telnet 150.1.3.3
Trying 150.1.1.1 ... Open

User Access Verification

Password: cisco

Rack1R1>exit

[Connection to 150.1.1.1 closed by foreign host]
```


Task 6.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IP Services > Router ICMP Settings

```
R6:
interface Serial 0/0/0
 no ip unreachables
!
ip icmp rate-limit unreachable 10
ip icmp rate-limit unreachable DF 1
```

Task 6.4 Breakdown

Controlling the rate of ICMP unreachable message is an important security feature, as with default settings router may be slowed down by sending too many ICMP responses. Limiting the rate of ICMP messages could have adverse side effects, such making pMTU less relevant. This is why the “packet too big” messages have separate tuning option by means of “DF-bit” unreachables.

🕮 Deep Dive

- What other ways of limiting router ICMP responses you may suggest?

Task 6.4 Verification

```
Rack1R6#show ip interface serial 0/0/0 | section ICMP
  ICMP redirects are always sent
  ICMP unreachables are never sent
  ICMP mask replies are never sent

Rack1R6#show ip icmp rate-limit

                       DF bit unreachables       All other unreachables
Interval (millisecond)     1                         10

Interface            # DF bit unreachables     # All other unreachables
---------            ---------------------     ------------------------
FastEthernet0/0            0                         0
FastEthernet0/1            0                         0
Serial0/0/0                0                         0
SSLVPN-VIF0                0                         0
Loopback0                  0                         0
```


Task 7.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

System Management > Generating Exception Core Dumps

R6:

```
ip ftp username R6CORE
ip ftp password CISCO
exception dump 148.1.3.100
exception protocol ftp
exception core-file R6DUMP.txt
!
ip default-gateway 148.1.68.8
```

Task 7.1 Breakdown

When a router crashes, it can be possible to create a core dump of what is currently in RAM. This output can be useful for TAC to troubleshoot a hardware or software problem. Core dumps can be created through TFTP, FTP, RCP, or direct to flash.

Task 7.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

System Management > NTP

R5:
```
ntp server 204.12.1.254
ntp peer 148.1.57.7
ntp master 5
```

SW1:
```
ntp server 204.12.1.254
```


Task 7.2 Verification

Verify NTP status and associations:

```
Rack1R5#show ntp associations

     address   ref clock    st  when  poll reach  delay  offset    disp
+~127.127.7.1  127.127.7.1   4     7    64  377     0.0    0.00     0.0
+~148.1.57.7   204.12.1.254  5    39    64  377     2.9   -2.49     2.2
*~204.12.1.254 127.127.7.1   4    62    64   37     7.5    0.70   876.3
 * master (synced), # master (unsynced), + selected, - candidate, ~
configured

Rack1R5#show ntp status
Clock is synchronized, stratum 5, reference is 204.12.1.254
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is
2**18
reference time is C83A20EA.5AFD0A73 (05:45:14.355 UTC Wed Jun 14 2006)
clock offset is 0.0000 msec, root delay is 6.87 msec
root dispersion is 378.75 msec, peer dispersion is 377.79 msec

Rack1SW1#show ntp associations

     address   ref clock    st  when  poll reach  delay  offset    disp
+ 148.1.57.5   204.12.1.254  5     4    64    7     0.8    3.52  7876.7
*~204.12.1.254 127.127.7.1   4     3    64  377     7.4    6.35     5.3
 * master (synced), # master (unsynced), + selected, - candidate, ~
configured

Rack1SW1#show ntp status
Clock is synchronized, stratum 5, reference is 204.12.1.254
nominal freq is 250.0000 Hz, actual freq is 249.9998 Hz, precision is
2**18
reference time is C83A20F2.E8E44FFD (05:45:22.909 UTC Wed Jun 14 2006)
clock offset is 6.3512 msec, root delay is 7.40 msec
root dispersion is 13.78 msec, peer dispersion is 7.40 msec
```


Task 7.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

System Management > NTP Authentication

R5:
```
ntp authentication-key 1 md5 CISCO
ntp authenticate
ntp trusted-key 1
ntp server 204.12.1.254 key 1 prefer
ntp peer 148.1.57.7 key 1
```

SW1:
```
ntp authentication-key 1 md5 CISCO
ntp authenticate
ntp trusted-key 1
ntp server 204.12.1.254 key 1 prefer
```

Task 7.3 Breakdown

The authentication process is straightforward – you only control the clients, so you need to set up authentication keys with proper key index (remember, it should match between the client and the server). Remember to authenticate the NTP peering session as well, otherwise you will lose it.

Task 7.3 Verification

Verify NTP authentication:

```
Rack1SW1#show ntp associations detail
204.12.1.254 configured,authenticated,our_master,sane, valid, stratum 4
ref ID 127.127.7.1, time C83A2190.82116150 (05:48:00.508 UTC Wed Jun 14
2006)
our mode client,peer mode server, our poll intvl 64, peer poll intvl 64
root delay 0.00 msec, root disp 0.03, reach 377, sync dist 4.959
delay 5.57 msec, offset 9.4310 msec, dispersion 2.15
precision 2**19, version 3
org time C83A21B2.EB5345F8 (05:48:34.919 UTC Wed Jun 14 2006)
rcv time C83A21B2.E99FB4B5 (05:48:34.912 UTC Wed Jun 14 2006)
xmt time C83A21B2.E7F278EE (05:48:34.906 UTC Wed Jun 14 2006)
filtdelay = 5.57   5.98    6.99    7.40    7.87    6.04    5.81    5.89
filtoffset =9.43   8.71    7.74    6.35    3.80    0.24    0.09    0.09
filterror = 0.02   0.99    1.97    2.94    3.92    4.90    4.91    4.93

Rack1R5#show ntp associations detail | begin 204.12.1.254 config
204.12.1.254 configured,authenticated,our_master,sane, valid, stratum 4
ref ID 127.127.7.1, time C83A21D0.8268E8C0 (05:49:04.509 UTC Wed Jun 14
2006)
our mode client, peer mode server, our poll intvl 64,peer poll intvl 64
root delay 0.00 msec, root disp 0.03, reach 377, sync dist 5.737
delay 9.26 msec, offset 3.7436 msec, dispersion 0.99
precision 2**19, version 3
org time C83A21DB.5BB3F3DE (05:49:15.358 UTC Wed Jun 14 2006)
rcv time C83A21DB.5BEE48E6 (05:49:15.359 UTC Wed Jun 14 2006)
xmt time C83A21DB.59558521 (05:49:15.348 UTC Wed Jun 14 2006)
filtdelay = 9.26  8.65    7.48    8.24    6.87    7.48    6.96    6.88
filtoffset =3.74  3.49    2.56    2.26    0.94    0.70   -0.65   -0.71
filterror = 0.02  0.76    1.74    2.72    3.69    4.67    5.65    5.66
```


Task 7.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IP Services > TCP Optimization

R1:
```
ip tcp synwait-time 5
```

Task 7.4 Breakdown

The TCP SYN wait time is the time the router will wait after sending a TCP SYN packet for a SYN/ACK to come back. If the SYN/ACK response has not been received before the timer expires, the connection is reset.

Task 7.5 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IP Services > IP Output Packet Accounting

R6:
```
interface Serial0/0/0
 ip accounting output-packets
!
ip accounting-threshold 1000
```


Task 7.5 Verification

Verify IP accounting:

```
Rack1SW2#ping 117.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 117.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/36 ms

Rack1SW2#ping 118.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 118.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/33/36 ms

Rack1SW2#ping 119.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 119.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/36 ms

Rack1R6#show ip accounting
   Source          Destination              Packets              Bytes
 148.1.68.8      117.0.0.1                      5                500
 148.1.68.8      118.0.0.1                      5                500
 148.1.68.8      119.0.0.1                      5                500
```


Task 7.6 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IP Services > TCP Load Distribution with NAT

```
R3:
!
! Enable NAT at the interface level
!
interface FastEthernet0/0
 ip nat inside
!
interface FastEthernet0/1
 ip nat outside
!
interface Serial1/0.302 point-to-point
 ip nat outside
!
interface Serial1/1
 ip nat outside
!
! The pool of "inside" addresses
!
ip nat pool REAL_SERVERS 148.1.3.110 148.1.3.112 prefix-length 24 type
rotary
ip nat inside destination list OLD_WEB_SERVER pool REAL_SERVERS

!
! The access-list defines IP addresses to be redirected
!
ip access-list extended OLD_WEB_SERVER
 permit tcp any host 148.1.3.100 eq www
 permit tcp any host 148.1.3.100 eq 8080
 permit tcp any host 148.1.3.100 eq 443
```


Task 7.6 Breakdown

Destination-based NAT is normally used for load-balancing purpose. You may use it for service redirection, but this task could be better accomplished using static NAT mapping. In this particular scenario, we configure destination-based NAT to distributed connections on three different ports among pool of servers, that all look as a single IP from the outside (the old IP address).

🕮 Deep Dive

- What is a better alternative to using destination-based NAT?

Task 7.6 Verification

```
Rack1R1#telnet 148.1.3.100 80
Trying 148.1.3.100, 80 ...
Rack1AS>3
[Resuming connection 3 to r3 ... ]

Rack1R3#show ip nat translations
Pro Inside global  Inside local     Outside local      Outside global
tcp 148.1.3.100:80 148.1.3.110:80   148.1.0.1:11004    148.1.0.1:11004

Rack1AS>6
[Resuming connection 6 to r6 ... ]

Rack1R6#telnet 148.1.3.100 80
Trying 148.1.3.100, 80 ...

Rack1AS>3
[Resuming connection 3 to r3 ... ]

Rack1R3#show ip nat translations
Pro Inside global  Inside local     Outside local      Outside global
tcp 148.1.3.100:80 148.1.3.110:80   148.1.0.1:11004    148.1.0.1:11004
tcp 148.1.3.100:80 148.1.3.111:80   148.1.68.6:21387   148.1.68.6:21387

Rack1AS>8
[Resuming connection 8 to SW2 ... ]

Rack1SW2#telnet 148.1.3.100 80
Trying 148.1.3.100, 80 ...

Rack1AS>3
[Resuming connection 3 to r3 ... ]

Rack1R3#show ip nat translations
Pro Inside global   Inside local    Outside local      Outside global
tcp 148.1.3.100:80  148.1.3.110:80  148.1.0.1:11004    148.1.0.1:11004
tcp 148.1.3.100:80  148.1.3.111:80  148.1.68.6:21387   148.1.68.6:21387
tcp 148.1.3.100:80  148.1.3.112:80  148.1.18.8:11000   148.1.18.8:11000
```


Task 8.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC Class-Based Generic Traffic Shaping

R6:

```
policy-map SHAPE
  class class-default
    shape average 5000000 160000 80000
!
interface Serial0/0/0
 bandwidth 45000
 service-policy output SHAPE
```

Task 8.1 Breakdown

For a rate of 5 Mbps, 5 kilobits are sent every millisecond. For a time interval of 32 ms, the corresponding number of bits is 160,000 bits. For an excess burst of 50% of the rate, the excess burst value will be 80,000 bits.

Task 8.1 Verification

Verify MQC configuration:

```
Rack1R6#show policy-map interface s0/0/0

 Serial0/0/0

  Service-policy output: SHAPE

    Class-map: class-default (match-any)
      1 packets, 13 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 4/83
      shape (average) cir 5000000, bc 160000, be 80000
      target shape rate 5000000
        lower bound cir 0,  adapt to fecn 0
```


Task 8.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC Class-Based GTS and CBWFQ
QoS > MQC Classification and Marking

R6:

```
class-map match-any PEER-TO-PEER
 match protocol fasttrack
 match protocol gnutella
 match protocol kazaa2
!

policy-map QOS_POLICY
  class PEER-TO-PEER
   police cir 8000
!
policy-map SHAPE
  class class-default
   service-policy QOS_POLICY
```

Task 8.1 Breakdown

The task uses NBAR classification to match on P2P protocols that use dynamic port numbers. The traffic is then policed to a very low rate, ensuring it does not interfere with the “useful” traffic.

Task 8.2 Verification

Verify the new MQC configuration:

```
Rack1R6#show policy-map interface s0/0/0

Serial0/0/0

  Service-policy output: SHAPE

    Class-map: class-default (match-any)
      45 packets, 2225 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 45/1021
      shape (average) cir 5000000, bc 160000, be 80000
      target shape rate 5000000
        lower bound cir 0,  adapt to fecn 0

      Service-policy : QOS_POLICY

        Class-map: PEER-TO-PEER (match-any)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: protocol fasttrack
            0 packets, 0 bytes
            5 minute rate 0 bps
          Match: protocol gnutella
            0 packets, 0 bytes
            5 minute rate 0 bps
          Match: protocol kazaa2
            0 packets, 0 bytes
            5 minute rate 0 bps
          police:
              cir 8000 bps, bc 1500 bytes
            conformed 0 packets, 0 bytes; actions:
              transmit
            exceeded 0 packets, 0 bytes; actions:
              drop
            conformed 0 bps, exceed 0 bps

        Class-map: class-default (match-any)
          6 packets, 163 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: any
```


Task 8.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC Class-Based GTS and CBWFQ
QoS > MQC Classification and Marking

R6:
```
class-map match-all HTTP
  match protocol http
!
class-map match-all SMTP
  match protocol smtp
!
policy-map QOS_POLICY
  class HTTP
    bandwidth 2000
  class SMTP
    bandwidth 1000
```

Task 8.3 Breakdown

The same NBAR classification is used to match other protocols dynamically. Matching simple protocols does not take much NBAR resources (e.g. memory space) as those are normally classified using the port numbers.

Task 8.3 Verification

Confirm the MQC configuration changes:

```
Rack1R6#show policy-map interface s0/0/0 | begin HTTP
<snip>
        Class-map: HTTP (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: protocol http
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
          bandwidth 2000 kbps

        Class-map: SMTP (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: protocol smtp
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
          bandwidth 1000 kbps

        Class-map: class-default (match-any)
          52 packets, 2564 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: any

          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 3/70
```


Task 8.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC LLQ and Remaining Bandwidth Reservation

R6:

```
class-map match-all #CISCO_UNDERNET_IRC
  match access-group name CISCO_UNDERNET_IRC
!
policy-map QOS_POLICY
  class #CISCO_UNDERNET_IRC
    priority 32
!
ip access-list extended CISCO_UNDERNET_IRC
 permit tcp host 148.1.6.10 any eq 6667
```


Task 8.4 Verification

Verify MQC priority configuration:

```
Rack1R6#show policy-map interface s0/0/0 | begin UNDERNET

        Class-map: #CISCO_UNDERNET_IRC (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: access-group name CISCO_UNDERNET_IRC
          Priority: 32 kbps, burst bytes 1500, b/w exceed drops: 0


        Class-map: class-default (match-any)
          51 packets, 1441 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: any

          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 22/498
```

Task 8.5 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > Selective Packet Discard

R3:
```
spd extended-headroom 100
spd headroom 50
!
ip spd mode aggressive
ip spd queue max-threshold 75
ip spd queue min-threshold 37
```

Task 8.5 Breakdown

Selective Packet Discard is an important routing for handling input route queue. Together with control plane protection is allows for flexible protection of control plane components such as signaling/routing protocols. SPD become critical part of router architecture in environments where router has to handle large amount of control plane traffic, such as in large-scale BGP route-reflector deployments. SPD commands are hidden as changing them without necessary precaution may severely impact router's performance.

🕮 Further Learning

Understanding Selective Packet Discard

http://www.cisco.com/en/US/products/hw/routers/ps167/products_tech_note09186a008012fb87.shtml

Task 8.5 Verification

```
Rack1R3#show ip spd
Current mode: normal.
Queue min/max thresholds: 37/75, Headroom: 50, Extended Headroom: 100
IP normal queue: 2, priority queue: 0.
SPD special drop mode: aggressively drop bad packets
```


VOL1 Scenario Reference

Bridging and Switching > STP Root Guard
Bridging and Switching > STP Root Bridge Election
Bridging and Switching > 802.1q Tunneling
Bridging and Switching > Layer 3 Etherchannel
Bridging and Switching > Etherchannel over 802.1q Tunneling
Bridging and Switching > Tuning STP Convergence Timers
Frame-Relay > Inverse-ARP
Frame-Relay > Point-to-Point Subinterfaces
IP Routing > Backup Interface
OSPF > OSPF MD5 Authentication
OSPF > Miscellaneous OSPF Features
EIGRP > EIGRP Convergence Timers
EIGRP > EIGRP MD5 Authentication
EIGRP > EIGRP Key-Chain Rotation
Security > VLAN Filters for IP Traffic
RIPv2 > RIPv2 Filtering with Passive Interfaces
RIPv2 > RIPv2 Filtering with Administrative Distance
EIGRP > EIGRP Summarization
BGP > BGP Auto-Summary
BGP > BGP Regular Expressions
BGP > BGP Aggregation
BGP > BGP Aggregation – Suppress Map
BGP > BGP Communities – No-Export
IPv6 > Automatic 6to4 Tunnels
IPv6 > IPv6 Filtering
MPLS VPN > VRF Lite
MPLS VPN > VRF Lite
MPLS VPN > PE-CE Routing with EIGRP
Multicast > PIM Dense Mode
Multicast > Multicast RPF Failure
Multicast > IGMP Timers

Security > Traffic Filtering with Extended Access-Lists
Security > Traffic Filtering with Extended Access-Lists
Security > Traffic Filtering with Time-Based Access-Lists
Security > Traffic Filtering with Extended Access-Lists
Security > Controlling Terminal Line Access
IP Services > Router ICMP Settings
System Management > Generating Exception Core Dumps
System Management > NTP
System Management > NTP Authentication
IP Services > TCP Optimization
IP Services > IP Output Packet Accounting
QoS > MQC Class-Based Generic Traffic Shaping
QoS > MQC Class-Based GTS and CBWFQ
QoS > MQC Classification and Marking
QoS > MQC Class-Based GTS and CBWFQ
QoS > MQC LLQ and Remaining Bandwidth Reservation
QoS > Selective Packet Discard

Deep Dive Answers

- Why do you need a separate VLAN for every Etherchannel link tunneled using QinQ?
    - Etherchannel protocols such as PaGP and LACP assume the underlying link is physically point to point. When running Etherchannel over an emulated point-to-point link, you need to implement the same behavior. If you are sharing the same VLAN among multiple ports, you effectively introduce protocol interference between different ports. Even when not using any link aggregation protocols you should use separate VLAN for every pair of ports connected via a tunnel or you will end up sending traffic to wrong ports.
- Is it possible to disable MAC address learning on the VLANs used to tunnel P2P traffic?
    - Yes, since the transport VLANs are in essence emulated point-to-point wires it is possible to disable MAC address learning without losing any efficiency. There is a command in the 3560 switches, **`no mac-address learning`** that could be used for this purpose.
- What other options you can use to implement backup routing as opposed to using backup interfaces?
    - Backup link uses the idea of "standby" physical interfaces, or active-standby pair in general. Other options include layer 2 or layer 3 mechanics, such as link bundling (if possible), dynamic routing of floating static routing. Floating static routing become an attractive alternative for small remote nodes, especially when combined with IP SLA object tracking.
- What regular expression would you use to accept BGP prefixes from your upstream customers, provided that the customers may use AS_PATH prepending?
    - You need to use the special "backtrack" reference for this. Let's say your upstream AS is 54, then the regular expression

      ```
      ^54(_[0-9]+)(\1)*
      ```

      allows for matching the upstream AS, the upstream peer AS and any number of repetitions of the peer AS (\1 represent backtrack to the previous match). Since there is no trailing "$" this regular expression also accommodates for any sub-AS in the end of the path. If you put the anchor $ in the end, you will match the routes originated in the peer AS, no matter how many prepends were applied.
- Is dynamic routing possible over 6to4 tunnels?

  - IGP cannot properly interpret a 6to4 tunnel as a single shared link and thus cannot properly exchange updates of form adjacencies unless a protocol similar to NHRP is used. However, BGP could be used over a 6to4 cloud to exchange IPv6 prefixes, thank to the recursive next-hop resolution. For every packet routed to a prefix learned via BGP over a 6to4 tunnel, the next hop will be built based on the NEXT_HOP attribute and proper tunnel header will be constructed. This allows for scalable overlay 6to4 networks.

- Do 6to4 tunnels have to use the fixed prefix 2002::/16?
  - No, this prefix was selected as a common ground to allow for dynamic prefix allocation to Internet sites. The tunnel behavior is not based on the prefix but rather the tunnel mode (configured manually) which ensures proper encapsulation into IPv4 header.
- What is the main scalability problem with VRF-lite?
  - VRF lite runs native protocol adjacencies over multiplexed links and does not share a single routing protocol in the network core. Furthermore, it requires separate cloud of tunnels in the network core for every VPN implemented. This significantly increases provisioning efforts and resource consumption, allowing for small deployments only.
- What other factors are important for fast multicast flow re-convergence after a link failure?
  - The main factor is fast IGP convergence, as multicast routing is bound to unicast routing table. With properly tuned IGP, RPF triggered backoff could be set to a minimum, to allow fast reaction to unicast events. Another good idea to protect multicast flows is using multicast multi-paths in the network, so that a single failure affects fewer flows.
- How do end nodes use the IGMP query max response time?
  - IGMP query max response time is integral part of IGMPv2 messages. It allows for suppressing storm of reports when multicast router queries for group membership. Every host selects a random interval within the range of maximum response time and delays report for this random interval. If it hears other host reporting the same group it dismisses its own report.
- Why time-based access-lists are not effective at high traffic-rates?
  - Time-based access-lists are hard to implement in hardware due to entry expiration and external polling process required to refresh the entries. The same problem plagued the reflexive-ACLs, which were dynamic and time based.
- What other means can you use to control inbound telnet access to a router?

    - It is possible to use local policy routing for the same purpose or control plane policing/protection to limit “aggregate” access to the router on selected ports.
- What other ways of limiting router ICMP responses you may suggest?
    - It is possible to completely disable ICMP unreachables on per-interface basis using the command **`no ip unreachables`** or use control-plane policing to limit the outbound traffic.
- What is a better alternative to using destination-based NAT?
    - Any dedicated server-load balancing solution, as destination-based NAT only allows for quick fix and cannot support large traffic volumes. Cisco has special solutions for server load-balancing and high-end switching platforms also support SLB (server load balancing), which is an appropriate replacement for the NAT-based load-balancing.

Lab 10 Solutions

Task 2.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

EIGRP > EIGRP Unequal-Cost Load-Balancing

R1:
```
interface Serial0/0
 bandwidth 256

!
interface Serial0/1
 bandwidth 1536
!
router eigrp 100
 variance 5
```

R3:
```
interface Serial1/1.23 point-to-point
 bandwidth 1280
```

Task 2.1 Breakdown

EIGRP is the only IGP that supports unequal cost load balancing. In order to enable this load balancing issue the **variance** command under the EIGRP process. In order for a path to be considered for unequal cost load balancing it must be a feasible successor with a metric less than or equal to the successor's metric times the variance.

To choose the best path through the network and prevent looping EIGRP's route selection uses the *feasibility condition*. In order to understand this calculation it is important to understand the difference between advertised distance and local distance. Advertised distance is the metric reported by the upstream neighbor as their cost to the destination. Local distance is the metric from the local device to the upstream neighbor.

First the local router looks through all advertised paths and chooses the path with the lowest advertised distance plus local distance. Like other protocols this is simply the lowest end to end metric for the path. The metric for this path is called the *feasible distance*. The path itself called the *successor*. The successor is the best route to the destination.

Once the successor has been found EIGRP does an additional check to see if there may be alternate paths throughout the network. These alternate paths are known as *feasible successors*. These are paths that could be (are feasible to be) the successor if the successor is lost. A path whose advertised distance is lower than the feasible distance of the successor is deemed a feasible successor. In the case that a router is advertising a lower distance than the local device is using as its successor it can be guaranteed that there is not a loop in the topology.

Now that the successor and all feasible successors have been chosen the router does a final check based on the input variance value to determine which feasible successors can be installed in the IP routing table along with the successor. If the end to end metric of a feasible successor is less than or equal to the metric of the successor times the variance it is valid to be installed as an additional path. EIGRP unequal cost load balancing also does efficient traffic sharing. For example if the successor has a metric of one and the feasible successor has a metric of two, two packets will be sent out the successor's path and one packet will be sent out the feasible successor's path. This ensures that higher bandwidth paths are more utilized than lower bandwidth paths.

In the above task, R1 is to be configured to send traffic out to the destination 164.X.26.0/24 to both R3 and R2 in a ratio of 5:1 respectively. In addition to this the question specifies what the underlying bandwidths of the network circuits are. The first step in accomplishing this goal is to set the appropriate **`bandwidth`** statement on the interface. In the above configuration this is done on the outgoing interfaces to reach the destination. Typically the bandwidth value is configured on both ends of the link to be the same value, but in this case it is not required to accomplish the goal.

After the **bandwidth** values are set the following output is seen on R1:

```
Rack1R1#show ip eigrp topology 164.1.26.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 164.1.26.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is
3026432
  Routing Descriptor Blocks:
  164.1.13.3 (Serial0/1), from 164.1.13.3, Send flag is 0x0
      Composite metric is (3026432/2514432), Route is Internal
      Vector metric:
        Minimum bandwidth is 1280 Kbit
        Total delay is 40100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
  164.1.12.2 (Serial0/0), from 164.1.12.2, Send flag is 0x0
      Composite metric is (10514432/28160), Route is Internal
      Vector metric:
        Minimum bandwidth is 256 Kbit
        Total delay is 20100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
```


From this output we can see that R1 has two paths, one through R3 and one through R2. The path through R3 has a metric of 3026432, while the path through R2 has a metric of 10514432. Since the metric through R3 is less it is the successor. Next the feasibility check is run, and R2's advertised distance of 28160 is compared against the feasible distance of 3026432. Since R2's advertised distance is less than the feasible distance the route through R2 is a feasible successor.

At this point if the **variance** command was configured traffic would be load balanced between R3 and R2 in a ratio of 10514432:3026432, or approximately 80:23. This can be seen in the **show ip route 164.1.26.0** output on R1:

```
Rack1R1#show ip route 164.1.26.0
Routing entry for 164.1.26.0/24
  Known via "eigrp 100", distance 90, metric 3026432, type internal
  Redistributing via eigrp 101
  Last update from 164.1.13.3 on Serial0/1, 00:04:00 ago
  Routing Descriptor Blocks:
  * 164.1.12.2, from 164.1.12.2, 00:04:00 ago, via Serial0/0
      Route metric is 10514432, traffic share count is 23
      Total delay is 20100 microseconds, minimum bandwidth is 256 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
    164.1.13.3, from 164.1.13.3, 00:04:00 ago, via Serial0/1
      Route metric is 3026432, traffic share count is 80
      Total delay is 40100 microseconds, minimum bandwidth is 1280 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
```

In order to achieve the desired ratio of 5:1 we must now modify the metric through R2 to be 5 times that of R3's metric, while still keeping the route through R2 a feasible successor. The easiest way to do this is to change the **delay** on R1's connection to R2 over the Frame Relay cloud. To determine the correct delay value we must first determine how the current composite metric value is derived. EIGRP metric calculation uses the formula:

Metric = [k1 * bandwidth + (k2 * bandwidth)/(256 - load) + k3 * delay] * [k5/(reliability + k4)]

The "k" values are derived from the **metric weights** command, where K1 and K3 are 1 by default and all other values are 0. This essentially means that only bandwidth and delay are taken into account. "Bandwidth" is the inverse bandwidth in Kbps times 10^7 ($10^7/BW_{Kbps}$). "Delay" is delay in tens of microseconds ($DLY_{usec}/10$). These values are added together and then scaled by a factor of 256. The composite metric is therefore represented by default as:

Metric = ($10^7/BW_{Kbps}$ + $DLY_{usec}/10$) * 256

Using the output from the **show ip eigrp topology 164.1.26.0 255.255.255.0** we can see that the metric through R3 has a minimum bandwidth value of 1280Kbps and a total delay of 40100 microseconds. The metric to R3 is then calculated as:

Metric_through_R3 = (10^7/1280 + 40100/10) * 256
Metric_through_R3 = (7812.5 + 4010) * 256
Metric_through_R3 = (11822.5) * 256
Metric_through_R3 ~ (11822) * 256
Metric_through_R3 ~ 3026432

In order to get our ratio of 5:1 we now need to modify our calculation as follows:

Metric_through_R3 * 5 = Metric_through_R2

Or more specifically:

(10^7/1280 + 40100/10) * 256 * 5 = (10^7/$BW_{Kbps-R2}$ + $DLY_{usec-R2}$/10) * 256

The value that we will modify through R2 is the delay, so we can use our current BW value to R2 of 256Kbps (as seen from the **show ip eigrp topology** output)

(10^7/1280 + 40100/10) * 256 * 5 = (10^7/$BW_{Kbps-R2}$ + $DLY_{usec-R2}$/10) * 256
(10^7/1280 + 40100/10) * 256 * 5 = (10^7/256 + $DLY_{usec-R2}$/10) * 256
(10^7/1280 + 40100/10) * 5 = (10^7/256 + $DLY_{usec-R2}$/10)
(7812.5 + 4010) * 5 = (39062.5 + $DLY_{usec-R2}$/10)
(7812 + 4010) * 5 ~ (39062 + $DLY_{usec-R2}$/10)
59110 ~ (39062 + $DLY_{usec-R2}$/10)
20048 ~ $DLY_{usec-R2}$/10
200480 ~ $DLY_{usec-R2}$

Based on this calculation we can see that if the end to end delay through R2 is 200480 the resulting composite metric through R2 will be five times that of through R3. Looking at the **show ip eigrp topology 164.1.26.0 255.255.255.0** output on R2 we can see that R2 already has a delay of 100 microseconds to reach this destination:


```
Rack1R2#show ip eigrp topology 164.1.26.0 255.255.255.0
IP-EIGRP (AS 101): Topology entry for 164.1.26.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 28160
  Routing Descriptor Blocks:
  0.0.0.0 (FastEthernet0/0), from Connected, Send flag is 0x0
      Composite metric is (28160/0), Route is Internal
      Vector metric:
        Minimum bandwidth is 100000 Kbit
        Total delay is 100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 0
```

This means that R1 should have a local delay to R2 of (200480 – 100), or 20038 *tens* of microseconds. Once the **`delay 20038`** command is configured on R1's Serial0/0 interface the traffic share is in a ratio of 5 to 1:

```
Rack1R1#show ip route 164.1.26.0
Routing entry for 164.1.26.0/24
  Known via "eigrp 101", distance 90, metric 3026432, type internal
  Redistributing via eigrp 101
  Last update from 164.1.13.3 on Serial0/1, 00:00:00 ago
  Routing Descriptor Blocks:
  * 164.1.12.2, from 164.1.12.2, 00:00:00 ago, via Serial0/0
      Route metric is 15132160, traffic share count is 1
      Total delay is 200480 microseconds, minimum bandwidth is 256 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
    164.1.13.3, from 164.1.13.3, 00:00:00 ago, via Serial0/1
      Route metric is 3026432, traffic share count is 5
      Total delay is 40100 microseconds, minimum bandwidth is 1280 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
```

🕮 Further Reading

How Does Unequal Cost Path Load Balancing (Variance) Work in IGRP and EIGRP?

For those of you that are not as good with calculating large equations, another method is to start with a low value and change the delay while looking at the output of **`show ip route`**: Starting without a delay configured, the ratio shows as 80/23.


```
Rack1R1(config-if)#do show ip route 164.1.26.0 | i from|share
  Last update from 164.1.12.2 on Serial0/0, 00:00:08 ago
  * 164.1.13.3, from 164.1.13.3, 00:00:08 ago, via Serial0/1
      Route metric is 3026432, traffic share count is 80
    164.1.12.2, from 164.1.12.2, 00:00:08 ago, via Serial0/0
      Route metric is 10514432, traffic share count is 23
```

After changing to a delay of 10,000, the ratio changes to 120/29.

```
Rack1R1(config-if)#delay 10000
Rack1R1(config-if)#do show ip route 164.1.26.0 | i from|share
  Last update from 164.1.12.2 on Serial0/0, 00:00:00 ago
  * 164.1.13.3, from 164.1.13.3, 00:00:00 ago, via Serial0/1
      Route metric is 3026432, traffic share count is 120
    164.1.12.2, from 164.1.12.2, 00:00:00 ago, via Serial0/0
      Route metric is 12562432, traffic share count is 29
```

After changing to a delay of 20000, the ratio changes to 5 to 1. This is not necessarily an exact value, due to how the router handles the forwarding internally, but it is fairly close.

```
Rack1R1(config-if)#delay 20000
Rack1R1(config-if)#do show ip route 164.1.26.0 | i from|share
  Last update from 164.1.12.2 on Serial0/0, 00:00:01 ago
  * 164.1.13.3, from 164.1.13.3, 00:00:01 ago, via Serial0/1
      Route metric is 3026432, traffic share count is 5
    164.1.12.2, from 164.1.12.2, 00:00:01 ago, via Serial0/0
      Route metric is 15122432, traffic share count is 1
```

🕮 Further Learning

Understanding Unequal Cost Load Balancing:

http://blog.ine.com/2009/05/01/understanding-unequal-cost-load-balancing/

Task 2.1 Verification

Verify the topology and routing table after load-balancing configuration has been configured:

```
Rack1R1#show ip eigrp topology 164.1.26.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 164.1.26.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is
3026432
  Routing Descriptor Blocks:
  164.1.13.3 (Serial0/1), from 164.1.13.3, Send flag is 0x0
      Composite metric is (3026432/2514432), Route is Internal
      Vector metric:
        Minimum bandwidth is 1280 Kbit
        Total delay is 40100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
  164.1.12.2 (Serial0/0), from 164.1.12.2, Send flag is 0x0
      Composite metric is (15132160/28160), Route is Internal
      Vector metric:
        Minimum bandwidth is 256 Kbit
        Total delay is 200480 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1

Rack1R1#show ip route 164.1.26.0
Routing entry for 164.1.26.0/24
  Known via "eigrp 100", distance 90, metric 3026432, type internal
  Redistributing via eigrp 100
  Last update from 164.1.13.3 on Serial0/1, 00:02:05 ago
  Routing Descriptor Blocks:
  * 164.1.12.2, from 164.1.12.2, 00:02:05 ago, via Serial0/0
      Route metric is 15132160, traffic share count is 1
      Total delay is 200480 microseconds, minimum bandwidth is 256 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
    164.1.13.3, from 164.1.13.3, 00:02:05 ago, via Serial0/1
      Route metric is 3026432, traffic share count is 5
      Total delay is 40100 microseconds, minimum bandwidth is 1280 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
```


Task 2.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

OSPF > OSPF Network Point-to-Point

R3:
```
router ospf 1
 router-id 150.1.3.3
 network 164.1.34.3 0.0.0.0 area 0
 network 164.1.35.3 0.0.0.0 area 0
```

R4:
```
interface Serial0/0/0
 ip ospf network point-to-point
!
router ospf 1
 router-id 150.1.4.4
 network 164.1.34.4 0.0.0.0 area 0
```

R5:
```
interface Serial0/0
 ip ospf network point-to-point
!
router ospf 1
 router-id 150.1.5.5
 network 164.1.5.5 0.0.0.0 area 0
 network 164.1.35.5 0.0.0.0 area 0
 network 164.1.55.5 0.0.0.0 area 0
```

Task 2.2 Breakdown

The task asks for configuring OSPF area 0 and advertising some networks in there. The wording points out you should be using the same area for the new prefixes.

Task 2.2 Verification

Check OSPF neighbors

```
Rack1R3#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         0   FULL/  -        00:00:38    164.1.35.5      Serial1/0.35
150.1.4.4         0   FULL/  -        00:00:35    164.1.34.4      Serial1/0.34
```

Verify OSPF routes:

```
Rack1R3#show ip route ospf
     164.1.0.0/16 is variably subnetted, 20 subnets, 3 masks
O IA    164.1.32.0/24 [110/784] via 164.1.34.4, 00:01:23, Serial1/0.34
O IA    164.1.45.0/29 [110/782] via 164.1.34.4, 00:29:51, Serial1/0.34
O IA    164.1.47.0/24 [110/782] via 164.1.34.4, 00:29:51, Serial1/0.34
O IA    164.1.43.0/24 [110/784] via 164.1.34.4, 00:01:23, Serial1/0.34
O       164.1.55.0/24 [110/782] via 164.1.35.5, 00:29:51, Serial1/0.35
O       164.1.5.0/24 [110/782] via 164.1.35.5, 00:29:51, Serial1/0.35
O IA    164.1.7.0/24 [110/783] via 164.1.34.4, 00:01:23, Serial1/0.34
O IA    164.1.14.0/24 [110/783] via 164.1.34.4, 00:01:23, Serial1/0.34
O IA    164.1.9.0/24 [110/784] via 164.1.34.4, 00:01:23, Serial1/0.34
O IA    164.1.31.0/24 [110/783] via 164.1.34.4, 00:01:23, Serial1/0.34
O IA    164.1.24.0/24 [110/784] via 164.1.34.4, 00:01:23, Serial1/0.34
     150.1.0.0/16 is variably subnetted, 10 subnets, 2 masks
O IA    150.1.5.0/24 [110/782] via 164.1.35.5, 00:29:51, Serial1/0.35
O IA    150.1.4.0/24 [110/782] via 164.1.34.4, 00:29:52, Serial1/0.34
O IA    150.1.10.10/32 [110/784] via 164.1.34.4, 00:01:24, Serial1/0.34
O IA    150.1.9.9/32 [110/784] via 164.1.34.4, 00:01:24, Serial1/0.34
O IA    150.1.7.7/32 [110/783] via 164.1.34.4, 00:01:24, Serial1/0.34
```


Task 2.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

EIGRP > EIGRP Network Statement
RIP > Basic RIP Configuration

R6:

```
!
! Redistribution metrics are arbitrary
!
router rip
 version 2
 no auto-summary
 network 54.0.0.0
 redistribute eigrp 100 metric 1
!
router eigrp 100
 redistribute rip metric 10000 1000 1 255 1500
```

Task 2.3 Verification

Verify the RIP routes received from BB1:

```
Rack1R6#show ip route rip
R    212.18.1.0/24 [120/1] via 54.1.2.254, 00:00:01, Serial0/0/0
R    212.18.0.0/24 [120/1] via 54.1.2.254, 00:00:01, Serial0/0/0
R    212.18.3.0/24 [120/1] via 54.1.2.254, 00:00:01, Serial0/0/0
R    212.18.2.0/24 [120/1] via 54.1.2.254, 00:00:01, Serial0/0/0
```

Verify redistribution:

```
Rack1R1#show ip route eigrp | include D EX
D EX    54.1.2.0 [170/15388160] via 164.1.12.2, 00:00:40, Serial0/0
D EX 212.18.1.0/24 [170/15388160] via 164.1.12.2, 00:00:40, Serial0/0
D EX 212.18.0.0/24 [170/15388160] via 164.1.12.2, 00:00:40, Serial0/0
D EX 212.18.3.0/24 [170/15388160] via 164.1.12.2, 00:00:40, Serial0/0
D EX 212.18.2.0/24 [170/15388160] via 164.1.12.2, 00:00:40, Serial0/0

Rack1R6#show ip rip database | include redistributed
150.1.1.0/24    redistributed
150.1.2.0/24    redistributed
150.1.3.0/24    redistributed
150.1.4.0/23    redistributed
<snip>
```


Task 2.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

EIGRP > EIGRP Summarization

R3:

```
interface Serial1/1.23
 ip summary-address eigrp 100 150.1.4.0 255.255.254.0
!
interface Serial1/2
 ip summary-address eigrp 100 150.1.4.0 255.255.254.0
!
router ospf 1
 redistribute eigrp 100 subnets metric 10
!
router eigrp 100
 redistribute ospf 1 metric 10000 1000 1 255 1500
```

SW2:

```
interface FastEthernet0/15
 ip summary-address eigrp 100 150.1.4.0 255.255.254.0 5
!
router ospf 1
 redistribute eigrp 100 subnets
!
router eigrp 100
 redistribute ospf 1 metric 10000 1000 1 255 1500
```


Task 2.4 Verification

Verify loopback networks summarization:

```
Rack1R1#show ip route 150.1.4.0
Routing entry for 150.1.4.0/23
  Known via "eigrp 100", distance 90, metric 514560, type internal
  Redistributing via eigrp 100
  Last update from 164.1.13.3 on Serial0/1, 00:00:29 ago
  Routing Descriptor Blocks:
  * 164.1.18.8, from 164.1.18.8, 00:00:29 ago, via FastEthernet0/0
      Route metric is 514560, traffic share count is 80
     Total delay is 10100 microseconds, minimum bandwidth is 10000 Kbit
      Reliability 1/255, minimum MTU 1500 bytes
      Loading 255/255, Hops 1
    164.1.13.3, from 164.1.13.3, 00:00:29 ago, via Serial0/1
      Route metric is 2434560, traffic share count is 17
     Total delay is 30000 microseconds, minimum bandwidth is 1536 Kbit
      Reliability 1/255, minimum MTU 1500 bytes
      Loading 255/255, Hops 1
```


Task 2.5 Solution

🕮 Before You Begin

This scenario deals with simple, two-point redistribution. If you haven't done so already it is recommended to read the following post on the INE blog to get better understanding of redistribution: "Understanding Redistribution Part I, II and III":

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

http://blog.ine.com/2008/02/19/understanding-redistribution-part-ii/

Additionally, we suggest completing VOL2 Lab 2 prior to this scenario, to get familiar with the routing loop analysis process. Notice that Lab 2's redistribution is more complicated than this particular scenario.

R3:

```
router eigrp 100
 redistribute ospf 1 metric 10000 1000 1 255 1500 route-map OSPF->EIGRP
!
router ospf 1
 redistribute eigrp 100 subnets tag 390 metric 10
!
route-map OSPF->EIGRP deny 10
 match tag 890
!
route-map OSPF->EIGRP permit 20
```

SW2:

```
router eigrp 100
 redistribute ospf 1 metric 10000 1000 1 255 1500 route-map OSPF->EIGRP
!
router ospf 1
 redistribute eigrp 100 subnets tag 890
!
route-map OSPF->EIGRP deny 10
 match tag 390
!
route-map OSPF->EIGRP permit 20
```


Task 2.5 Breakdown

Commonly, with route redistribution there is more than one possible solution to resolve most issues. In this task, route tags were used to ensure that any new routes redistributed into EIGRP on R6 will not be passed back into EIGRP from OSPF on R3 or SW2. You may notice that suboptimal routing may occur on R3 or SW2 to reach the routes redistributed on R6, but unless specifically asked for in the task, suboptimal routing is not necessarily an issue that needs to be resolved. Remember that the lab is just looking for reachability and not "optimal reachability".

Pitfall

Note that the EIGRP to OSPF redistribution metric is lower on R3 than it is on SW2. R3's Loopback0 network is advertised as an internal route into EIGRP only, which means the OSPF domain will have to choose between R3 and SW2 as the exit points to reach the external route. If the prefix is routed out via SW2, reachability problems will occur once the BGP section is complete.

Although this configuration technically doesn't relate directly to the points associated with the redistribution section, remember that stable IGP reachability is always a requirement of a fully functional BGP network.

Task 2.5 Verification

Check the routing tables on R3 and SW2:

```
Rack1R3#show ip route | i 212
D EX 212.18.1.0/24 [170/2770432] via 164.1.23.2, 00:24:59, Serial1/1.23
D EX 212.18.0.0/24 [170/2770432] via 164.1.23.2, 00:24:59, Serial1/1.23
D EX 212.18.3.0/24 [170/2770432] via 164.1.23.2, 00:24:59, Serial1/1.23
D EX 212.18.2.0/24 [170/2770432] via 164.1.23.2, 00:24:59, Serial1/1.23

Rack1SW2#show ip route | i 212
O E2 212.18.1.0/24 [110/10] via 164.1.32.9, 00:13:06, Port-channel23
O E2 212.18.0.0/24 [110/10] via 164.1.32.9, 00:13:06, Port-channel23
O E2 212.18.3.0/24 [110/10] via 164.1.32.9, 00:13:06, Port-channel23
O E2 212.18.2.0/24 [110/10] via 164.1.32.9, 00:13:06, Port-channel23

Rack1SW2#show ip eigrp topology 212.18.1.0 255.255.255.0
EIGRP-IPv4:(100) (AS 100): Topology Default-IP-Routing-Table(0) entry
for 212.18.1.0/24
  State is Passive, Query origin flag is 1, 0 Successor(s), FD is
4294967295
  Descriptor Blocks:
  164.1.18.1 (FastEthernet0/15), from 164.1.18.1, Send flag is 0x0
      Composite metric is (3284992/3282432), Route is External
      Vector metric:
        Minimum bandwidth is 1280 Kbit
        Total delay is 50200 microseconds
        Reliability is 1/255
        Load is 255/255
        Minimum MTU is 1500
        Hop count is 4
      External data:
        Originating router is 150.1.6.6
        AS number of route is 0
        External protocol is RIP, external metric is 1
        Administrator tag is 0 (0x00000000)
```


Task 2.6 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

OSPF > OSPF Conditional Default Routing

```
R3:
router ospf 1
 default-information originate route-map CONDITIONAL_DEFAULT
!
ip prefix-list R1_or_R2 seq 5 permit 164.1.13.0/24
ip prefix-list R1_or_R2 seq 10 permit 164.1.23.0/24
!
route-map CONDITIONAL_DEFAULT permit 10
 match ip address prefix-list R1_or_R2
 match metric 1 +- 1
```

Task 2.6 Breakdown

OSPF conditional default routing uses a route-map attached to the **default-information originate** command. The route-map verifies existence of IP prefixes in the routing table. Based on this, the route-map can only match the attributes present in the routing table, such as IP prefix, next-hop, metric, but for example not on BGP attributes. On R3 we have the following:

- If Serial1/2 port goes down on R3, so does the Serial port 0/1 on R1.
- If the Serial1/1.23 interface goes down R2's Serial0/0.23 interface will remain in the up state, as long as the corresponding PVC is active.

Therefore, the loss of connection on R3 will result in the loss of the local connected prefix. However, the same prefix may leak back to R3 by means of redistribution on R2. Based on this, we need to originated conditional default route only if the matched prefix is directly connected.We accomplish this using the criterion **match metric 1 +- 1**, which filters the possibility of matching on non-connected route.

🕮 Deep Dive

- What is the best practice of originating a conditional default route?

Task 2.6 Verification

Check default route, when both R3's EIGRP-enabled links are up:

```
Rack1R5#show ip route ospf | include 0.0.0.0
O*E2 0.0.0.0/0 [110/1] via 164.1.35.3, 00:00:24, Serial0/0
```

Shutdown both of the EIGRP enabled links at R3 and observe the output from the debug:

```
Rack1R3#debug ip ospf lsa-generation
OSPF summary lsa generation debugging is on
Rack1R3#conf t
Rack1R3(config)#interface s1/1.23
Rack1R3(config-subif)#shutdown
Rack1R3(config)#interface s1/2
Rack1R3(config-if)#shutdown

OSPF: Generate external LSA 0.0.0.0, mask 0.0.0.0, type 5, age 3600,
metric 16777215, tag 1, metric-type 2, seq 0x80000002
OSPF: 0.0.0.0/0 type: 5 is already maxaged
```

Verify that OSPF domain lost default route:

```
Rack1R5#show ip route ospf | include 0.0.0.0
Rack1R5#

Rack1R3(config)#interface s1/1.23
Rack1R3(config-subif)#no shutdown
Rack1R3(config)#interface s1/2
Rack1R3(config-if)#no shutdown
```


Use the TCL script below to test reachability:

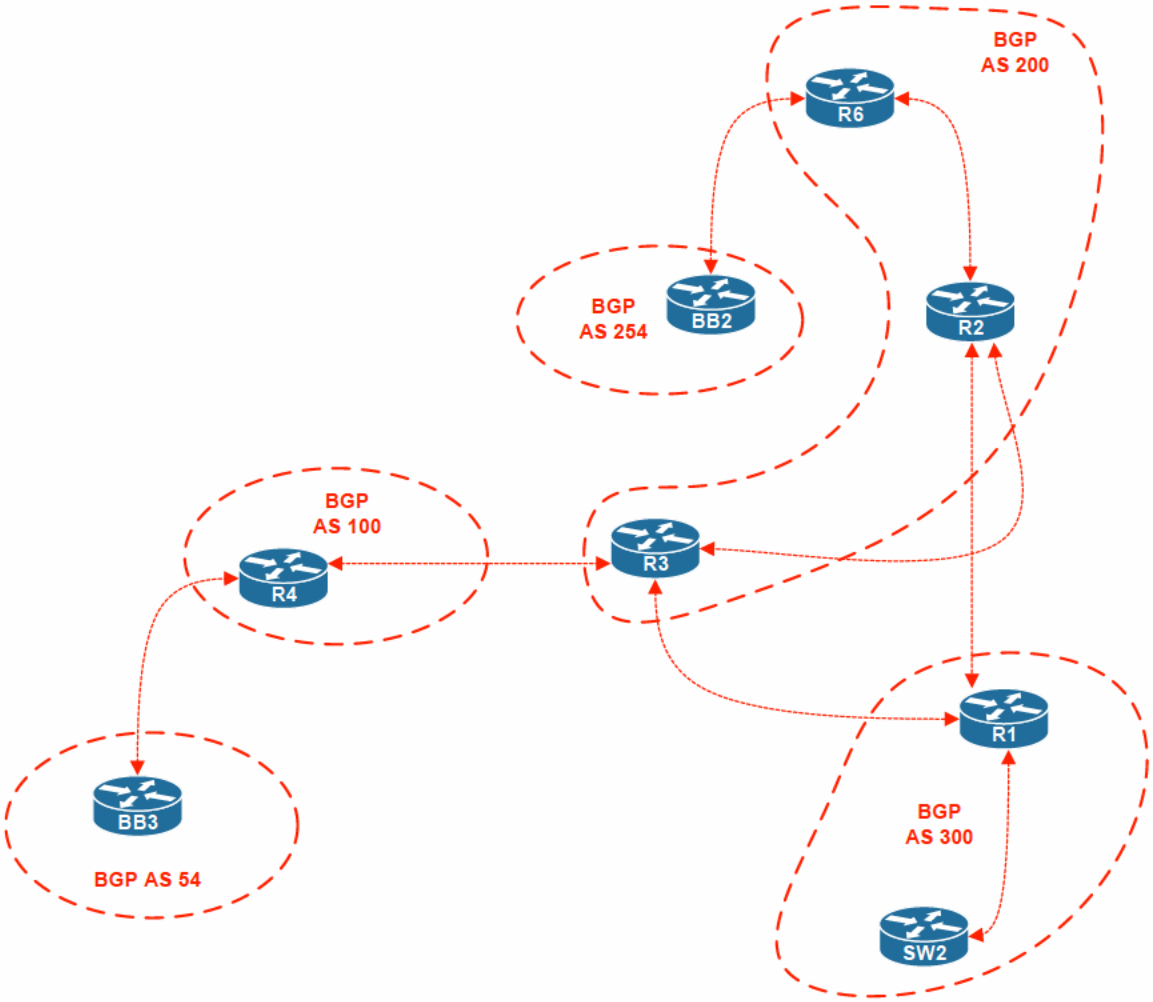
```
foreach i {
150.1.1.1
164.1.12.1
164.1.13.1
164.1.18.1
150.1.2.2
164.1.12.2
164.1.23.2
164.1.26.2
164.1.34.3
164.1.35.3
150.1.3.3
164.1.13.3
164.1.23.3
164.1.34.4
164.1.45.4
164.1.47.4
150.1.4.4
164.1.35.5
164.1.45.5
164.1.55.5
150.1.5.5
164.1.5.5
54.1.2.6
150.1.6.6
164.1.26.6
164.1.47.7
150.1.7.7
164.1.7.7
164.1.14.7
164.1.31.7
150.1.8.8
164.1.24.8
164.1.32.8
164.1.18.8
164.1.43.9
150.1.9.9
164.1.31.9
164.1.32.9
164.1.43.10
150.1.10.10
164.1.14.10
164.1.24.10

} { ping $i  }
```

Note that VLAN43, VLAN62, and VLAN3 are not a part of any IGP and are not tested for reachability.

Task 2.7 Solution

🕮 Before You Begin

Make a BGP diagam for this scenario, discovering existing BGP peering sessions. Watch out for possible misconfigurations in process (e.g. peering sessions failure, RR misconfigurations, existing filters):



This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Aggregation

```
R3:
!
! Advertise a network for aggregation
!
router bgp 200
 network 164.1.3.0 mask 255.255.255.0

R4:
router bgp 100
 aggregate-address 164.1.0.0 255.255.0.0 summary-only

R6:
router bgp 200
 aggregate-address 164.1.0.0 255.255.0.0 summary-only
```

Task 2.7 Verification

Verify the summary generation. For instance on R6:

```
Rack1R6#show ip bgp | include 164|Net
   Network          Next Hop            Metric LocPrf Weight Path
*> 164.1.0.0        0.0.0.0                            32768 i
s>i164.1.3.0/24     164.1.23.3               0    100      0 i
```


Task 2.8 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Default Routing

R1:
```
router bgp 300
 neighbor 164.1.18.8 default-originate
 neighbor 164.1.18.8 prefix-list DEFAULT out
!
ip prefix-list DEFAULT seq 5 permit 0.0.0.0/0
```

SW2:
```
router ospf 1
 distribute-list PREFER_DEFAULT_VIA_BGP in
!
ip access-list standard PREFER_DEFAULT_VIA_BGP
 deny   0.0.0.0
 permit any
```

Task 2.8 Breakdown

Recall from the previous OSPF configuration that R3 originated a default route into OSPF. When SW2 learns the iBGP default route from R1, and the OSPF External default route from SW3 and SW4, the OSPF route with the lower administrative distance is preferred. To ensure that the BGP route is installed, a distribute-list is applied to the routing table on SW2 to stop the OSPF default route from being used. The result of this is that the iBGP learned default route from R1 makes it into the table.

Task 2.8 Verification

Verify that R1 advertises the default route to SW2

```
Rack1R1#show ip bgp neighbors 164.1.18.8 advertised-routes
BGP table version is 36, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

Originating default network 0.0.0.0

   Network          Next Hop            Metric LocPrf Weight Path

Total number of prefixes 0
```

Verify BGP routes on SW2:

```
Rack1SW2#show ip bgp | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*>i0.0.0.0          164.1.18.1               0    100      0 i

Rack1SW2#show ip route bgp
B*    0.0.0.0/0 [200/0] via 164.1.18.1, 00:01:53
```

Task 2.9 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Regular Expressions

R2:

```
ip as-path access-list 1 permit ^$
!
router bgp 200
 neighbor 164.1.12.1 filter-list 1 out
```

R3:

```
ip as-path access-list 1 permit ^$
!
router bgp 200
 neighbor 164.1.13.1 filter-list 1 out
```


Task 2.9 Breakdown

The above task states that AS 200 cannot be used as transit for users in AS 300. Therefore by only advertising prefixes that were originated inside AS 200, AS 300 cannot use AS 200 to reach any other ASs. In the above solution this is accomplished through the usage of filtering based on AS-Path information.

Since the AS-Path of a prefix is not added until the prefix leaves the AS, prefixes which have been originated within the AS will have an empty AS-Path. This can be easily matched with a regular expression which specifies that the end of the line comes immediately after the end of the line, and is denoted as ^$.

Task 2.9 Verification

Verify the routes that R2 and R3 advertise to AS 300:

```
Rack1R2#show ip bgp neighbors 164.1.12.1 advertised-routes
BGP table version is 17, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i164.1.0.0        164.1.26.6               0    100      0 i
*>i164.1.3.0/24     164.1.23.3               0    100      0 i

Total number of prefixes 2

Rack1R3#show ip bgp neighbors 164.1.13.1 advertised-routes
BGP table version is 17, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i164.1.0.0        164.1.26.6               0    100      0 i
*> 164.1.3.0/24     0.0.0.0                  0         32768 i

Total number of prefixes 2
```


Task 2.10 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

BGP > BGP Regular Expressions

R4:

```
router ospf 1
 redistribute bgp 100 subnets route-map BGP2OSPF
!
ip as-path access-list 1 permit ^54_
!
route-map BGP2OSPF permit 10
 match as-path 1
```

Task 2.10 Breakdown

The above task describes a case where reachability is lost to certain BGP networks when the primary Frame Relay connection of R4 is down. When the Frame Relay connection is down, all of R4's traffic destined to R3 must transit R5. The problem, however, is that R5 does not participate in BGP routing. Therefore, although BGP network layer reachability information is successfully transmitted throughout the network, traffic may be black holed when it reaches R5.

In order to resolve this issue, BGP has been redistributed into IGP. R4 has been configured to redistribute all BGP information learned from AS 54 into OSPF. For traffic in the opposite direction, it doesn't matter, since R3 is originating a default route.

🕮 Deep Dive

- What is the default metric for BGP routes redistributed into OSPF?
- Why can't you safely redistribute IGP routes into iBGP and back into IGP?

Task 2.10 Verification

Before applying the solution, verify reachability to AS54's prefixes when R4's Frame Relay link is up:

```
Rack1R3#traceroute 28.119.16.1

Type escape sequence to abort.
Tracing the route to 28.119.16.0

  1 164.1.34.4 28 msec 32 msec 28 msec
  2 204.12.1.254 32 msec 28 msec 32 msec
```

Now shutdown R4's Serial0/0/0 interface and repeat the traceroute: traffic is looped between R3 and R5:

```
Rack1R3#trace 28.119.16.1

Type escape sequence to abort.
Tracing the route to 28.119.16.0

  1 164.1.35.5 32 msec 28 msec 28 msec
  2 164.1.35.3 72 msec 57 msec 60 msec
```

Try traceroute after redistribution has been applied:

```
Rack1R3#trace 28.119.16.1

Type escape sequence to abort.
Tracing the route to 28.119.16.0

  1 164.1.35.5 32 msec 28 msec 32 msec
  2 164.1.45.4 28 msec 28 msec 32 msec
  3 204.12.1.254 32 msec 32 msec 32 msec
```

Verify the OSPF routes on R5:

```
Rack1R5#show ip route ospf | i \.4,
<output omitted>
O E2 112.0.0.0/8 [110/1] via 164.1.45.4, 00:04:27, Serial0/1/0
O E2    28.119.17.0 [110/1] via 164.1.45.4, 00:04:27, Serial0/1/0
O E2    28.119.16.0 [110/1] via 164.1.45.4, 00:04:27, Serial0/1/0
<output omitted>
```


Task 3.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IPv6 > IPv6 Link-Local Addressing
IPv6 > IPv6 Global Aggregatable Addressing

R1:
```
ipv6 unicast-routing
!
interface Serial0/1
 ipv6 address 2001:164:1:13::1/64
!
interface Serial0/0
 ipv6 address 2001:164:1:12::1/64
 frame-relay map ipv6 2001:164:1:12::2 102 broadcast
```

R2:
```
ipv6 unicast-routing
!
interface Serial0/0.12 point-to-point
 ipv6 address 2001:164:1:12::2/64
!
interface Serial0/0.23 point-to-point
 ipv6 address 2001:164:1:23::2/64
```

R3:
```
ipv6 unicast-routing
!
interface Serial1/2
 ipv6 address 2001:164:1:13::3/64
!
interface Serial1/1.23 point-to-point
 ipv6 address 2001:164:1:23::3/64
```


Task 3.1 Verification

Check IPv6 addressing

```
Rack1R2#show ipv6 interface brief
FastEthernet0/0            [up/up]
Serial0/0                  [up/up]
Serial0/0.12               [up/up]
    FE80::20D:BCFF:FE16:2720
    2001:164:1:12::2
Serial0/0.23               [up/up]
    FE80::20D:BCFF:FE16:2720
    2001:164:1:23::2
Serial0/1                  [administratively down/down]
Loopback0                  [up/up]

Rack1R2#ping 2001:164:1:12::1
```

Check IPv6 connectivity

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:164:1:12::1, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/60 ms

Rack1R2#ping 2001:164:1:23::3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:164:1:23::3, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms

Rack1R1#ping 2001:164:1:13::3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:164:1:13::3, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
```


Task 3.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IPv6 > IPv6 Link-Local Addressing
IPv6 > IPv6 OSPFv3

R1:

```
!
! The default OSPF network type is NBMA
!
interface Serial0/0
 ipv6 ospf 1 area 123
 ipv6 ospf network point-to-point
 frame map ipv6 fe80::2 102
!
! Adjust cost to ensure path manipulation
!
interface Serial0/1
 ipv6 ospf 1 area 123
 ipv6 ospf cost 455
```

R2:

```
interface Serial0/0.12 point-to-point
 ipv6 ospf 1 area 123
 ipv6 address fe80::2 link-local
!
interface Serial0/0.23
 ipv6 ospf 1 area 123
```

R3:

```
interface Loopback100
 ipv6 address 2001:150:1:3::3/64
 ipv6 ospf 1 area 123
!
interface Serial1/1.23 point-to-point
 ipv6 ospf 1 area 123
!
interface Serial1/2
 ipv6 ospf 1 area 123
```


Task 3.2 Breakdown

There are couple of things to watch for here. There will be a network type mismatch between R2 and R1, and the metrics need to be manipulated so that R1 prefers the connection to R3's loopback via R2. Since R1 will have the network learned from R2 resolved in the routing table "via" the link-local address of R2, make sure that you also have a frame map for that address. Here, the link-local address is manually configured for simplicity.

🕮 Deep Dive

- Can OSPFv3 be used to route IPv4 traffic?

Task 3.2 Verification

Check OSPFv3 neigbhors

```
Rack1R2#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
150.1.3.3       1     FULL/  -        00:00:34    19              Serial0/0.23
150.1.1.1       1     FULL/  -        00:00:33    5               Serial0/0.12
```

```
Rack1R1#show ipv6 route ospf
IPv6 Routing Table - 8 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:150:1:3::3/128 [110/454]
     via FE80::2, Serial0/0
O   2001:164:1:23::/64 [110/454]
     via FE80::2, Serial0/0
Rack1R1#
```

Check the preferred path to R3's Loopback prefix

```
Rack1R1#traceroute 2001:150:1:3::3

Type escape sequence to abort.
Tracing the route to 2001:150:1:3::3

  1 2001:164:1:12::2 40 msec 40 msec 44 msec
  2 2001:150:1:3::3 48 msec 88 msec 49 msec
```


Task 4.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

MPLS VPN > VRF Lite

SW1, SW2, SW3, SW4:
```
vlan 99
```

SW3:
```
interface vlan 99
 ip address 99.99.99.3 255.255.255.0
```

SW4:
```
interface vlan 99
 ip address 99.99.99.4 255.255.255.0
```

SW3, SW4:
```
sdm prefer extended-match
```

SW3:
```
ip vrf SW3TEST
 rd 99:3
!
interface vlan 99
 ip vrf forwarding SW3TEST
 ip address 99.99.99.3 255.255.255.0
```

SW4:
```
ip vrf SW4TEST
 rd 99:4
!
interface vlan 99
 ip vrf forwarding SW4TEST
 ip address 99.99.99.4 255.255.255.0
```


Test connectivity within the VRF.

```
Rack1SW4#ping vrf SW4TEST 99.99.99.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 99.99.99.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms

Rack1SW4#ping vrf SW4TEST 99.99.99.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 99.99.99.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Rack1SW4#
```

Task 5.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Multicast > PIM Sparse-Dense Mode
Multicast > PIM Sparse-Mode

R3:

```
interface Loopback0
 ip pim sparse-dense-mode
```

R2, R3, R4, and SW1:

```
ip pim rp-address 150.1.3.3 3
!
access-list 3 permit 225.10.0.0 0.48.255.255
```


Task 5.1 Breakdown

To find the minimum amount of statements to match these groups first examine the groups:

```
225.10.0.0 - 225.10.255.255
225.26.0.0 - 225.26.255.255
225.42.0.0 - 225.42.255.255
225.58.0.0 - 255.58.255.255
```

From this output it is evident that the first octet will always be 225, and the third and fourth octets can be anything. Next write out the second octet in binary for comparison:

```
10 = 00001010
26 = 00011010
42 = 00101010
58 = 00111010
```

These four networks differ in only the 3rd and 4th most significant bits, and can be matched with a wildcard mask as follows:

```
      10 = 00001010
      26 = 00011010
      42 = 00101010
      58 = 00111010
Wildcard = 00110000
```

Resulting in: access-list 3 permit 225.10.0.0 0.48.255.255

Task 5.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Multicast > PIM Sparse-Dense Mode
Multicast > PIM Sparse-Mode

R4:

```
interface Loopback0
 ip pim sparse-dense-mode
```

R2, R3, R4, and SW1:

```
ip pim rp-address 150.1.4.4 4
!
access-list 4 permit 226.37.0.0 1.8.255.255
```

Task 5.2 Breakdown

To find the minimum amount of statements to match these groups first examine the groups:

```
226.37.0.0 - 226.37.255.255
226.45.0.0 - 226.45.255.255
227.37.0.0 - 227.37.255.255
227.45.0.0 - 227.45.255.255
```

From this output it is evident that the third and fourth octets can be anything. Next write out the first and second octets in binary for comparison:

```
226 = 11100010
227 = 11100011

 37 = 00100101
 45 = 00101101
```

These bit patterns result in four combinations which can be matched as follows:

```
      226 = 11100010
      227 = 11100011
 Wildcard = 00000001

       37 = 00100101
       45 = 00101101
 Wildcard = 00001000

   226.37 = 11100010.00100101
   226.45 = 11100011.00101101
   227.37 = 11100010.00100101
   227.45 = 11100011.00101101
Wildcard = 00000001.00001000
```

Resulting in: `access-list 4 permit 226.37.0.0 1.8.255.255`

Tasks 5.1 – 5.2 Verification

Verify static RP configuration, for instance on R2:

```
Rack1R2#show ip pim rp mapping
PIM Group-to-RP Mappings

Acl: 3, Static
    RP: 150.1.3.3 (?)
Acl: 4, Static
    RP: 150.1.4.4 (?)

Rack1R2#show ip access-lists 3
Standard IP access list 3
    10 permit 225.10.0.0, wildcard bits 0.48.255.255

Rack1R2#show ip access-lists 4
Standard IP access list 4
    10 permit 226.37.0.0, wildcard bits 1.8.255.255
```

Task 5.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Multicast > IGMP Timers
Multicast > Multicast Boundary

```
R3:
interface FastEthernet0/1
 ip multicast boundary 1
 ip igmp query-max-response-time 3
 ip igmp query-interval 5
!
access-list 1 deny   226.37.1.1
access-list 1 permit any
```


Task 5.3 Breakdown

The task requires tuning IGMP timers and setting up multicast traffic filter. The wording points towards changing the periodic IGMP polling and query maximum response timer. The multicast filtering is said to be done without applying an access-group to the interface. An optimum solution would be using filter that blocks both control and data plane traffic. The multicast boundary blocks the data plane traffic for denied groups and also inspects IGMP/PIM joins for the denied groups. This allows for effective flooding reduction for the groups that should never be joined by a given segment.

🕮 Deep Dive

- What routing protocol was using IGMP as its transport?

Task 5.3 Verification

Verify IGMP configuration at R3:

```
Rack1R3#show ip igmp interface fa0/1
FastEthernet0/1 is up, line protocol is up
  Internet address is 164.1.3.3/24
  IGMP is enabled on interface
  Current IGMP host version is 2
  Current IGMP router version is 2
  IGMP query interval is 5 seconds
  IGMP querier timeout is 10 seconds
  IGMP max query response time is 3 seconds
  Last member query count is 2
  Last member query response interval is 1000 ms
  Inbound IGMP access group is not set
  IGMP activity: 1 joins, 0 leaves
  Multicast routing is enabled on interface
  Multicast TTL threshold is 0
  Multicast designated router (DR) is 164.1.3.3 (this system)
  IGMP querying router is 164.1.3.3 (this system)
  No multicast groups joined by this system
```


Verify the multicast boundary configuration:

```
Rack1R3#show ip multicast interface fa0/1
FastEthernet0/1 is up, line protocol is up
  Internet address is 164.1.3.3/24
  Multicast routing: enabled
  Multicast switching: fast
  Multicast packets in/out: 0/0
  Multicast boundary: 1  (in/out)
  Multicast TTL threshold: 0
  Multicast Tagswitching: disabled
```

Task 6.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Access Control with Dynamic Access Lists

SW1:

```
username RDP password 0 CISCO
!
interface Vlan41
 ip access-group REMOTE_DESKTOP in
!
interface FastEthernet0/13
 ip access-group REMOTE_DESKTOP in
!
interface Port-channel14
 ip access-group REMOTE_DESKTOP in
!
ip access-list extended REMOTE_DESKTOP
 dynamic RDP timeout 10 permit tcp any host 164.1.7.100 eq 3389
 deny   tcp any host 164.1.7.100 eq 3389
 permit ip any any
!
! Command activated by line login
!
line vty 0 15
 login local
 autocommand access-enable host
```


Task 6.1 Breakdown

This type of access-list configuration is known as a lock-and-key, or a dynamic access-list. When the access-list is applied, the dynamic entry does not yet exist in the list. This is similar to how an entry can be inactive when referencing a time range. When the command **access-enable** is executed, all dynamic entries are inserted into the access-list.

The command **autocommand access-enable** means that when a user logs in via the VTY line, the command **access-enable** will automatically execute. This is simply a way to automate the running of the command. The **autocommand access-enable** command can also be placed in the local user database on a per user basis. In the above case the autocommand applies to anyone entering the VTY lines. Notice that this will disallow normal logins on these VTY lines at the same time. This issue is addressed in the next task.

The **host** option of the access-enable statement dictates that only the host that authenticated will be allowed access through the dynamic statement. This is accomplished by dynamically creating a copy of the configured dynamic entry, or entries, with the source IP address as the authenticated address. The timeout value of 10 minutes applied to the access-list is an absolute timeout, and the user will have to re-authenticate once this timer expires.

🕮 Deep Dive

- How authentication proxy is different from dynamic access-lists?

Task 6.1 Verification

List the access-list contents:

```
Rack1SW1#show ip access-lists REMOTE_DESKTOP
Extended IP access list REMOTE_DESKTOP
    10 Dynamic RDP permit tcp any host 164.1.7.100 eq 3389
    20 deny tcp any host 164.1.7.100 eq 3389
    30 permit ip any any (8 matches)
Rack1SW1#
```

Telnet to SW1 and login with the user to activate the dynamic ACL entry

```
Rack1R4#telnet 164.1.47.7
Trying 164.1.47.7 ... Open

User Access Verification

Username: RDP
Password: CISCO
```

Check the dynamic ACL contents once again

```
Rack1SW1#show ip access-lists REMOTE_DESKTOP
Extended IP access list REMOTE_DESKTOP
    10 Dynamic RDP permit tcp any host 164.1.7.100 eq 3389
    10   permit tcp host 164.1.47.4 host 164.1.7.100 eq 3389
    20 deny tcp any host 164.1.7.100 eq 3389
    30 permit ip any any (31 matches)
```


Task 6.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Access Control with Dynamic Access Lists
System Management > Terminal Line Settings

SW1:

```
username NOC password 0 CISCO
!
access-list 100 permit tcp any any eq telnet
access-list 100 permit tcp any any eq 3023
!
line vty 0
 login local
 autocommand access-enable host
!
line vty 1 4
 no autocommand access-enable
 access-class 100 in
 rotary 23
```


Task 6.2 Breakdown

Since the command **autocommand access-enable** applies to all users starting an exec process through the VTY line, regular telnet access at port 23 is no longer available for the management on the CLI. In order to still allow users to be able to telnet into the router to manage it, the properties applied to the VTY lines have been split into two.

The first VTY line (VTY 0) is left with the **autocommand access-enable** command. All users that telnet to the router at port 23 will hit this line. The **rotary** command under the VTY line allows the router to listen for telnet sessions at higher port ranges (30xx, 50xx, 70xx, 100xx, where x is the configured rotary option), so users can still telnet in to access the CLI.

Task 6.2 Verification

Connect to the rotary group port number

```
Rack1SW1#telnet 150.1.7.7 3023
Trying 150.1.7.7, 3023 ... Open


User Access Verification

Username: NOC
Password: CISCO
Rack1SW1>
```


Task 6.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Router IP Traffic Export

R6:

```
!
! Define traffic filter
!
ip access-list extended FILTER
 permit udp 164.1.0.0 0.0.255.255 any eq 53
!
! Create export profile
!
ip traffic-export profile EXPORT
  interface FastEthernet0/1
  incoming access-list FILTER
  mac-address 1234.5678.9abc
  exit
!
! Apply export profile
!
interface FastEthernet 0/1
 ip traffic-export apply EXPORT
```

Task 6.3 Breakdown

Routers are commonly not designed to intercept and redirect traffic for the purpose of intrusion prevention or lawful intercept. However, in some cases using a dedicated device could be complicated, such as in the case of WAN connection. RITE allows for controlled traffic export to a specified, directly attached device. The capturing device must connect via Ethernet and exported traffic could not be routed.

🕮 Deep Dive

- What are the drawbacks of using SPAN/RSPAN compared to RITE?
- How is ERSPAN different from RSPAN?

Task 6.3 Verification

Check traffic export configuration

```
Rack1R6#show ip traffic-export
Router IP Traffic Export Parameters
Monitored Interface             FastEthernet0/1
        Export Interface                FastEthernet0/1
        Destination MAC address 1234.5678.9abc
        bi-directional traffic export is off
Input IP Traffic Export Information     Packets/Bytes Exported    0/0
        Packets Dropped           3
        Sampling Rate             one-in-every 1 packets
        Access List         FILTER [named extended IP]
        Profile EXPORT is Active
```


Task 6.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Flexible Packet Matching

R4:

```
Load protocol system:/fpm/phdf/ip.phdf
Load protocol system:/fpm/phdf/udp.phdf

class-map type stack match-all MYUDP
 match field ip prot eq 0x11 next udp

class-map type access-control match-all SLAM
 match field udp dest-port eq 0x59A
 match field ip length eq 0x194
 match start l3-start offset 224 size 4 eq 0x4011010

policy-map type access-control MYFPM
  class SLAM
   drop
!
policy-map type access-control MYINT
 class MYUDP
  service-policy MYFPM
interface Fa0/1
 service-policy type access-control input MYINT
```


Task 6.4 Breakdown

Flexible Packet Matching is a very powerful tool that could be used for deep packet inspection. Some people call it as “access-list on steroids”. Configuration can be a bit tricky, so it is recommended that you save before starting. Newer versions of IOS include the PHDF files in the system folder, so you may not need to copy the files to flash. After loading the PHDF files, the fields can be matched in the class maps. A nested policy is used, referencing the stack class map in the parent policy.

```
Rack1R4#show policy-map type access-con int fa0/1
 FastEthernet0/1
  Service-policy access-control input: MYINT
    Class-map: MYUDP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps
      Match: field IP protocol eq 0x11 next UDP

      Service-policy access-control : MYFPM

        Class-map: SLAM (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps
          Match: field UDP dest-port eq 0x59A
          Match: field IP length eq 0x194
          Match: start l3-start offset 224 size 4 eq 0x4011010
      drop

        Class-map: class-default (match-any)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: any
    Class-map: class-default (match-any)
      7 packets, 472 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
Rack1R4#
```


Task 6.5 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Zone Based Firewall

R6:

```
!
! Class matching TCP and UDP traffic
!
class-map type inspect match-any TCPUDP
 match protocol tcp
 match protocol udp
!
! ICMP packets
!
class-map type inspect match-any ICMP
 match protocol icmp
!
policy-map type inspect TEST
 class TCPUDP
  inspect
 class ICMP
  pass
!
policy-map type inspect ICMP
 class ICMP
  pass
!
zone security inside
zone security outside
!
zone-pair security INOUT source inside destination outside
 service-policy type inspect TEST
!
zone-pair security OUTIN source outside destination inside
 service-policy type inspect ICMP
!
interface Fa0/0
 zone-member security inside
!
interface Fa0/1
 zone-member security inside
!
interface Serial 0/0/0
 zone-member security outside
```


Task 6.5 Breakdown

Start with class maps to define the traffic, configure policy maps with actions for the traffic classes, and apply the policies to the zone pair. Configure the interfaces for the appropriate zones. Notice that is is possible to use access-lists for classification, but since we are only interested in protocols and not specific sources/destiations it is enough to simply match the protocols in class-maps.

🕮 Deep Dive

- What are the specifics of ZFW “self” zone?

Task 6.5 Verification

To generate some traffic, you can telnet to BB1 from R2 and issue “show ip route”. For ICMP traffic, you can ping from R2 to BB1.

```
Rack1R6#show policy-map type inspect zone-pair INOUT

policy exists on zp INOUT
 Zone-pair: INOUT

  Service-policy inspect : TEST

    Class-map: TCPUDP (match-any)
      Match: protocol tcp
        1 packets, 24 bytes
        30 second rate 0 bps
      Match: protocol udp
        0 packets, 0 bytes
        30 second rate 0 bps

   Inspect
        Packet inspection statistics [process switch:fast switch]
        tcp packets: [0:99]

        Session creations since subsystem startup or last reset 1
        Current session counts (estab/half-open/terminating) [1:0:0]
        Maxever session counts (estab/half-open/terminating) [1:1:0]
        Last session created 00:00:37
        Last statistic reset never
        Last session creation rate 1
        Maxever session creation rate 1
        Last half-open session total 0


    Class-map: ICMP (match-any)
      Match: protocol icmp
        584 packets, 46720 bytes
        30 second rate 8000 bps
      Pass
        584 packets, 46720 bytes

    Class-map: class-default (match-any)
      Match: any
      Drop
        0 packets, 0 bytes
```



```
Rack1R6#show policy-map type inspect zone-paiR INOUT session

policy exists on zp INOUT
 Zone-pair: INOUT

  Service-policy inspect : TEST

    Class-map: TCPUDP (match-any)
      Match: protocol tcp
        1 packets, 24 bytes
        30 second rate 0 bps
      Match: protocol udp
        0 packets, 0 bytes
        30 second rate 0 bps

   Inspect

      Number of Established Sessions = 1
      Established Sessions
        Session 4960EAA0 (164.1.26.2:17775)=>(54.1.2.254:23) tcp
SIS_OPEN
          Created 00:00:46, Last heard 00:00:25
          Bytes sent (initiator:responder) [52:9817]


    Class-map: ICMP (match-any)
      Match: protocol icmp
        0 packets, 0 bytes
        30 second rate 0 bps
      Pass
        0 packets, 0 bytes

    Class-map: class-default (match-any)
      Match: any
      Drop
        0 packets, 0 bytes
Rack1R6#
```


Task 6.6 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

Security > Role Based CLI

R5:
```
username CISCO secret CISCO
username INTERN secret INTERN
!
! RBAC requires AAA
!
aaa new-model
aaa authentication login default none
aaa authentication login VTY local
!
line vty 0 988
 login authent VTY
```

Telnet to R5, and log in with one of the user accounts. Start by switching to VIEW mode, with the command **enable view**. Note: This is a global command, not a configuration command.

R5:
```
enable VIEW
!
conf t
!
parser view INT
 secret cisco
 commands exec include show int
 commands exec include show clock
!
aaa authorization exec VTY local
!
! Apply views (roles) to the previously created users
!
username CISCO view root
username INTERN view INT
!
line vty 0 807
 authorization exec VTY
```


Task 6.6 Breakdown

Role based CLI allows you to be very granular for the access that you are giving people. Commands can be assigned to views, and then the users tied to the individual roles. You can also create superviews, which bundle the capability of multiple individual views. Role-based CLI does require AAA to be configured.

Task 6.6 Verification

Enter a view

```
Rack1R5#enable view INT
Rack1R5#?
Exec commands:
  enable  Turn on privileged commands
  exit    Exit from the EXEC
  show    Show running system information
```

Check commands available in the view

```
Rack1R5#show ?
  clock       Display the system clock
  flash:      display information about flash: file system
  interfaces  Interface status and configuration
  parser      Show parser commands

Rack1R5#show interfaces fastEthernet 0/0 | section Hard
  Hardware is Gt96k FE, address is 0013.1a68.b04c (bia 0013.1a68.b04c)

Rack1R5#conf t
       ^
% Invalid input detected at '^' marker.
```

Task 7.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

System Management > NTP

R4:

```
ntp master 2
!
interface FastEthernet0/0
 ntp broadcast
```

SW1:

```
interface Vlan41
 ntp broadcast client
```


Task 7.1 Breakdown

Since the scenario restricts the use of explicit NTP peering configuration, the only option is using NTP broadcasting on the segment shared by both routers.

Task 7.1 Verification

Confirm NTP synchronization

```
Rack1SW1#show ntp status
Clock is synchronized, stratum 3, reference is 164.1.47.4
nominal freq is 119.2092 Hz, actual freq is 119.2094 Hz, precision is
2**17
reference time is CE5C491B.7AEF095A (05:35:23.480 UTC Thu Sep 17 2009)
clock offset is -0.0261 msec, root delay is 1.53 msec
root dispersion is 16812.73 msec, peer dispersion is 15875.02 msec
```

Task 7.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IP Services > DHCP Client

R3:

```
interface FastEthernet0/0
 ip dhcp client hostname ROUTER3
 ip dhcp client lease 1 4 0
 ip address dhcp
```


Task 7.2 Breakdown

Make sure that you read the task carefully. The task only requires you configuring the interface for DHCP, not configuring a DHCP server.

Task 7.3 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

System Management > Kron Command Scheduler

R3:

```
kron occurrence TASK7.3-O in 3:0 recurring
 policy-list TASK7.3
!
kron policy-list TASK7.3
 cli renew dhcp FastEthernet 0/0
```


Task 7.3 Breakdown

The task asks for periodic state refreshes. Since you cannot control the lease duration on the client, the only option is ensuring the client re-requests the DHCP address every 3 hours. This could be done by manually issuing the DHCP client renew command, but in order to automate is a periodic scheduling facility should be used. Kron system is configured to accomplish this, issuing the exec “renew” command every 3 hours.

🕮 Deep Dive

- How could you manually shutdown and no-shutdown an interface every 3 hours?

Task 7.3 Verification

Kron will allow you to schedule a recurring task.

```
Rack1R3#show kron sched
Kron Occurrence Schedule
TASK7.3-O inactive, will run again in 0 days 02:59:29
```

Task 7.4 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

IP Services > IP Event Dampening

```
R5:
interface FastEthernet 0/0
 dampening 30 1000 2000 30 restart 2000
!
interface FastEthernet 0/1
 dampening 30 1000 2000 30 restart 2000
```


Task 7.4 Breakdown

For the dampening, a restart penalty of 2000 and reuse value of 1000 means that a half life period will bring the penalty from 2000 to 1000. Since the half life is 30, it will take 30 seconds. Reload the router and run the command **show interface dampening** a few times when the router first comes up.

```
Rack1R5#show int damp
FastEthernet0/0
  Flaps Penalty    Supp ReuseTm   HalfL  ReuseV   SuppV  MaxSTm    MaxP
Restart
      1    1289    TRUE       11       30    1000    2000       30    2000
2000
Rack1R5#

Rack1R5#show int damp
FastEthernet0/0
  Flaps Penalty    Supp ReuseTm   HalfL  ReuseV   SuppV  MaxSTm    MaxP
Restart
      1    1046    TRUE        2       30    1000    2000       30    2000
2000

%IFDAMP-5-UPDOWN: interface interface is %ssuppressed upda
te FastEthernet0/0 state to IP Routing, interface is UPsuppressed
Rack1R5#
```

🕮 Further Reading

"Optimizing IP Event Dampening"

http://blog.ine.com/2010/05/03/optimizing-ip-event-dampening/

Task 7.4 Verification

Verify dampening settings:

```
Rack1R5#show interfaces dampening
FastEthernet0/0
  Flaps Penalty    Supp ReuseTm   HalfL  ReuseV   SuppV  MaxSTm    MaxP Restart
     0       0   FALSE        0       30    1000    2000       30    2000    2000
FastEthernet0/1
  Flaps Penalty    Supp ReuseTm   HalfL  ReuseV   SuppV  MaxSTm    MaxP Restart
     0       0   FALSE        0       30    1000    2000       30    2000    2000
```


Task 8.1 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > Legacy Frame-Relay Traffic Shaping
QoS > Legacy FRTS with Per-VC Fragmentation

R3:
```
interface Serial1/0
 frame-relay traffic-shaping
!
interface Serial1/0.34
 frame-relay interface-dlci 304
  class DLCI_304
!
interface Serial1/0.35
 frame-relay interface-dlci 305
  class DLCI_305
!
map-class frame-relay DLCI_304
 frame-relay cir 256000
 frame-relay bc 2560
 frame-relay fragment 320
!
map-class frame-relay DLCI_305
 frame-relay cir 256000
 frame-relay bc 2560
 frame-relay fragment 320
```

R4:
```
interface Serial0/0/0
 frame-relay traffic-shaping
 frame-relay interface-dlci 403
  class DLCI_403
!
map-class frame-relay DLCI_403
 frame-relay cir 256000
 frame-relay bc 2560
 frame-relay fragment 320
```

R5:
```
interface Serial0/0
 frame-relay traffic-shaping
 frame-relay interface-dlci 503
  class DLCI_503
!
map-class frame-relay DLCI_503
 frame-relay cir 256000
 frame-relay bc 2560
 frame-relay fragment 320
```


Task 8.1 Breakdown

The task wording provides explcit CIR values and requires the shaping interval to be the lowest possible, which is 10ms. Furthermore, packet fragmentation is requested to ensure minimum latency for VoIP traffic. Since the interface speed equals to their CIR, both the Bc and fragment size are based on the same rate of 256000. Thus we come with the Tc of 2560 and fragment size of 320 bytes. Since the task explicity asks for legacy FRTS, we use the **`map-class frame-relay`** syntax.

🕮 Deep Dive

- Why fragmentation is not efficient without interleaving?
- What other ways to enable FRF.12 do you know?

Task 8.1 Verification

Verify Frame-Relay traffic-shaping configuration:

```
Rack1R3#show traffic-shape

Interface   Se1/0
       Access Target     Byte   Sustain   Excess    Interval  Increment Adapt
VC     List   Rate       Limit  bits/int  bits/int  (ms)      (bytes)   Active
302           56000      875    7000      0         125       875       -
301           56000      875    7000      0         125       875       -

Interface   Se1/0.34
       Access Target     Byte   Sustain   Excess    Interval  Increment Adapt
VC     List   Rate       Limit  bits/int  bits/int  (ms)      (bytes)   Active
304           256000     320    2560      0         10        320       -

Interface   Se1/0.35
       Access Target     Byte   Sustain   Excess    Interval  Increment Adapt
VC     List   Rate       Limit  bits/int  bits/int  (ms)      (bytes)   Active
305           256000     320    2560      0         10        320       -
```


Task 8.2 Solution

🕮 Before You Begin

This task assumes preliminary technology knowledge at least equivalent to the following VOL1 scenarios:

QoS > MQC Classification and Marking
QoS > MQC LLQ and Remaining Bandwidth Reservation
QoS > Using MQC CBWFQ with Legacy FRTS

R3:

```
!
! Configure LLQ policy
!
class-map match-all VoIP
  match access-group name VoIP
!
policy-map LLQ
  class VoIP
   priority 200
!
! Classify VoIP Traffic
!
ip access-list extended VoIP
 permit udp any any range 16384 32767
!
map-class frame-relay DLCI_304
 frame-relay mincir 256000
 service-policy output LLQ
```

R4:

```
class-map match-all VoIP
  match access-group name VoIP
!
policy-map LLQ
  class VoIP
   priority 200
!
ip access-list extended VoIP
 permit udp any any range 16384 32767
!
map-class frame-relay DLCI_403
 frame-relay mincir 256000
 service-policy output LLQ
```


Note

The MQC uses the **mincir** value in the Frame Relay map-class to determine the available bandwidth on a VC. Since MINCIR defaults to half of the configured CIR, it may be required to adjust the MINCIR value higher if the reserved bandwidth exceeds half of the configured CIR, regardless of whether adaptive shaping is enabled.

Task 8.2 Breakdown

The solution builds on the previous configuration. Since legacy FRTS has been chosen for traffic shaping, we don't have many options to provide priority voice treatment. It could be either LLQ or IP RTP Priority. Since the latter option is ruled out, we are only left with LLQ config. The only tricky part is keeping in mind that a CBWFQ policy associated with a map-class assumes the available bandwidth of mincir, which commonly reflects the CIR value in the SP cloud. Finally, the scenario explicitly tells to classify the VoIP bearer packets based on the fixed UDP port range. Therefore, we create an access-list to accomplish classification.

Deep Dive

- What is the reason for using the mincir as the bandwidth limit for MQC policy nested in a map-class?

Task 8.2 Verification

Verify the policy-map for LLQ configuration:

```
Rack1R3#show policy-map interface
 Serial1/0.34: DLCI 304 -

  Service-policy output: LLQ

    Class-map: VoIP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name VoIP
      Queueing
        Strict Priority
        Output Queue: Conversation 40
        Bandwidth 200 (kbps) Burst 5000 (Bytes)
        (pkts matched/bytes matched) 0/0
        (total drops/bytes drops) 0/0

    Class-map: class-default (match-any)
      1210 packets, 120910 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```



```
Rack1R4#show policy-map interface
 Serial0/0/0: DLCI 403 -

  Service-policy output: LLQ

    queue stats for all priority classes:

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

    Class-map: VoIP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name VoIP
      Priority: 200 kbps, burst bytes 5000, b/w exceed drops: 0


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

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


VOL1 Scenario Reference

OSPF > OSPF Network Point-to-Point
OSPF > OSPF Conditional Default Routing
EIGRP > EIGRP Network Statement
EIGRP > EIGRP Unequal-Cost Load-Balancing
EIGRP > EIGRP Summarization
RIP > Basic RIP Configuration
BGP > BGP Aggregation
BGP > BGP Default Routing
BGP > BGP Regular Expressions
BGP > BGP Regular Expressions
IPv6 > IPv6 Link-Local Addressing
IPv6 > IPv6 Global Aggregatable Addressing
IPv6 > IPv6 Link-Local Addressing
IPv6 > IPv6 OSPFv3
MPLS VPN > VRF Lite
Multicast > PIM Sparse-Dense Mode
Multicast > PIM Sparse-Mode
Multicast > PIM Sparse-Dense Mode
Multicast > PIM Sparse-Mode
Multicast > IGMP Timers
Multicast > Multicast Boundary
Security > Access Control with Dynamic Access Lists
Security > Access Control with Dynamic Access Lists
System Management > Terminal Line Settings
Security > Router IP Traffic Export
Security > Flexible Packet Matching
Security > Zone Based Firewall
Security > Role Based CLI
System Management > NTP
IP Services > DHCP Client
System Management > Kron Command Scheduler
IP Services > IP Event Dampening
QoS > Legacy Frame-Relay Traffic Shaping
QoS > Legacy FRTS with Per-VC Fragmentation
QoS > MQC Classification and Marking
QoS > MQC LLQ and Remaining Bandwidth Reservation
QoS > Using MQC CBWFQ with Legacy FRTS

Deep Dive Answers

- What is the best practice of originating a conditional default route?
  - When originating a default route based on presence of the default route learned from upstream peers it is recommended to have a floating static default route configured locally. Thus, if the upstream default route is lost, the local one will be advertised. This effectively allows for easier troubleshooting and reduction of ICMP unreachable storms sent by the access-level routers.
- What is the default metric for BGP routes redistributed into OSPF?
  - By default, BGP routers advertised into OSPF have the external metric value of one. All other routes will have the default metric value of 20.
- Why can't you safely redistribute IGP routes into iBGP and back into IGP?
  - iBGP does not have any loop prevention mechanics, so it is possible to create redistribution loops. While it is possible to enforce iBGP redistribution into IGP, you need to make sure you don't create feedback loops by using route tagging and filtering.
- Can OSPFv3 be used to route IPv4 traffic?
  - Even though OSPFv3 was designed to be address-family agnostic, in its original design it only supported IPv6 routing. A recent modification, that changes interpretation of OSPFv3 instance ID to address-family identifier allows for using OSPv3 for routing any address family. This allows for using OSPFv3 for integrated routing of both IPv4 and IPv6.
- What routing protocol was using IGMP as its transport?
  - DVMRP, the original distance-vector protocol for propagating multicast source subnets was using IGMPv1 messages for transporting its updates. Both protocols have been largely deprecated since their original use for Internet Mbone.
- How authentication proxy is different from dynamic access-lists?
  - Dynamic access-lists create openings based on certain router command executing, triggered by a user login. Authentication proxy is triggered by an HTTP/Telnet/FTP session going across the router and inspected by IOS firewall mechanics. The actually policy is not locally defined but pulled down from the AAA server as a set of access-list rules applied per user's IP address.
- What are the drawbacks of using SPAN/RSPAN compared to RITE?
  - SPAN works only for Ethernet ports and traffic. RSPAN has the benefit of transporting captured traffic across switched fabric, but neither of SPAN flavors have the ability to capture traffic on WAN media. The main benefit of using RITE, therefore, is being able to capture and monitor WANT traffic, using flexible filtering conditions.

- How is ERSPAN different from RSPAN?
    - Encapsulated Remote SPAN is a variant of SPAN that encapsulates captured frame into an IP based tunnel at the line rate and send them to the capturing device using IP routing. This allows for more flexible placement of capturing devices, and avoiding using system-wide RSPAN VLAN to groom the captured traffic to its destination.
- What are the specifics of ZFW "self" zone?
    - The "self" zone represent router itself and has some limitations as related to traffic inspection. Specifically, you are allowed to inspect only TCP, UDP and ICMP protocol, all other protocols support only stateless behavior using access-lists and drop/pass statement. Also, by default, the "self" zone is allowed to send/receive traffic from any other zone, unless you apply a zone-pair filtering policy.
- How could you manually shutdown and no-shutdown an interface every 3 hours?
    - In order to execute configuration commands from kron policies you need to use EEM scripts registered with "none" event handler and run the manually within the kron policy-lists. Kron CLI commands are limited only to single-line exec-level commands.
- Why fragmentation is not efficient without interleaving?
    - Interleaving is an essential step for fragmentation, as otherwise fragments of the large packet will be sent sequentially, effectively resulting in the same delay as sending the whole large packet. This is the reason why ATM cell-based fragmentation had no effect on VoIP quality when running IP over ATM using the most command AAL SNAP adaptation layer. ATM did not support cell interleaving for this AAL, rendering the fragmentation process useless.
- What other ways to enable FRF.12 do you know?
    - FRF.12 could be enabled using either legacy map-class configuration or interface-level command "frame-relay fragment". The second option is supported with MQC-based FRTS or Class-Based GTS.
- What is the reason for using the mincir as the bandwidth limit for MQC policy nested in a map-class?
    - Typically, minCIR reflects the CIR value of the FR service provider and the PVC CIR rate reflects the PIR value (if available). Therefore, in case of SP network congestion the shaper rate will shrink to minCIR (the SP actual CIR) and it is important for the QoS settings to not exceed this "extreme" value.

Lab 11 Solutions

Task 1.1 Solution

SW1:
```
spanning-tree vlan 4,28,38,56 priority 4096
spanning-tree vlan 1,3,5,7,17,23 priority 61440
```

SW1:
```
interface FastEthernet0/14
 spanning-tree vlan 4,28,38,56 port-priority 16
!
interface FastEthernet0/15
 spanning-tree vlan 4,28,38,56 port-priority 32
```

SW1:
```
interface FastEthernet0/15
 spanning-tree vlan 3,5,7,17,23 cost 1
```

Task 1.1 Breakdown

When discovering the topology you'll notice that SW4 is isolated, having no connections with SW1, SW2 or SW3.

Spanning-tree root bridge election is determined by the lowest bridge-ID. Bridge-ID is made up of two portions, the bridge priority and a MAC address. The bridge priority defaults to 32768, half of the maximum value 65535. Since each bridge-ID must be unique, and since each VLAN (by default) runs its own instance of spanning-tree, there must be some way to distinguish bridge-IDs between difference spanning-tree instances.

In older platforms, this was accomplished by assigning a single MAC address per VLAN. This solution results in a waste of MAC addresses, since each VLAN requires its own simply for identification. New Cisco switch platforms use the system-id extension to deal with this problem. The bridge-ID for a specific spanning-tree VLAN instance will be the configured priority plus the system-id extension. The system-id extension is effectively the VLAN number. Therefore, in order to ensure that SW1 is the root for VLANs 4, 28, 38, and 56 (even VLANs), and that SW2 is the root for VLANs 3, 5, 7, 17, and 23 (odd VLANs), the priority must be adjusted accordingly on SW1. Since a lower priority value is better, SW1 has been set with a low priority value, 4096, for even VLANs.

For odd VLANs, SW1's priority has been set to the configurable maximum value of 61440. These values are arbitrary as long as SW1 priority for the even VLANs is less than SW2's default priority (32768) plus the system-id extension (VLAN number). Furthermore, SW1 can use any arbitrary number to force SW2 to be the root for the odd VLANs, as long as it is greater than SW2's priority plus the system-id extension.

Note

SW3's spanning-tree priority is set to 61440 in the initial configuration. This should have been noticed before starting the lab.

Task 1.1 Verification

Verify that SW1 is the root bridge for even VLANs:

```
Rack1SW1#show spanning-tree bridge address
VLAN0001         001f.6ce6.8700
VLAN0003         001f.6ce6.8700
VLAN0004         001f.6ce6.8700
VLAN0005         001f.6ce6.8700
VLAN0007         001f.6ce6.8700
VLAN0017         001f.6ce6.8700
VLAN0023         001f.6ce6.8700
VLAN0028         001f.6ce6.8700
VLAN0038         001f.6ce6.8700
VLAN0056         001f.6ce6.8700

Rack1SW1#show spanning-tree root

                                        Root    Hello Max Fwd
Vlan                Root ID             Cost    Time  Age Dly  Root Port
---------------- ------------------- -------- ----- --- ---  --------
VLAN0001         32769 0018.738d.9e00         1    2   20  15  Fa0/15
VLAN0003         32771 0018.738d.9e00         1    2   20  15  Fa0/15
VLAN0004          4100 001f.6ce6.8700         0    2   20  15
VLAN0005         32773 0018.738d.9e00         1    2   20  15  Fa0/15
VLAN0007         32775 0018.738d.9e00         1    2   20  15  Fa0/15
VLAN0017         32785 0018.738d.9e00         1    2   20  15  Fa0/15
VLAN0023         32791 0018.738d.9e00         1    2   20  15  Fa0/15
VLAN0028          4124 001f.6ce6.8700         0    2   20  15
VLAN0038          4134 001f.6ce6.8700         0    2   20  15
VLAN0056          4152 001f.6ce6.8700         0    2   20  15
```



```
Rack1SW1#show span vlan 1-4094 | i root|VLAN
VLAN0001
VLAN0003
VLAN0004
             This bridge is the root
VLAN0005
VLAN0007
VLAN0017
VLAN0023
VLAN0028
             This bridge is the root
VLAN0038
             This bridge is the root
VLAN0056
             This bridge is the root
```

Verify that SW2 is the root bridge for odd VLANs:

```
Rack1SW2#show spanning-tree bridge address
VLAN0001         0018.738d.9e00
VLAN0003         0018.738d.9e00
VLAN0004         0018.738d.9e00
VLAN0005         0018.738d.9e00
VLAN0007         0018.738d.9e00
VLAN0017         0018.738d.9e00
VLAN0023         0018.738d.9e00
VLAN0028         0018.738d.9e00
VLAN0038         0018.738d.9e00
VLAN0056         0018.738d.9e00

Rack1SW2#show spanning-tree root

                                        Root    Hello Max Fwd
Vlan                Root ID             Cost    Time  Age Dly  Root Port
---------------- ------------------- -------- ----- --- ---  ---------
VLAN0001         32769 0018.738d.9e00       0    2   20  15
VLAN0003         32771 0018.738d.9e00       0    2   20  15
VLAN0004          4100 001f.6ce6.8700      19    2   20  15  Fa0/14
VLAN0005         32773 0018.738d.9e00       0    2   20  15
VLAN0007         32775 0018.738d.9e00       0    2   20  15
VLAN0017         32785 0018.738d.9e00       0    2   20  15
VLAN0023         32791 0018.738d.9e00       0    2   20  15
VLAN0028          4124 001f.6ce6.8700      19    2   20  15  Fa0/14
VLAN0038          4134 001f.6ce6.8700      19    2   20  15  Fa0/14
VLAN0056          4152 001f.6ce6.8700      19    2   20  15  Fa0/14
```



```
Rack1SW2#show span vlan 1-4094 | i VLAN|root
VLAN0001
               This bridge is the root
VLAN0003
               This bridge is the root
VLAN0004
VLAN0005
               This bridge is the root
VLAN0007
               This bridge is the root
VLAN0017
               This bridge is the root
VLAN0023
               This bridge is the root
VLAN0028
VLAN0038
VLAN0056

Rack1SW3#show span vlan 1-4094 | i VLAN|root
VLAN0001
VLAN0003
VLAN0004
VLAN0005
VLAN0007
VLAN0017
VLAN0023
VLAN0028
VLAN0038
VLAN0056
```

Verify the spanning-tree root ports for even numbered VLANs on SW2:

```
Rack1SW2#show spanning-tree root port
VLAN0001         This bridge is root
VLAN0003         This bridge is root
VLAN0004         FastEthernet0/14
VLAN0005         This bridge is root
VLAN0007         This bridge is root
VLAN0017         This bridge is root
VLAN0023         This bridge is root
VLAN0028         FastEthernet0/14
VLAN0038         FastEthernet0/14
VLAN0056         FastEthernet0/14
```


Shutdown Fa0/14 on SW1 and view the spanning-tree information:

```
Rack1SW2#show spanning-tree root port
VLAN0001         This bridge is root
VLAN0003         This bridge is root
VLAN0004         FastEthernet0/15
VLAN0005         This bridge is root
VLAN0007         This bridge is root
VLAN0017         This bridge is root
VLAN0023         This bridge is root
VLAN0028         FastEthernet0/15
VLAN0038         FastEthernet0/15
VLAN0056         FastEthernet0/15
```


Verify the spanning-tree root ports for odd numbered VLANs:

```
Rack1SW1#show spanning-tree root port
VLAN0001         FastEthernet0/15
VLAN0003         FastEthernet0/15
VLAN0004         This bridge is root
VLAN0005         FastEthernet0/15
VLAN0007         FastEthernet0/15
VLAN0017         FastEthernet0/15
VLAN0023         FastEthernet0/15
VLAN0028         This bridge is root
VLAN0038         This bridge is root
VLAN0056         This bridge is root
```

Shutdown Fa0/15 on SW2 and view the spanning-tree information:

```
Rack1SW1#show spanning-tree root port
VLAN0001         FastEthernet0/13
VLAN0003         FastEthernet0/13
VLAN0004         This bridge is root
VLAN0005         FastEthernet0/13
VLAN0007         FastEthernet0/13
VLAN0017         FastEthernet0/13
VLAN0023         FastEthernet0/13
VLAN0028         This bridge is root
VLAN0038         This bridge is root
VLAN0056         This bridge is root
```


Task 1.2 Solution

SW2:

```
interface FastEthernet0/24
 snmp trap mac-notification change added
!
snmp-server enable traps MAC-Notification
snmp-server host 187.1.3.100 CISCOTRAP  MAC-Notification
mac address-table notification change
```

Task 1.2 Breakdown

To enable SNMP trapping when a MAC address is added or removed from the CAM table, issue the global configuration commands **mac-address-table notification** and **snmp-server enable traps MAC-Notification**. Then, these traps are selectively enabled on a per-interface basis by issuing the **snmp trap mac-notifications** interface level command. These traps are then forwarded to the NMS station located at 187.1.3.100, using the community string CISCOTRAP.

Task 1.2 Verification

Verify SNMP MAC Address logging configuration:

```
Rack1SW2#clear mac address-table dynamic interface fa0/24
Rack1SW2#show mac address-table notification change
MAC Notification Feature is Enabled on the switch
Interval between Notification Traps : 1 secs
Number of MAC Addresses Added : 1
Number of MAC Addresses Removed : 0
Number of Notifications sent to NMS : 1
Maximum Number of entries configured in History Table : 1
Current History Table Length : 1
MAC Notification Traps are Enabled
History Table contents
----------------------
History Index 1, Entry Timestamp 4513569, Despatch Timestamp 4513569
MAC Changed Message :
Operation: Added   Vlan: 28  MAC Addr: 0011.bbbd.2da0 Dot1dBasePort: 26

Rack1SW2#show snmp host
Notification host: 187.1.3.100  udp-port: 162   type: trap
user: CISCOTRAP security model: v1
```


Task 1.3 Solution

SW1, SW2 and SW3:

```
ip access-list extended IPONLY
 permit ip any any
!
mac access-list extended IP_ARP
 permit any any 0x806 0x0
!
mac access-list extended PVST_PLUS
 permit any any 0x010B 0x0
!
mac access-list extended PVST
 permit any any lsap 0x4242 0x0
!
vlan access-map IPONLY 10
 action forward
 match ip address IPONLY
!
vlan access-map IPONLY 20
 action forward
 match mac address IP_ARP
!
vlan access-map IPONLY 30
 action forward
 match mac address PVST_PLUS
!
vlan access-map IPONLY 40
 action forward
 match mac address PVST
!
vlan access-map IPONLY 100
 action drop
!
vlan filter IPONLY vlan-list 56
```

SW1-SW3:

```
spanning-tree backbonefast
```

SW1:

```
spanning-tree vlan 4,28,38,56 forward-time 10
```

SW2:

```
spanning-tree vlan 1,3,5,7,17,23 forward-time 10
```


Task 1.3 Breakdown

The task describes a seemingly straightforward scenario in which only IP traffic is allowed to transit VLAN 56. This is accomplished by creating a VLAN access-list (VACL) which permits IP traffic, and denies all other. However, when this access-map is applied, other behind the scenes protocols stop working. These protocols include IP ARP and STP (PVST+ in our case). PVST+ BPDUs are transported in Ethernet frames using 802.3 LLC SNAP encapsulation over 802.1q trunks, having PID (Protocol ID) of 0x010B. Additionally, some PVST+ BPDUs are encapsulated into Ethernet 802.3 LLC frames, having SSAP/DSAP 0x42 to interoperate with classic IEEE STP.

In addition to permitting IP, these above protocols must be permitted. Although IP uses the ethertype 0x800, IP ARP uses its own ethertype value of 0x806. This value must also be permitted, otherwise ARP cannot work. Note that even though PVST+ uses LLC SNAP encapsulation, you can match the PID value using the “ethertype” keyword in MAC access-lists.

Task 1.3 Verification

To verify filtering, you can simulate a simple IPX network between R5 and R6, assuming the IOS versions support it.

```
R5:
ipx routing
!
interface FastEthernet 0/1
 ipx encapsulation sap
 ipx network 56

R6:
ipx routing
!
interface FastEthernet0/0
 ipx encapsulation sap
 ipx network 56
```

With the VLAN filter applied, try to IPX ping R6 from R5:

```
Rack1R6#show ipx interface FastEthernet0/0
FastEthernet0/0 is up, line protocol is up
  IPX address is 56.0015.62d0.4830, SNAP [up]
  Delay of this IPX network, in ticks is 1
  IPXWAN processing not enabled on this interface.
  IPX SAP update interval is 60 seconds
  IPX type 20 propagation packet forwarding is disabled
<output omitted>
```



```
Rack1R5#ping 56.0015.62d0.4830

Translating "56.0015.62d0.4830"

Type escape sequence to abort.
Sending 5, 100-byte IPX Novell Echoes to 56.0015.62d0.4830, timeout is
2 seconds:
 ...
Success rate is 0 percent (0/5)
```

Ensure that IP/ARP works fine:

```
Rack1R5#ping 187.1.56.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 187.1.56.6, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 1/2/4 ms
```

Verify the spanning-tree status. You should see a root port on SW2:

```
Rack1SW2#show spanning-tree vlan 56

VLAN0056
  Spanning tree enabled protocol rstp
  Root ID     Priority    24632
              Address     000f.8fe0.3500
              Cost        19
              Port        13 (FastEthernet0/13)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32824  (priority 32768 sys-id-ext 56)
              Address     000f.8fb2.e800
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/6            Desg FWD 19        128.8    P2p
Fa0/13           Altn BLK 19        128.15   P2p
Fa0/14           Root FWD 19        128.16   P2p
Fa0/15           Altn BLK 19        128.17   P2p
Fa0/16           Desg FWD 19        128.18   P2p
```

Remove VLAN filter:

```
Rack1SW1(config)#no vlan filter IPONLY vlan-list 56
Rack1SW2(config)#no vlan filter IPONLY vlan-list 56
Rack1SW3(config)#no vlan filter IPONLY vlan-list 56
```



```
Rack1R5#ping 56.0015.62d0.4830

Translating "56.0015.62d0.4830"

Type escape sequence to abort.
Sending 5, 100-byte IPX Novell Echoes to 56.0015.62d0.4830, timeout is
2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

Make sure that you turn the vlan filter back on, or you will lose the points for the section. Now verify STP backbone fast configuration:

```
Rack1SW1#show spanning-tree vlan 4 | include Forward
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 10 sec
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 10 sec
...

Rack1SW1#show spanning-tree vlan 1 | include Forward
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 10 sec
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
...

Rack1SW1#show spanning-tree backbonefast
BackboneFast is enabled

BackboneFast statistics
-----------------------
Number of transition via backboneFast (all VLANs)           : 0
Number of inferior BPDUs received (all VLANs)               : 0
Number of RLQ request PDUs received (all VLANs)             : 0
Number of RLQ response PDUs received (all VLANs)            : 0
Number of RLQ request PDUs sent (all VLANs)                 : 0
Number of RLQ response PDUs sent (all VLANs)                : 0
```

Task 1.4 Solution

```
SW1:
mls qos
!
interface FastEthernet0/7
 switchport access vlan 17
 switchport voice vlan 7
 mls qos trust cos
!
interface FastEthernet0/8
 switchport access vlan 17
 switchport voice vlan 7
 mls qos trust cos
!
define interface-range VPORTS FastEthernet 0/7 - 8
```


Task 1.4 Breakdown

The first step in configuring the 3560 to communicate with Cisco IP phones is to define how VoIP traffic will be carried. This task states that data traffic will be encapsulated in VLAN 7, and VoIP traffic will be encapsulated in VLAN 17. As the default port state of the 3560 is dynamic, a dot1q trunk will automatically be negotiated with the Cisco IP phone. The only configuration required to communicate with the phone is to apply both the access and voice VLAN to the port. Ensure that these VLANs are defined in the VLAN database.

Quality of Service processing is disabled on the 3560 by default. To enable QoS processing, issue the **`mls qos`** global configuration command. Next, the command **`mls qos trust cos`** has been issued on the interfaces connected to the IP phones. This instructs the switch to maintain the CoS value that is received on the interface.

Lastly, an interface range macro has been defined named VPORTS. This macro can be used in the future to reference ports Fa0/7 and Fa0/8 together. These macros can be used to reduce the administrative overhead of keeping track of which interfaces contain the same configuration. For example, if a certain range of interfaces are configured in an EtherChannel bundle, a macro could be created to manage all the member interfaces. This way, the member interfaces could be referenced by the macro, and it would be ensured that all member interfaces receive the same configuration.

Task 1.4 Verification

Verify MLS QoS configuration:

```
Rack1SW1#show mls qos int f0/7
FastEthernet0/7
trust state: trust cos
trust mode: trust cos
trust enabled flag: ena
COS override: dis
default COS: 0
DSCP Mutation Map: Default DSCP Mutation Map
Trust device: none
qos mode: port-based

Rack1SW1#show mls qos int f0/8
FastEthernet0/8
trust state: trust cos
trust mode: trust cos
trust enabled flag: ena
COS override: dis
default COS: 0
DSCP Mutation Map: Default DSCP Mutation Map
Trust device: none
qos mode: port-based
```

Verify Voice VLAN and appliance trust:

```
Rack1SW1#show interfaces fa0/7 switchport | inc Voice|Appl
Voice VLAN: 7 (VLAN0007)
Appliance trust: none

Rack1SW1#show interfaces fa0/8 switchport | inc Voice|Appl
Voice VLAN: 7 (VLAN0007)
Appliance trust: none
```


Task 1.5 Solution

R4:

```
!
! Configure PPP authentication using RADIUS and local fallback
!
aaa new-model
aaa authentication ppp RADIUS_LOCAL group radius local

!
username Rack1R5 password 0 C1SC0?2000
!
interface Serial0/1/0
 encapsulation ppp
 ppp authentication chap RADIUS_LOCAL
 clock rate 64000

radius-server host 192.10.1.100 key CISCO
ip radius source-interface Loopback0
radius-server deadtime 1
```

R5:

```
!
interface Serial0/1/0
 encapsulation ppp
 clockrate 64000
 ppp chap password 0 C1SC0?2000
 ppp chap hostname Rack1R5
```


Task 1.5 Breakdown

Since there is no real RADIUS server present, it is mandatory to specify the local authentication in the same AAA list. This way, the authenticator will try the local username database after failing to communicate with the RADIUS server. Make sure there is a list to disable console authentication, as enabling the AAA new model automatically enables console authentication.

Note that the escape sequence CTRL-V or ESC-Q must be used in order to enter a question mark in the password field. This username/password pair must also be configured in R4's local username database in order to authenticate R5.

The **username** and **ppp chap** commands with the "0" option after the password is telling the router that the password to come is in plain text format (i.e. unencrypted). This is also the default option when entering a password so the commands below will achieve the same result:

```
username Rack1R5 password 0 C1SC0?2000
username Rack1R5 password C1SC0?2000
```

If the commands are used with the "7" option after the password, the router will be expecting the password to come to be in encrypted form. Commonly this is used when a configuration is being copied from one router that has the **service password-encryption** command applied to another router. Below is the output of the command with the password in encrypted form:

```
username Rack1R5 password 7 123A5424312453567A7B74
```

Task 1.5 Verification

Verify PPP authentication:

```
Rack1R4#debug ppp negotiation
PPP protocol negotiation debugging is on

Rack1R4#debug ppp authentication
PPP authentication debugging is on

Rack1R4#debug aaa authentication
AAA Authentication debugging is on


Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface serial 0/1/0
Rack1R4(config-if)#no shutdown
```



```
Se0/1/0 PPP: Outbound cdp packet dropped
%SYS-5-CONFIG_I: Configured from console by console
%LINK-3-UPDOWN: Interface Serial0/1/0, changed state to up
AAA/BIND(0000002C): Bind i/f Serial0/1/0
Se0/1/0 PPP: Using default call direction
Se0/1/0 PPP: Treating connection as a dedicated line
Se0/1/0 PPP: Session handle[AC00002A] Session id[41]
Se0/1/0 PPP: Phase is ESTABLISHING, Active Open
Se0/1/0 PPP: Authorization NOT required
Se0/1/0 LCP: O CONFREQ [Closed] id 87 len 15
Se0/1/0 LCP:    AuthProto CHAP (0x0305C22305)
Se0/1/0 LCP:    MagicNumber 0x15DEC93A (0x050615DEC93A)
Se0/1/0 LCP: I CONFREQ [REQsent] id 41 len 10
Se0/1/0 LCP:    MagicNumber 0x1396F403 (0x05061396F403)
Se0/1/0 LCP: O CONFACK [REQsent] id 41 len 10
Se0/1/0 LCP:    MagicNumber 0x1396F403 (0x05061396F403)
Se0/1/0 LCP: I CONFACK [ACKsent] id 87 len 15
Se0/1/0 LCP:    AuthProto CHAP (0x0305C22305)
Se0/1/0 LCP:    MagicNumber 0x15DEC93A (0x050615DEC93A)
Se0/1/0 LCP: State is Open
Se0/1/0 PPP: Phase is AUTHENTICATING, by this end
Se0/1/0 CHAP: O CHALLENGE id 41 len 28 from "Rack1R4"
Se0/1/0 CHAP: I RESPONSE id 41 len 28 from "Rack1R5"
Se0/1/0 PPP: Phase is FORWARDING, Attempting Forward
Se0/1/0 PPP: Phase is AUTHENTICATING, Unauthenticated User
AAA/AUTHEN/PPP (0000002C): Pick method list 'RADIUS_LOCAL'
Se0/1/0 PPP: Sent CHAP LOGIN Request
Se0/1/0 PPP: Outbound cdp packet dropped
Se0/1/0 PPP: Outbound cdp packet dropped
Se0/1/0 CHAP: I RESPONSE id 41 len 28 from "Rack1R5"
Se0/1/0 CHAP: Ignoring Additional Response
Se0/1/0 AUTH: Timeout 1
Se0/1/0 PPP: Received LOGIN Response PASS
Se0/1/0 PPP: Phase is FORWARDING, Attempting Forward
Se0/1/0 PPP: Phase is AUTHENTICATING, Authenticated User
Se0/1/0 CHAP: O SUCCESS id 41 len 4
Se0/1/0 PPP: Phase is UP
Se0/1/0 IPCP: O CONFREQ [Closed] id 1 len 10
Se0/1/0 IPCP:    Address 148.1.45.4 (0x030694012D04)
Se0/1/0 CDPCP: O CONFREQ [Closed] id 1 len 4
<snip>
Se0/1/0 IPCP:    Address 148.1.45.4 (0x030694012D04)
Se0/1/0 IPCP: State is Open
Se0/1/0 IPCP: Install route to 148.1.45.5
Se0/1/0 CDPCP: I CONFACK [ACKsent] id 1 len 4
Se0/1/0 CDPCP: State is Open
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0, changed
state to up

Rack1R4#ping 187.1.45.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 187.1.45.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
```

Task 2.1 Solution

R1:
```
interface Serial0/0.134 multipoint
 ip ospf network point-to-multipoint
!
router ospf 1
 area 134 range 187.1.134.0 255.255.255.0
 area 134 virtual-link 150.1.3.3
 network 187.1.134.1 0.0.0.0 area 134
```

R3:
```
interface Serial1/0
 ip ospf network point-to-multipoint
!
router ospf 1
 area 134 range 187.1.134.0 255.255.255.0
 area 134 virtual-link 150.1.1.1
 area 134 virtual-link 150.1.4.4
 network 187.1.134.3 0.0.0.0 area 134
```

R4:
```
interface Serial0/0/0.134 multipoint
 ip ospf network point-to-multipoint
!
router ospf 1
 area 134 range 187.1.134.0 255.255.255.0
 area 134 virtual-link 150.1.3.3
 network 187.1.134.4 0.0.0.0 area 134
```

R1:
```
interface FastEthernet0/0
ip ospf authentication null
!
router ospf 1
area 134 virtual-link 150.1.3.3 authentication authentication-key CISCO
```


R3:
```
router ospf 1
 area 134 virtual-link 150.1.1.1 authentication authentication-key
CISCO
 area 134 virtual-link 150.1.4.4 authentication message-digest
 area 134 virtual-link 150.1.4.4 message-digest-key 1 md5 CISCO
```

R4:
```
router ospf 1
 area 134 virtual-link 150.1.3.3 authentication message-digest
 area 134 virtual-link 150.1.3.3 message-digest-key 1 md5 CISCO
```

SW1:
```
!
interface Vlan17
 ip ospf authentication null
!
router ospf 1
 router-id 150.1.7.7
 network 150.1.7.7 0.0.0.0 area 0
 network 187.1.13.7 0.0.0.0 area 7
 network 187.1.7.7 0.0.0.0 area 7
 network 187.1.17.7 0.0.0.0 area 0
```

SW2:
```
!
router ospf 1
 router-id 150.1.8.8
 network 150.1.8.8 0.0.0.0 area 38
 network 187.1.38.8 0.0.0.0 area 38
```

SW3:
```
!
ip routing
!
router ospf 1
 network 187.1.13.9 0.0.0.0 area 7
```


Tasks 2.1 Verification

Verify the OSPF neighbors:

```
Rack1R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.3.3         0   FULL/  -        -           187.1.134.3     OSPF_VL0
150.1.7.7         1   FULL/DR         00:00:33    187.1.17.7      FastEthernet0/0
150.1.3.3         0   FULL/  -        00:01:44    187.1.134.3     Serial0/0.134

Rack1R3#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.4.4         0   FULL/  -        -           187.1.134.4     OSPF_VL1
150.1.1.1         0   FULL/  -        -           187.1.134.1     OSPF_VL0
150.1.8.8         1   FULL/           00:00:35    187.1.38.8      FastEthernet0/1
150.1.4.4         0   FULL/  -        00:01:46    187.1.134.4     Serial1/0
150.1.1.1         0   FULL/           00:01:45    187.1.134.1     Serial1/0

Rack1R4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.3.3         0   FULL/  -        -           187.1.134.3     OSPF_VL0
150.1.5.5         0   FULL/  -        00:00:32    187.1.45.5      Serial0/1/0
150.1.3.3         0   FULL/  -        00:01:33    187.1.134.3     Serial0/0/0.134

Rack1R4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.3.3         0   FULL/  -        -           187.1.134.3     OSPF_VL0
150.1.5.5         0   FULL/  -        00:00:32    187.1.45.5      Serial0/1/0
150.1.3.3         0   FULL/  -        00:01:33    187.1.134.3     Serial0/0/0.134
```

Verify the OSPF network type on Frame Relay segment between R1, R3, and R4:

```
Rack1R3#show ip ospf interface s1/0
Serial1/0 is up, line protocol is up
  Internet Address 187.1.134.3/24, Area 134
  Process ID 1, Router ID 150.1.3.3, Network Type POINT_TO_MULTIPOINT,
Cost: 781
  Transmit Delay is 1 sec, State POINT_TO_MULTIPOINT,
  Timer intervals configured, Hello 30, Dead 120, Wait 120,Retransmit 5
<output omitted>
```


Verify that R1 and R4 see routes advertised by themselves with next-hop of R3, due to the OSPF network type:

```
Rack1R4#show ip route 150.1.1.1
Routing entry for 150.1.1.1/32
  Known via "ospf 1", distance 110, metric 846, type intra area
  Last update from 187.1.134.3 on Serial0/0/0.134, 03:54:06 ago
  Routing Descriptor Blocks:
  * 187.1.134.3, from 150.1.1.1, 03:54:06 ago, via Serial0/0/0.134
      Route metric is 846, traffic share count is 1

Rack1R1#show ip route 150.1.4.4
Routing entry for 150.1.4.4/32
  Known via "ospf 1", distance 110, metric 846, type intra area
  Last update from 187.1.134.3 on Serial0/0.134, 03:27:06 ago
  Routing Descriptor Blocks:
  * 187.1.134.3, from 150.1.4.4, 03:27:06 ago, via Serial0/0.134
      Route metric is 846, traffic share count is 1
```

Verify the OSPF virtual-link authentication:

```
Rack1R3#show ip ospf virtual-links
Virtual Link OSPF_VL1 to router 150.1.4.4 is up
  Run as demand circuit
  DoNotAge LSA allowed.
  Transit area 134, via interface Serial1/0, Cost of using 781
  Transmit Delay is 1 sec, State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:07
    Adjacency State FULL (Hello suppressed)
   Index 2/5, retransmission queue length 0, number of retransmission 0
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 0, maximum is 0
    Last retransmission scan time is 0 msec, maximum is 0 msec
  Message digest authentication enabled
    Youngest key id is 1
Virtual Link OSPF_VL0 to router 150.1.1.1 is up
  Run as demand circuit
  DoNotAge LSA allowed.
  Transit area 134, via interface Serial1/0, Cost of using 781
  Transmit Delay is 1 sec, State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:07
    Adjacency State FULL (Hello suppressed)
   Index 1/4, retransmission queue length 0, number of retransmission 1
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 1, maximum is 1
    Last retransmission scan time is 0 msec, maximum is 0 msec
  Simple password authentication enabled
```


Confirm that no authentication is enabled on area0 interfaces on R1 and SW1:

```
Rack1R1#show ip ospf interface fa0/0
FastEthernet0/0 is up, line protocol is up
  Internet Address 187.1.17.1/24, Area 0
  Process ID 1, Router ID 150.1.1.1, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State BDR, Priority 1
  Designated Router (ID) 150.1.7.7, Interface address 187.1.17.7
  Backup Designated router (ID) 150.1.1.1, Interface address 187.1.17.1
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:06
  Supports Link-local Signaling (LLS)
  Index 2/2, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 3
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.7.7  (Designated Router)
  Suppress hello for 0 neighbor(s)

Rack1SW1#show ip ospf interface vl17
Vlan17 is up, line protocol is up
  Internet Address 187.1.17.7/24, Area 0
  Process ID 1, Router ID 150.1.7.7, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 150.1.7.7, Interface address 187.1.17.7
  Backup Designated router (ID) 150.1.1.1, Interface address 187.1.17.1
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:01
  Supports Link-local Signaling (LLS)
  Index 2/4, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.1.1  (Backup Designated Router)
  Suppress hello for 0 neighbor(s)
```

Verify that the FR subnet is seen as 187.x.134.0/24:

```
Rack1R5#show ip route | i 187.1.134
O IA    187.1.134.0/24 [110/64] via 187.1.45.4, 04:12:17, Serial0/1/0

Rack1SW3#show ip route | i 187.1.134
O IA    187.1.134.0 [110/2] via 187.1.13.7, 04:07:07, FastEthernet0/15

Rack1SW2#show ip route | i 187.1.134
O IA    187.1.134.0 [110/1] via 187.1.38.3, 04:04:42, Vlan38
```


Task 2.2 Solution

R2:

```
router eigrp 10
  eigrp stub connected leak-map EIGRP_LEAK
!
route-map EIGRP_LEAK permit 10
!
interface Serial0/0.235 multipoint
 no ip split-horizon eigrp 10
```

R3:

```
interface Serial1/1.235 multipoint
 no ip split-horizon eigrp 10
```

R5:

```
interface Serial0/0/0
 no ip split-horizon eigrp 10
```

R3 and R5:

```
ip access-list standard EVEN
 permit 0.0.0.0 254.255.255.255
!
route-map EIGRP_TO_OSPF deny 5
 match tag 110
!
route-map EIGRP_TO_OSPF permit 10
 match ip address EVEN
 set metric-type type-1
 set tag 90
!
route-map EIGRP_TO_OSPF permit 20
 set metric 100
 set tag 90
!
route-map OSPF_TO_EIGRP deny 5
 match tag 90
!
route-map OSPF_TO_EIGRP permit 10
 set tag 110
```


R5:

```
!
router eigrp 10
 redistribute connected route-map CONNECTED_TO_EIGRP
 redistribute ospf 1 metric 1500 10 255 1 1500 route-map OSPF_TO_EIGRP
!
router ospf 1
 redistribute eigrp 10 subnets route-map EIGRP_TO_OSPF
 distance 171 0.0.0.0 255.255.255.255 R3_R6_LOOPBACKS
!

! R5 needs to see R3 Loopback0 via EIGRP, as R3 will be the MA in
! the scenario 5.1. Otherwise, RPF failures will occur
!
!ip access-list standard R3_R6_LOOPBACKS permit 150.1.6.6
  permit 150.1.3.0
!
route-map CONNECTED_TO_EIGRP permit 10
 match interface Loopback0
!
route-map CONNECTED_TO_EIGRP permit 20
 match interface Serial0/1/0
!
route-map CONNECTED_TO_OSPF permit 10
 no set metric 20
 set metric-type type-1
```

R3:

```
router eigrp 10
 redistribute ospf 1 metric 1500 10 255 1 1500 route-map OSPF_TO_EIGRP
!
router ospf 1
 redistribute eigrp 10 subnets route-map EIGRP_TO_OSPF
 distance 171 0.0.0.0 255.255.255.255 R6_LOOPBACK
!
ip access-list standard R6_LOOPBACK permit 150.1.6.6
```

SW2:

```
router rip
 version 2
 no auto-summary
 network 192.10.1.0
!
key chain RIP
 key 1
  key-string CISCO
!
interface Vlan28
 ip rip authentication mode md5
 ip rip authentication key-chain RIP
```



```
router ospf 1
 redistribute rip subnets route-map RIP_TO_OSPF

!
! Note that at this moment, as SW2 is running both RIP and eBGP with
! BB2, it will prefer all learned prefixes through eBGP due to the
! lower AD. Thus there are no prefixes in the routing table known via
! RIP; this will be fixed because of the requirements on task 2.4
!
router rip
 redistribute ospf 1 metric 1
!
access-list 1 permit 0.0.0.0 254.255.255.255
!
route-map RIP_TO_OSPF permit 10
 match ip address 1
 set metric-type type-1
!
route-map RIP_TO_OSPF permit 20
 set metric 100
 set metric-type type-2
```


Task 2.2 Breakdown

To isolate R2 from the query process, it should be set up as EIGRP stub. By default, when stub is configured on a router, it will advertise connected and summary routes; in our case only connected is needed.

To work around a potential link failure between any of R2, R3 and R5, split-horizon needs to be disabled on all 3 routers. Since R2 is configured as stub it will not advertise back any routes (though split-horizon is disabled) as a stub router cannot be a transit router. For this reason, R2 is configured as stub with leak-map.

The tasks says to configure two mutual points of redistribution between OSPF and EIGRP on R3 and R5. In order to prevent any transient routing loops during some route re-convergence, routes are tagged when re-distributed both from EIGRP into OSPF and from OSPF into EIGRP. It is always recommended within IGP domains that if a network is advertised by routing protocol X (example RIP), all routers speaking routing protocol X should install that route through X and not through other routing protocols (learned by re-distribution and prefer by lower AD). R6 loopback is redistributed as connected into EIGRP and is thus known in the EIGRP domain as external. To implement this logic, both R3 and R3 are set with OSPF AD of 171 for R6's Loopback0 prefix.

Task 2.2 Verification

Verify EIGRP stub settings:

```
Rack1R5#show ip eigrp neighbors detail
IP-EIGRP neighbors for process 10
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   187.1.235.2             Se0/0/0          170 01:53:46  151    906  0  40
   Restart time 01:53:31
   Version 12.4/1.2, Retrans: 1, Retries: 0, Prefixes: 22
   Stub Peer Advertising ( CONNECTED ) Routes
   Suppressing queries
1   187.1.235.3             Se0/0/0          148 1d12h     114    684  0  50
   Restart time 01:53:31
   Version 12.4/1.2, Retrans: 1, Retries: 0, Prefixes: 20
```


Ensure leak map is working on R2:

```
Rack1R3#show ip eigrp topology 187.1.56.0 255.255.255.0
IP-EIGRP (AS 10): Topology entry for 187.1.56.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is
20514560
  Routing Descriptor Blocks:
  187.1.235.5 (Serial1/1.235), from 187.1.235.5, Send flag is 0x0
      Composite metric is (20514560/28160), Route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 20100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
  187.1.235.2 (Serial1/1.235), from 187.1.235.2, Send flag is 0x0
      Composite metric is (21026560/2172416), Route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 40100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
```

Verify the external routes redistributed into OSPF:

```
Rack1R4#show ip route ospf | i E1|E2
O E2    187.1.235.0/24 [110/100] via 187.1.134.3, 01:56:45,
Serial0/0/0.134
O E2    187.1.56.0/24 [110/100] via 187.1.134.3, 01:56:45,
Serial0/0/0.134
O E2    187.1.5.0/24 [110/100] via 187.1.134.3, 01:56:45,
Serial0/0/0.134
O E1 192.10.1.0/24 [110/85] via 187.1.134.3, 01:55:53, Serial0/0/0.134
O E1    150.1.2.0/24 [110/84] via 187.1.134.3, 01:56:45,
Serial0/0/0.134
O E1    150.1.6.6/32 [110/84] via 187.1.134.3, 01:38:35,
Serial0/0/0.134
O E1    150.1.5.5/32 [110/84] via 187.1.134.3, 01:38:35,
Serial0/0/0.134
```

Check that OSPF routes are being re-distributed into EIGRP; check that R3 and R5 prefer EIGRP path to R6's Lopback0 prefix:

```
Rack1R6#show ip route eigrp | i D EX
D EX    187.1.134.4/32 [170/2223616] via 187.1.56.5, 01:42:02,
FastEthernet0/0
D EX    187.1.134.0/24 [170/2223616] via 187.1.56.5, 01:43:06,
FastEthernet0/0
D EX    187.1.134.1/32 [170/2223616] via 187.1.56.5, 01:42:02,
FastEthernet0/0
D EX    187.1.45.4/32 [170/2172416] via 187.1.56.5, 02:10:51,
FastEthernet0/0
D EX    187.1.45.0/24 [170/2172416] via 187.1.56.5, 02:10:51,
FastEthernet0/0
D EX    187.1.38.0/24 [170/2223616] via 187.1.56.5, 01:43:06,
FastEthernet0/0
D EX    187.1.17.0/24 [170/1711616] via 187.1.56.5, 01:42:02,
FastEthernet0/0
D EX    187.1.13.0/24 [170/1711616] via 187.1.56.5, 01:42:02,
FastEthernet0/0
D EX    187.1.3.0/24 [170/2223616] via 187.1.56.5, 01:43:06,
FastEthernet0/0
D EX    187.1.7.0/24 [170/1711616] via 187.1.56.5, 01:42:02,
FastEthernet0/0
D EX    187.1.4.0/24 [170/1711616] via 187.1.56.5, 01:42:02,
FastEthernet0/0
D EX 192.10.1.0/24 [170/1711616] via 187.1.56.5, 01:42:02,
FastEthernet0/0
D EX    150.1.3.0/24 [170/2223616] via 187.1.56.5, 02:00:18,
FastEthernet0/0
D EX    150.1.8.8/32 [170/1711616] via 187.1.56.5, 01:42:03,
FastEthernet0/0
D EX    150.1.7.7/32 [170/1711616] via 187.1.56.5, 01:42:03,
FastEthernet0/0
D EX    150.1.5.5/32 [170/156160] via 187.1.56.5, 02:10:53,
FastEthernet0/0
D EX    150.1.4.4/32 [170/1711616] via 187.1.56.5, 01:42:03,
FastEthernet0/0
D EX    150.1.3.3/32 [170/1711616] via 187.1.56.5, 01:43:07,
FastEthernet0/0
D EX    150.1.1.1/32 [170/1711616] via 187.1.56.5, 01:42:03,
FastEthernet0/0

Rack1R3#show ip route | i 150.1.6
D EX    150.1.6.6/32 [170/20517120] via 187.1.235.5, 01:42:46,
Serial1/1.235

Rack1R5#show ip route | i 150.1.6
D EX    150.1.6.6/32 [170/261120] via 187.1.56.6, 01:43:08,
FastEthernet0/1
```


R3's Loopback0 has a /24 network mask configured, and this prefix will show up as /32 in OSPF due OSPF network type. At the same time, when redistributed as connected into EIGRP it will show up as /24. Therefore, R5 will see the same Loopback0 destination as /24 and /32 networks via EIGRP and OSPF respectively.

However, for multicast section below, R5 needs to prefer the EIGRP path to R3's Loopback0. Therefore, In the multicast section, R3's Loopback OSPF network type will be changed to "Point-to-Point" so that it is advertised in OSPF as /24 as well.

```
Rack1R5#show ip route | i 150.1.3
D EX    150.1.3.0/24 [170/2221056] via 187.1.235.3, 11:44:44,
Serial0/0/0
O IA    150.1.3.3/32 [171/129] via 187.1.45.4, 11:27:33, Serial0/1/0
```

Test full connectivity between all internal networks with the following TCL script. Note that VLAN23 and the Frame Relay link between R6 and BB1 are excluded from any IGP and thus are not verified.

```
foreach i {
187.1.134.1
150.1.1.1
187.1.17.1
187.1.235.2
150.1.2.2
187.1.134.3
187.1.235.3
150.1.3.3
187.1.38.3
187.1.134.4
187.1.45.4
150.1.4.4
187.1.4.4
187.1.235.5
187.1.56.5
187.1.45.5
150.1.5.5
187.1.5.5
187.1.56.6
150.1.6.6
150.1.7.7
187.1.17.7
187.1.7.7
187.1.13.7
187.1.13.9
187.1.38.8
150.1.8.8
192.10.1.8
} { ping $i }
```


Task 2.3 Solution

R3:
```
router bgp 200
 neighbor 187.1.235.2 remove-private-as
 neighbor 187.1.235.5 remove-private-as
```

R4:
```
router bgp 200
 neighbor 187.1.45.5 remove-private-as
```

SW1:
```
interface Loopback77
 ip address 187.1.77.7 255.255.255.0
!
router bgp 65017
 network 187.1.77.0 mask 255.255.255.0
```

SW2:
```
router bgp 200
 neighbor 192.10.1.254 remove-private-as
```

R2:
```
router bgp 100
 network 187.1.235.0 mask 255.255.255.0
 aggregate-address 187.1.0.0 255.255.0.0 summary-only
 neighbor 204.12.1.254 unsuppress-map UNSUPPRESS
!
ip prefix-list NETWORK_235 seq 5 permit 187.1.235.0/24
!
route-map UNSUPPRESS permit 10
 match ip address prefix-list NETWORK_235
```

R6:
```
router bgp 100
 aggregate-address 187.1.0.0 255.255.0.0 summary-only
```

Task 2.3 Breakdown

The task states that BGP devices outside AS 200 should see this prefix as originated in AS 200. By removing the private AS number when AS 200 passes updates upstream, the private AS configuration is transparent to the rest of the network.

When BGP aggregation is configured, the aggregate-address (along with all subnets of the aggregate) are candidate to be advertised to the rest of the BGP domain. By adding the **summary-only** keyword, these subnets advertisements are suppressed. By configuring unsuppress map on R2, traffic from AS 54 will prefer to come in to R2. This is due to the fact that all routers throughout the network will always choose the longest match in the IP routing table. As R6 is only advertising the shorter match, this path will not be used unless the subnet information is lost from R2.

Task 2.3 Verification

Verify Loopback77 prefix in the BGP table on R3:

```
Rack1R3#show ip bgp | include 77|Netw
   Network          Next Hop            Metric LocPrf Weight Path
*> 187.1.77.0/24    187.1.134.1                            0 65017 i
```

Verify the same prefix in AS100:

```
Rack1R5#show ip bgp 187.1.77.0
BGP routing table entry for 187.1.77.0/24, version 49
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
        1    3    4
  200
    187.1.45.4 from 187.1.45.4 (150.1.4.4)
      Origin IGP, localpref 100, valid, external
  200
    187.1.235.3 from 187.1.235.3 (150.1.3.3)
      Origin IGP, localpref 100, valid, external, best
```

Verify the prefixes advertised to AS54 by R6:

```
Rack1R6#show ip bgp neighbors 54.1.1.254 advertised-routes
BGP table version is 26, 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
*> 187.1.0.0        0.0.0.0                            32768 i
*>i205.90.31.0      187.1.235.3              0    100      0 200 254 ?
*>i220.20.3.0       187.1.235.3              0    100      0 200 254 ?
*>i222.22.2.0       187.1.235.3              0    100      0 200 254 ?
```


Verify the prefixes advertised to AS54 by R2:

```
Rack1R2#show ip bgp neighbors 204.12.1.254 advertised-routes
BGP table version is 41, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 187.1.0.0        0.0.0.0                            32768 i
s> 187.1.235.0/24   0.0.0.0                  0         32768 i
*> 205.90.31.0      187.1.235.3                            0 200 254 ?
*> 220.20.3.0       187.1.235.3                            0 200 254 ?
*> 222.22.2.0       187.1.235.3                            0 200 254 ?
```

Task 2.4 Solution

SW2:

```
router bgp 200
 network 192.10.1.0
 network 205.90.31.0
 network 220.20.3.0
 network 222.22.2.0
!
router bgp 200
 distance bgp 121 200 200
```

R3:

```
!
interface Loopback 33
 ip address 150.1.33.33 255.255.255.0
!
interface Loopback 133
 ip address 150.1.133.133 255.255.255.0
!
ip prefix-list LOOPBACK33 permit 150.1.33.0/24
ip prefix-list LOOPBACK133 permit 150.1.133.0/24
!
route-map SET_COMMUNITY permit 10
 match ip address prefix-list LOOPBACK33
 set community 100:542
!
route-map SET_COMMUNITY permit 20
 match ip address prefix-list LOOPBACK133
 set community 100:546
!
ip bgp-community new-format
!
router bgp 200
 network 150.1.33.0 mask 255.255.255.0 route-map SET_COMMUNITY
 network 150.1.133.0 mask 255.255.255.0 route-map SET_COMMUNITY
 neighbor 187.1.235.2 send-community
 neighbor 187.1.235.5 send-community
 neighbor 187.1.134.4 send-community
```


R4:
```
ip bgp-community new-format
```

R5:
```
router bgp 100
 neighbor 187.1.56.6 send-community
 neighbor 187.1.235.2 send-community
!
ip bgp-community new-format
```

R2:
```
ip bgp-community new-format
!
ip community-list standard 100:542 permit 100:542
!
route-map TO_BB3 permit 10
 match community 100:542
 set as-path prepend 100 100 100
!
route-map TO_BB3 permit 100

router bgp 100
 neighbor 187.1.235.5 send-community
 neighbor 204.12.1.254 route-map TO_BB3 out
```

R6:
```
ip bgp-community new-format
!
ip community-list standard 100:546 permit 100:546
!
route-map TO_BB1 permit 10
 match community 100:546
 set as-path prepend 100 100 100
!
route-map TO_BB1 permit 100
!
router bgp 100
 neighbor 187.1.56.5 send-community
 neighbor 54.1.1.254 route-map TO_BB1 out
```

Task 2.4 Breakdown

Debugging RIP will show you what routes are learned. Switch 2 is learning routes via both RIP and BGP. BGP will win for the routing table, since eBGP has an AD of 20, compared to the AD of 120 for RIP. In order to make SW2 prefer the prefixes through RIP, we'll configure BGP with a distance for eBGP prefixes higher than RIP which is 120:

```
07:21:13: RIP: received v2 update from 192.10.1.254 on Vlan28
07:21:13:      205.90.31.0/24 via 0.0.0.0 in 7 hops
07:21:13:      220.20.3.0/24 via 0.0.0.0 in 7 hops
07:21:13:      222.22.2.0/24 via 0.0.0.0 in 7 hops
```


Task 2.4 Verification

See if prefixes appear in BGP table and are still preferred via RIP:

```
Rack1SW2#show ip bgp
BGP table version is 47, local router ID is 150.1.8.8
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
<output omitted>
*> 192.10.1.0       0.0.0.0                  0         32768 i
*> 205.90.31.0      192.10.1.254             7         32768 i
*                   192.10.1.254             0             0 254 ?
*> 220.20.3.0       192.10.1.254             7         32768 i
*                   192.10.1.254             0             0 254 ?
*> 222.22.2.0       192.10.1.254             7         32768 i
*                   192.10.1.254             0             0 254 ?

Rack1SW2#show ip route rip
R    222.22.2.0/24 [120/7] via 192.10.1.254, 00:00:13, Vlan28
R    220.20.3.0/24 [120/7] via 192.10.1.254, 00:00:13, Vlan28
R    205.90.31.0/24 [120/7] via 192.10.1.254, 00:00:13, Vlan28

Rack1R6#show ip bgp 150.1.33.0
BGP routing table entry for 150.1.33.0/24, version 55
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  dvertised to update-groups:
        2
  200
    187.1.235.3 (metric 2172416) from 187.1.56.5 (150.1.5.5)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Community: 100:542

Rack1R6#show ip bgp 150.1.133.0
BGP routing table entry for 150.1.133.0/24, version 56
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
        2
  200
    187.1.235.3 (metric 2172416) from 187.1.56.5 (150.1.5.5)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Community: 100:546
```



```
Rack1R2#show ip bgp 150.1.33.0
BGP routing table entry for 150.1.33.0/24, version 12
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           3
  200
    187.1.235.3 from 187.1.235.5 (150.1.5.5)
      Origin IGP, metric 0, localpref 100, valid, internal
      Community: 100:542
  200
    187.1.235.3 from 187.1.235.3 (150.1.3.3)
      Origin IGP, metric 0, localpref 100, valid, external, best
      Community: 100:542

Rack1R2#show ip bgp 150.1.133.0
BGP routing table entry for 150.1.133.0/24, version 13
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           3
  200
    187.1.235.3 from 187.1.235.5 (150.1.5.5)
      Origin IGP, metric 0, localpref 100, valid, internal
      Community: 100:546
  200
    187.1.235.3 from 187.1.235.3 (150.1.3.3)
      Origin IGP, metric 0, localpref 100, valid, external, best
      Community: 100:546

Rack1R4#show ip bgp 150.1.33.0
BGP routing table entry for 150.1.33.0/24, version 100
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Flag: 0x880
  Advertised to update-groups:
        2
  Local
    187.1.134.3 (metric 64) from 187.1.134.3 (150.1.3.3)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Community: 100:542

Rack1R4#show ip bgp 150.1.133.0
BGP routing table entry for 150.1.133.0/24, version 99
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Flag: 0x880
  Advertised to update-groups:
        2
  Local
    187.1.134.3 (metric 64) from 187.1.134.3 (150.1.3.3)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Community: 100:546
```



```
BB1>show ip bgp
BGP table version is 477, local router ID is 212.18.3.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
...
*> 150.1.33.0/24    54.1.1.6                               0 100 200 i
*  150.1.133.0/24   54.1.1.6                               0 100 100
100 100 200 i
*>i                 172.16.4.3               0    100      0 100 200 i

BB3>show ip bgp
BGP table version is 579, local router ID is 31.3.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i150.1.33.0/24    172.16.4.1               0    100      0 100 200 i
*                   204.12.1.2                             0 100 100
100 100 200 i
*> 150.1.133.0/24   204.12.1.2                             0 100 200 i
```

Task 3.1 Solution

R1:

```
ipv6 unicast-routing
!
interface Tunnel14
 ipv6 address 2001:187:1:14::1/64
 tunnel source 150.1.1.1
 tunnel destination 150.1.4.4
!
interface Tunnel16
 ipv6 address 2001:187:1:16::1/64
 tunnel source 150.1.1.1
 tunnel destination 150.1.6.6
!
interface FastEthernet0/0
 ipv6 address 2001:187:1:17::1/64
```

R4:

```
ipv6 unicast-routing
!
interface Tunnel14
 ipv6 address 2001:187:1:14::4/64
 tunnel source 150.1.4.4
 tunnel destination 150.1.1.1
!
interface Tunnel46
 ipv6 address 2001:187:1:46::4/64
 tunnel source 150.1.4.4
 tunnel destination 150.1.6.6
!
interface FastEthernet0/0
 ipv6 address 2001:187:1:4::4/64
```

R6:

```
ipv6 unicast-routing
!
interface Tunnel16
 ipv6 address 2001:187:1:16::6/64
 tunnel source 150.1.6.6
 tunnel destination 150.1.1.1
!
interface Tunnel46
 ipv6 address 2001:187:1:46::6/64
 tunnel source 150.1.6.6
 tunnel destination 150.1.4.4
!
interface FastEthernet 0/0
 ipv6 address 2001:187:1:56::6/64
```


SW1:
```
sdm prefer dual-ipv4-and-ipv6 routing
!
! Reboot SW1
!
ipv6 unicast-routing
!
interface Vlan17
 ipv6 address 2001:187:1:17::7/64
```

R1:
```
interface Tunnel14
 ipv6 ospf 10 area 1 instance 99
!
interface Tunnel16
 ipv6 ospf 10 area 1 instance 99
!
interface FastEthernet0/0
 ipv6 ospf 10 area 0 instance 99

ipv6 router ospf 1
 area 1 range 2001:187:1::/57
```

R4:
```
interface Tunnel14
 ipv6 ospf 40 area 1 inst 99
!
interface Tunnel46
 ipv6 ospf 40 area 1 inst 99
!
interface FastEthernet0/0
 ipv6 ospf 40 area 1 inst 99
```

R6:
```
interface Tunnel16
 ipv6 ospf 6 area 1 inst 99
!
interface Tunnel46
 ipv6 ospf 6 area 1 inst 99
!
interface FastEthernet 0/0
 ipv6 ospf 6 area 1 inst 99
```

SW1:
```
interface Vlan 17
 ipv6 ospf 100 area 0 instance 99
```


Task 3.1 Verification

Verify basic connectivity:

```
Rack1R1#ping 2001:187:1:14::4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:187:1:14::4, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 140/141/144
ms

Rack1R1#ping 2001:187:1:16::6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:187:1:16::6, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 156/157/160
ms

Rack1R1#ping 2001:187:1:17::7

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:187:1:17::7, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/4 ms

Rack1R4#ping 2001:187:1:46::6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:187:1:46::6, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 88/88/89 ms
```


Verify the OSPF routes on R1 and R4:

```
Rack1R6#show ipv6 route ospf
IPv6 Routing Table - Default - 10 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:187:1:4::/64 [110/1001]
     via FE80::215:62FF:FE41:FD9C, Tunnel46
O   2001:187:1:14::/64 [110/2000]
     via FE80::215:62FF:FE41:FD9C, Tunnel46
OI  2001:187:1:17::/64 [110/1001]
     via FE80::213:80FF:FEAD:7A80, Tunnel16

Rack1R1#show ipv6 route ospf
IPv6 Routing Table - 11 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:187:1::/57 [110/0]
     via ::, Null0
O   2001:187:1:4::/64 [110/11112]
     via FE80::215:62FF:FE41:FD9C, Tunnel14
O   2001:187:1:46::/64 [110/12111]
     via FE80::215:62FF:FE41:FD9C, Tunnel14
     via FE80::21A:2FFF:FE78:4678, Tunnel16
O   2001:187:1:56::/64 [110/11112]
     via FE80::21A:2FFF:FE78:4678, Tunnel16

Rack1R4#show ipv6 rout ospf
IPv6 Routing Table - Default - 10 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:187:1:16::/64 [110/2000]
     via FE80::21A:2FFF:FE78:4678, Tunnel46
OI  2001:187:1:17::/64 [110/1001]
     via FE80::213:80FF:FEAD:7A80, Tunnel14
O   2001:187:1:56::/64 [110/1001]
     via FE80::21A:2FFF:FE78:4678, Tunnel46
```



```
Rack1SW1#show ipv6 route ospf
IPv6 Routing Table - Default - 4 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       R - RIP, D - EIGRP, EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
OI  2001:187:1::/57 [110/12112]
     via FE80::213:80FF:FEAD:7A80, Vlan17
```

Task 3.2 Solution

R1:

```
interface tunnel16
 ipv6 ospf cost 22222
```

Task 3.2 Verification

```
Rack1SW1#traceroute 2001:187:1:56::6

Type escape sequence to abort.
Tracing the route to 2001:187:1:56::6

  1 2001:187:1:17::1 0 msec 9 msec 0 msec
  2 2001:187:1:14::4 109 msec 117 msec 109 msec
  3 2001:187:1:46::6 143 msec 143 msec 142 msec


Rack1R6#traceroute  2001:187:1:17::7

Type escape sequence to abort.
Tracing the route to 2001:187:1:17::7

  1 2001:187:1:16::1 120 msec 120 msec 124 msec
  2 2001:187:1:17::7 120 msec 125 msec 144 msec
```


Task 4.1 Solution

R4:

```
ip vrf R4
 rd 4:4
 route-target export 1:1
 route-target import 1:1

interface FastEthernet 0/1
 ip vrf forwarding R4
 ip address 192.168.4.4 255.255.255.0
 no shut

router rip
 address-family ipv4 vrf R4
 version 2
 network 192.168.4.0
```

R6:

```
ip vrf R6
 rd 6:6
 route-target export 1:1
 route-target import 1:1
!
interface fastEthernet 0/1
 ip vrf forwarding R6
 ip address 192.168.6.6 255.255.255.0
 no shut
!
router eigrp 1
 address-family ipv4 vrf R6
 autonomous-system 99
 network 192.168.6.6 0.0.0.0
```


Task 4.1 Verification

Verify that R4/R6 receive RIP/EIGRP routes:

```
Rack1R6#show ip route vrf R6 eigrp
     64.0.0.0/32 is subnetted, 1 subnets
D       64.64.64.64 [90/156160] via 192.168.6.100, 00:37:32,
FastEthernet0/1
     62.0.0.0/32 is subnetted, 1 subnets
D       62.62.62.62 [90/156160] via 192.168.6.100, 00:37:32,
FastEthernet0/1
     61.0.0.0/32 is subnetted, 1 subnets
D       61.61.61.61 [90/156160] via 192.168.6.100, 00:37:32,
FastEthernet0/1
     60.0.0.0/32 is subnetted, 1 subnets
D       60.60.60.60 [90/156160] via 192.168.6.100, 00:37:32,
FastEthernet0/1

Rack1R4#show ip route vrf R4 rip
     42.0.0.0/32 is subnetted, 1 subnets
R       42.42.42.42 [120/1] via 192.168.4.100, 00:00:10,
FastEthernet0/1
     40.0.0.0/32 is subnetted, 1 subnets
R       40.40.40.40 [120/1] via 192.168.4.100, 00:00:10,
FastEthernet0/1
     41.0.0.0/32 is subnetted, 1 subnets
R       41.41.41.41 [120/1] via 192.168.4.100, 00:00:10,
FastEthernet0/1
     44.0.0.0/32 is subnetted, 1 subnets
R       44.44.44.44 [120/1] via 192.168.4.100, 00:00:10,
FastEthernet0/1
```


Task 4.2 Solution

R4:

```
interface tunnel46
 ip address 187.1.46.4 255.255.255.0
!
router bgp 200
 no bgp default ipv4-unicast
 neighbor 187.1.46.6 remote-as 100
 address-family vpnv4 unicast
 neighbor 187.1.46.6 activate
 address-family ipv4 vrf R4
redist rip

router rip
 address-fami ipv4 vrf R4
  redist bgp 200 metric 3
  network 192.168.4.0
```

R6:

```
interface tunnel46
 ip address 187.1.46.6 255.255.255.0

router bgp 100
 no bgp default ipv4-unicast
 neighbor 187.1.46.4 remote-as 200
 address-family vpnv4 uni
 neigh 187.1.46.4 activate
 address-family ipv4 vrf R6
 redist eigrp 99
!
router eigrp 1
address-fam ipv4 vrf R6
 redist bgp 100 metric 10000 1 255 1 1500
 network 192.168.6.0
```


Task 4.2 Verification

```
Rack1SW4#show ip route vrf A rip
     64.0.0.0/32 is subnetted, 1 subnets
R       64.64.64.64 [120/3] via 192.168.4.4, 00:00:00, FastEthernet0/4
R    192.168.6.0/24 [120/3] via 192.168.4.4, 00:00:00, FastEthernet0/4
     62.0.0.0/32 is subnetted, 1 subnets
R       62.62.62.62 [120/3] via 192.168.4.4, 00:00:00, FastEthernet0/4
     61.0.0.0/32 is subnetted, 1 subnets
R       61.61.61.61 [120/3] via 192.168.4.4, 00:00:00, FastEthernet0/4
     60.0.0.0/32 is subnetted, 1 subnets
R       60.60.60.60 [120/3] via 192.168.4.4, 00:00:00, FastEthernet0/4

Rack1SW4#show ip route vrf B eigrp
     42.0.0.0/32 is subnetted, 1 subnets
D EX    42.42.42.42 [170/258816] via 192.168.6.6, 00:01:26,
FastEthernet0/6
D EX 192.168.4.0/24 [170/258816] via 192.168.6.6, 00:01:26,
FastEthernet0/6
     40.0.0.0/32 is subnetted, 1 subnets
D EX    40.40.40.40 [170/258816] via 192.168.6.6, 00:01:26,
FastEthernet0/6
     41.0.0.0/32 is subnetted, 1 subnets
D EX    41.41.41.41 [170/258816] via 192.168.6.6, 00:01:26,
FastEthernet0/6
     44.0.0.0/32 is subnetted, 1 subnets
D EX    44.44.44.44 [170/258816] via 192.168.6.6, 00:01:26,
FastEthernet0/6

Rack1SW4#ping vrf A 61.61.61.61

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 61.61.61.61, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/36/40 ms

Rack1SW4#ping vrf B 41.41.41.41

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 41.41.41.41, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/36/40 ms

Rack1SW4#ping vrf B 44.44.44.44 sou 62.62.62.62

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 44.44.44.44, timeout is 2 seconds:
Packet sent with a source address of 62.62.62.62
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/36/40 ms
```


Task 5.1 Solution

R3:

```
interface Loopback0
 ip pim sparse-mode
 ip ospf network point-to-point
!
ip pim bsr-candidate Loopback0 0
```

R4:

```
ip pim rp-candidate Serial0/0/0.134 group-list R4_GROUP
!
ip access-list standard R4_GROUP
 permit 224.0.0.0 7.255.255.255
```

R5:

```
ip pim rp-candidate Serial0/0/0 group-list R5_GROUP
!
ip access-list standard R5_GROUP
 permit 232.0.0.0 7.255.255.255
```

Task 5.1 Verification

Previous manipulations done in the route redistribution section ensure that R5 prefers reaching R3's Loopback0 via EIGRP. All you need to do is make sure RP mapping have propagated across the domain.

```
Rack1R1#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/5
  RP 187.1.134.4 (?), v2
    Info source: 150.1.3.3 (?), via bootstrap, priority 0, holdtime 150
         Uptime: 00:00:27, expires: 00:02:08
Group(s) 232.0.0.0/5
  RP 187.1.235.5 (?), v2
    Info source: 150.1.3.3 (?), via bootstrap, priority 0, holdtime 150
         Uptime: 00:00:27, expires: 00:02:09
```


Task 5.2 Solution

R1:

```
!
interface Tunnel14
 ip unnumbered Loopback0
 ip pim sparse-mode
 !
ip mroute 0.0.0.0 0.0.0.0 Tunnel 14
```

R4:

```
!
interface Tunnel14
 ip unnumbered Loopback0
 ip pim sparse-mode
```

SW1:

```
!
interface Vlan7
 ip igmp join-group 228.34.28.100
```

Task 5.2 Verification

Try pinging multicast group from R4 before configuring the tunnel:

```
Rack1R4#ping 228.34.28.100 repeat 5 source Fastethernet0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 228.34.28.100, timeout is 2 seconds:
Packet sent with a source address of 187.1.4.4
 ...

Rack1R3#show ip mroute 228.34.28.100
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 228.34.28.100), 00:00:05/00:02:55, RP 187.1.134.4, flags: SP
  Incoming interface: Serial1/0, RPF nbr 187.1.134.4
  Outgoing interface list: Null
```



```
Rack1R1#show ip mroute 228.34.28.100
Group 228.34.28.100 not found
```

Now configure the tunnel and try pinging again:

```
Rack1R4#ping 228.34.28.100 repeat 5 source FastEthernet0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 228.34.28.100, timeout is 2 seconds:

Reply to request 0 from 187.1.17.7, 132 ms
Reply to request 1 from 187.1.17.7, 112 ms
Reply to request 2 from 187.1.17.7, 108 ms
Reply to request 3 from 187.1.17.7, 112 ms
Reply to request 4 from 187.1.17.7, 108 ms

Rack1R1#show ip mroute 228.34.28.100
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 228.34.28.100), 00:01:09/00:03:07, RP 187.1.134.4, flags: S
  Incoming interface: Tunnel14, RPF nbr 150.1.4.4, Mroute
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:01:09/00:03:07

(187.1.4.4, 228.34.28.100), 00:01:01/00:02:45, flags: T
  Incoming interface: Tunnel14, RPF nbr 150.1.4.4, Mroute
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:01:01/00:03:07
```


Task 5.3 Solution

R4 & R5:

```
ip access-list extended R3_R4_GROUPS
 permit ip host 150.1.3.3 any
 permit ip host 150.1.4.4 any
!
ip pim accept-register list R3_R4_GROUPS
```

R3 & R4:

```
ip pim register-source Loopback0
```

R1, R3, R4, R5, and SW1:

```
ip access-list standard R4_GROUP
 permit 224.0.0.0 7.255.255.255
!
ip access-list standard R5_GROUP
 permit 232.0.0.0 7.255.255.255
!
ip pim accept-rp 187.1.134.4 R4_GROUP
ip pim accept-rp 187.1.235.5 R5_GROUP
```


Task 5.3 Verification

Verify that before applying the configuration, for example, R5 can is allowed to send multicast traffic streams:

```
Rack1R5#ping 228.34.28.100 repeat 5 source fastEthernet 0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 228.34.28.100, timeout is 2 seconds:
Packet sent with a source address of 187.1.5.5

Reply to request 0 from 187.1.17.7, 196 ms
Reply to request 1 from 187.1.17.7, 196 ms
Reply to request 2 from 187.1.17.7, 196 ms
Reply to request 3 from 187.1.17.7, 196 ms
Reply to request 4 from 187.1.17.7, 196 ms

Rack1R1#show ip mroute 228.34.28.100
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 228.34.28.100), 11:32:52/00:03:14, RP 187.1.134.4, flags: S
  Incoming interface: Tunnel14, RPF nbr 150.1.4.4, Mroute
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 11:32:52/00:03:14

(187.1.5.5, 228.34.28.100), 00:01:45/00:03:07, flags: T
  Incoming interface: Tunnel14, RPF nbr 150.1.4.4, Mroute
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:01:45/00:03:14
```


Now apply the configuration; R5 should not be allowed, but R3 and R4 should:

```
Rack1R5#ping 228.34.28.100 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 228.34.28.100, timeout is 2
seconds:
 ...

Rack1R4#debug ip pim
PIM debugging is on


%PIM-1-INVALID_RP_REG: Received Register from router 187.1.235.5 for
group 228.34.28.100, 187.1.134.4 not willing to be RP

%PIM-4-INVALID_SRC_REG: Received Register from 187.1.235.5 for
(187.1.5.5, 228.34.28.100), not willing to be RP

Rack1R4#ping 228.34.28.100 repeat 5 source fastEthernet 0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 228.34.28.100, timeout is 2 seconds:
Packet sent with a source address of 187.1.4.4

Reply to request 0 from 187.1.17.7, 128 ms
Reply to request 1 from 187.1.17.7, 124 ms
Reply to request 2 from 187.1.17.7, 124 ms
Reply to request 3 from 187.1.17.7, 124 ms
Reply to request 4 from 187.1.17.7, 124 ms

Rack1R3#ping 228.34.28.100 repeat 5 source fastEthernet 0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 228.34.28.100, timeout is 2 seconds:
Packet sent with a source address of 187.1.3.3

Reply to request 0 from 187.1.17.7, 192 ms
Reply to request 1 from 187.1.17.7, 188 ms
Reply to request 2 from 187.1.17.7, 185 ms
Reply to request 3 from 187.1.17.7, 188 ms
Reply to request 4 from 187.1.17.7, 188 ms
```


Task 5.4 Solution

SW4

```
no ip multicast-routing distributed
mvr
mvr vlan 44
mvr group 239.6.6.6
mvr mode dynamic
!
vlan 44,66
 exit
!
interface FastEthernet 0/4
 switchport
 no ip address
 mvr type source
 switchport access vlan 44
 switchport mode access
!
interface FastEthernet 0/6
 switchport
 no ip address
 mvr type receiver
 switchport access vlan 66
 switchport mode access
!
interface Vlan44
 ip vrf forwarding A
 ip address 192.168.4.100 255.255.255.0
!
interface Vlan66
 ip vrf forwarding B
 ip address 192.168.6.100 255.255.255.0
```

R4:

```
!
! We enable dense-mode to allow flooding out of this interface
!
interface FastEthernet 0/1
  ip pim dense-mode
```

R6:

```
interface FastEthernet 0/1
 ip pim dense-mode
 ip igmp join-group 239.6.6.6
```


Task 5.4 Verification

```
Rack1SW4#show mvr
MVR Running: TRUE
MVR multicast VLAN: 44
MVR Max Multicast Groups: 256
MVR Current multicast groups: 1
MVR Global query response time: 5 (tenths of sec)
MVR Mode: dynamic

Rack1SW4#show mvr members
MVR Group IP    Status          Members
------------    ------          -------
239.006.006.006 ACTIVE          Fa0/6(d)

Rack1SW4#show mvr interface
Port      Type       Status           Immediate Leave
----      ----       ------           ---------------
Fa0/4     SOURCE     ACTIVE/UP        DISABLED
Fa0/6     RECEIVER   ACTIVE/UP        DISABLED

Rack1R4#show ip igmp vrf R4 groups
IGMP Connected Group Membership
Group Address    Interface                Uptime    Expires   Last
Reporter   Group Accounted
239.6.6.6        FastEthernet0/1          00:01:20  00:02:36
192.168.6.6
224.0.1.40       FastEthernet0/1          00:26:43  00:02:40
192.168.4.4

Rack1R4#ping vrf R4 239.6.6.6 repeat 5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 239.6.6.6, timeout is 2 seconds:

Reply to request 0 from 192.168.6.6, 20 ms
Reply to request 1 from 192.168.6.6, 16 ms
Reply to request 2 from 192.168.6.6, 16 ms
Reply to request 3 from 192.168.6.6, 16 ms
Reply to request 4 from 192.168.6.6, 16 ms
```



```
Rack1R4#show ip mroute vrf R4 239.6.6.6
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group,
       V - RD & Vector, v - Vector
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.6.6.6), 00:03:20/stopped, RP 0.0.0.0, flags: DC
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/1, Forward/Dense, 00:03:20/00:00:00

(192.168.4.4, 239.6.6.6), 00:03:20/00:02:13, flags: PT
  Incoming interface: FastEthernet0/1, RPF nbr 0.0.0.0
  Outgoing interface list: Null
```

Task 6.1 Solution

R2, R6, and SW2:

```
access-list 100 permit tcp any any
access-list 100 permit udp any any
access-list 100 deny 53 any any log
access-list 100 deny 55 any any log
access-list 100 deny 77 any any log
access-list 100 deny 103 any any log
access-list 100 permit ip any any
!
logging 187.1.38.100
```

R2:

```
interface FastEthernet0/0
 ip access-group 100 in
 ip access-group 100 out
```

R6:

```
interface Serial0/0/0
 ip access-group 100 in
 ip access-group 100 out
```

SW2:

```
interface Vlan28
 ip access-group 100 in
 ip access-group 100 out
```

Task 6.1 Breakdown

For the most part, this section is very straightforward. You are given specific items to block, and the devices are explicitly stated. There are a few additional items to keep in mind. When logging ACL entries, make sure that you have the logging set to an appropriate level. Since those messages are informational, logging will need to be at level 6 or 7. In this particular case, the default logging level is high enough, so no further configuration is needed. One other thing to watch carefully is the section that states “interest in the amount of packets that are denied by this filtering policy”. In this case, we are just logging when traffic is denied. It is possible that the section could also be alluding to using IP accounting with the “access-violations” option. This would be an example of a section where you may want to get additional clarification from the proctor whether they were just looking for general information, or tracking statistics for the denied traffic.

Task 6.1 Verification

```
Rack1R2#show ip interface fastEthernet 0/0 | i Inbound|Outgoing
  Outgoing access list is 100
  Inbound  access list is 100

Rack1SW2#show ip interface vlan 28 | i Inbound|Outgoing
  Outgoing access list is 100
  Inbound  access list is 100
```

Task 6.2 Solution

```
R6:
ip inspect name FIREWALL http audit-trail on
ip inspect name FIREWALL ftp audit-trail on
ip inspect name FIREWALL dns audit-trail off
ip inspect name FIREWALL h323 audit-trail off router-traffic
ip inspect audit-trail
!
ip access-list extended FROM_BB1
 permit tcp any eq bgp any
 permit tcp any any eq bgp
 deny 53 any any log
 deny 55 any any log
 deny 77 any any log
 deny 103 any any log

!
interface Serial0/0/0
 ip inspect FIREWALL out
 ip access-group FROM_BB1 in
```

Task 6.2 Breakdown

The access list created here takes the place of the access list from the prior section. Since you can only apply a single access list to an interface per direction, make sure that the access list integrates the requirements of both sections. In this case, the earlier section's requirements were to block some IP protocols, and to permit other traffic. This section's requirements are to only allow certain traffic inbound. By merging the earlier ACL deny entries with a couple permit statements for BGP, the requirements for both sections are met.


```
Rack1R6#show ip inspect all
Session audit trail is disabled
Session alert is enabled
one-minute (sampling period) thresholds are [400:500] connections
max-incomplete sessions thresholds are [400:500]
max-incomplete tcp connections per host is 50. Block-time 0 minute.
tcp synwait-time is 30 sec -- tcp finwait-time is 5 sec
tcp idle-time is 3600 sec -- udp idle-time is 30 sec
dns-timeout is 5 sec
Inspection Rule Configuration
 Inspection name FIREWALL
    http alert is on audit-trail is on timeout 3600
    ftp alert is on audit-trail is on timeout 3600
    dns alert is on audit-trail is off timeout 30
    sip alert is on audit-trail is off timeout 30
    h323 alert is on audit-trail is off timeout 3600
 inspection of router local traffic is enabled

Interface Configuration
 Interface Serial0/0
  Inbound inspection rule is not set
  Outgoing inspection rule is FIREWALL
    http alert is on audit-trail is on timeout 3600
    ftp alert is on audit-trail is on timeout 3600
    dns alert is on audit-trail is off timeout 30
    sip alert is on audit-trail is off timeout 30
    h323 alert is on audit-trail is off timeout 3600
 inspection of router local traffic is enabled
  Inbound access list is FROM_BB1
  Outgoing access list is 100

Rack1R6#show ip bgp summary
...

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down
State/PfxRcd
54.1.1.254      4    54     632     638       76    0    0 10:24:23       10
187.1.56.5      4   100    2030    2017       76    0    0 10:32:31

Rack1R5#copy running-config http://116.0.0.1/test
Address or name of remote host [116.0.0.1]?
Destination filename [test]?
%Error writing http://116.0.0.1/test (I/O error)

Rack1R6#
%FW-6-SESS_AUDIT_TRAIL_START: Start http session: initiator
(187.1.56.5:38248) -- responder (116.0.0.1:80)

%FW-6-SESS_AUDIT_TRAIL: Stop http session: initiator (187.1.56.5:38248)
sent 0 bytes -- responder (116.0.0.1:80) sent 0 bytes
```


Task 7.1 Solution

R6:

```
archive
 log config
  logging enable
  logging size 500
  notify syslog
!
logging 187.1.5.155
```

Task 7.1 Verification

Verify the change logging configuration:

```
Rack1R6#show archive log config all
idx   sess           user@line      Logged command
    1     1        console@console  |  logging enable
    2     1        console@console  |  logging size 500
    3     1        console@console  |  notify syslog
    4     1        console@console  |  logging 187.1.5.155

Rack1R6#show logging | section Trap logging
    Trap logging: level informational, 166 message lines logged
        Logging to 187.1.38.100  (udp port 514,  audit disabled,
              authentication disabled, encryption disabled, link up),
              80 message lines logged,
              0 message lines rate-limited,
              0 message lines dropped-by-MD,
              xml disabled, sequence number disabled
              filtering disabled
        Logging to 187.1.5.155  (udp port 514,  audit disabled,
              authentication disabled, encryption disabled, link up),
              4 message lines logged,
              0 message lines rate-limited,
              0 message lines dropped-by-MD,
              xml disabled, sequence number disabled
              filtering disabled
```


Task 7.2 Solution

```
R3:
ip wccp web-cache redirect-list 25
!
interface FastEthernet0/0
 ip wccp web-cache redirect in
!
access-list 25 deny 187.1.3.50
access-list 25 permit any
```

Task 7.2 Breakdown

By default, traffic from all hosts received or sent on an interface (depending on how redirection is configured) is candidate for redirection to a web cache engine. In the above scenario, all traffic except that which is sourced from 187.1.3.50 is eligible for caching.

Task 7.2 Verification

Verify the WCCP configuration:

```
Rack1R3#show ip wccp web-cache
Global WCCP information:
    Router information:
        Router Identifier:                   -not yet determined-
        Protocol Version:                    2.0

    Service Identifier: web-cache
        Number of Cache Engines:             0
        Number of routers:                   0
        Total Packets Redirected:            0
        Process:                             0
        Fast:                                0
        CEF:                                 0
        Redirect access-list:                25
        Total Packets Denied Redirect:       0
        Total Packets Unassigned:            0
        Group access-list:                   -none-
        Total Messages Denied to Group:      0
        Total Authentication failures:       0
        Total Bypassed Packets Received:     0

Rack1R3#show ip wccp interfaces
WCCP interface configuration:
    FastEthernet0/0
        Output services: 0
        Input services:  1
        Mcast services:  0
        Exclude In:      FALSE
```


Task 8.1 Solution

R3:

```
interface Serial1/0
 frame-relay traffic-shaping
 frame-relay class FRTS
!
map-class frame-relay FRTS
 frame-relay cir 192000
 frame-relay bc 19200
 frame-relay be 12800
```

Task 8.1 Breakdown

This task states that R3 should average 192Kbps on both VC 301 and 304, and that traffic bursts of up to 320Kbps should be allowed for a maximum period of 100ms. The following values can therefore be inferred from this description:

CIR = 192000bps
AR = 320000bps
Tc = 100ms

Using the formula Bc = CIR * Tc/1000:

Bc = 192000 * 100/1000
Bc = 192000 * 1/10
Bc = 19200

Using the formula Be = (AR - CIR) * Tc/1000

Be = (320000 - 192000) * 100/1000
Be = 128000 * 1/10
Be = 12800

Task 8.1 Verification

```
Rack1R3#show frame-relay pvc 304

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

DLCI = 304, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial1/0

  input pkts 2593          output pkts 2711         in bytes 221401
  out bytes 242072         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 971       out bcast bytes 75926
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 06:09:50, last time pvc status changed 06:09:45
  cir 192000    bc 19200     be 12800     byte limit 4000   interval
100
  mincir 96000     byte increment 2400  Adaptive Shaping none
  pkts 6         bytes 528       pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/40, 0 drop, 0 dequeued

Rack1R3#show frame-relay pvc 301

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

DLCI = 301, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial1/0

  input pkts 2373          output pkts 2752         in bytes 202607
  out bytes 246973         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 972       out bcast bytes 75960
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 06:09:53, last time pvc status changed 06:09:28
  cir 192000    bc 19200     be 12800     byte limit 4000   interval
100
  mincir 96000     byte increment 2400  Adaptive Shaping none
  pkts 7         bytes 868       pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/40, 0 drop, 0 dequeued
```


Task 8.2 Solution

R1:

```
class-map match-any CRITICAL
 match packet length min 80 max 100
 match protocol ospf
!
class-map ANY
 match any
!
policy-map MARK
 class CRITICAL
 class ANY
  set fr-de
!
interface Serial 0/0.134
  service-policy output MARK
```

Task 8.2 Verification

Rack1R1#show frame-relay pvc 103

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

DLCI = 103, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0.134

  input pkts 30970         output pkts 27394        in bytes 2863566
  out bytes 2645571        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 21
  out bcast pkts 10381     out bcast bytes 624772
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 1d08h, last time pvc status changed 1d06h
```



```
Rack1R1#show policy-map interface serial 0/0.134

 Serial0/0.134

  Service-policy output: MARK

    Class-map: CRITICAL (match-any)
      5 packets, 420 bytes
      5 minute offered rate 0 bps
      Match: packet length min 80 max 100
        5 packets, 420 bytes
        5 minute rate 0 bps
      Match: protocol ospf
        0 packets, 0 bytes
        5 minute rate 0 bps

    Class-map: ANY (match-all)
      26 packets, 2552 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      QoS Set
        fr-de
          Packets marked 24

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

Task 8.3 Solution

```
SW2:
mls qos
!
interface FastEthernet0/24
 speed 10
 srr-queue bandwidth shape  0  0  10  0
 srr-queue bandwidth limit 30
```


Task 8.3 Verification

```
Rack1SW2#show mls qos maps dscp-output-q
   Dscp-outputq-threshold map:
     d1 :d2     0      1      2      3      4      5      6      7      8      9
     ------------------------------------------------------------------------
      0 :     02-01  02-01  02-01  02-01  02-01  02-01  02-01  02-01  02-01  02-01
      1 :     02-01  02-01  02-01  02-01  02-01  02-01  03-01  03-01  03-01  03-01
      2 :     03-01  03-01  03-01  03-01  03-01  03-01  03-01  03-01  03-01  03-01
      3 :     03-01  03-01  04-01  04-01  04-01  04-01  04-01  04-01  04-01  04-01
      4 :     01-01  01-01  01-01  01-01  01-01  01-01  01-01  01-01  04-01  04-01
      5 :     04-01  04-01  04-01  04-01  04-01  04-01  04-01  04-01  04-01  04-01
      6 :     04-01  04-01  04-01  04-01

Rack1SW2#show mls qos interface FastEthernet 0/24 queueing
FastEthernet0/24
Egress Priority Queue : disabled
Shaped queue weights (absolute) :  0 0 10 0
Shared queue weights  :  25 25 25 25
The port bandwidth limit : 30  (Operational Bandwidth:30.44)
The port is mapped to qset : 1
```

Task 8.3 Breakdown

Setting the speed to 10M will meet the objective of the connection to BB2 being 10Mbps. The command srr-queue bandwidth limit will limit the overall egress to 30% of the physical speed, for the 3Mbps requirement. The command srr-queue bandwidth shape 0 0 10 0 will give 1/10th to the third queue. As mentioned in the QoS section of Volume 1, shaped weights still apply to the physical speed, not the bandwidth limit, when calculating queue rates.

Task 8.4 Solution

R5:

```
policy-map MARK
 class class-default
  police cir 256000 pir 512000
    conform-action set-prec-transmit 1
    exceed-action set-prec-transmit 0
    violate-action drop
!
interface FastEthernet 0/0
 service-policy input MARK
!
interface FastEthernet 0/1
 service-policy input MARK
```

Task 8.4 Verification

```
Rack1R5#show policy-map interface fastEthernet 0/0
 FastEthernet0/0

  Service-policy input: MARK

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      police:
          cir 256000 bps, bc 8000 bytes
          pir 512000 bps, be 16000 bytes
        conformed 0 packets, 0 bytes; actions:
          set-prec-transmit 1
        exceeded 0 packets, 0 bytes; actions:
          set-prec-transmit 0
        violated 0 packets, 0 bytes; actions:
          drop
        conformed 0 bps, exceed 0 bps, violate 0 bps

Rack1R5#show policy-map interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy input: MARK

    Class-map: class-default (match-any)
      2 packets, 148 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      police:
          cir 256000 bps, bc 8000 bytes
          pir 512000 bps, be 16000 bytes
        conformed 2 packets, 148 bytes; actions:
          set-prec-transmit 1
        exceeded 0 packets, 0 bytes; actions:
          set-prec-transmit 0
        violated 0 packets, 0 bytes; actions:
          drop
        conformed 0 bps, exceed 0 bps, violate 0 bps
```


Lab 12 Solutions

Task 1.1 Solution

SW1:

```
mac-address-table static 0030.1369.87a0 vlan 17 drop

errdisable recovery cause psecure-violation
errdisable recovery interval 60
!
interface FastEthernet0/7
 switchport mode access
 switchport port-security maximum 2
 switchport port-security
!
interface FastEthernet0/8
 switchport mode access
 switchport port-security maximum 2
 switchport port-security
```

Task 1.1 Breakdown

In addition to being used to restrict access to a specific MAC address, port-security can be used to limit the amount of MAC addresses that are allowed to send traffic into a port. This can be used on shared segments of the network in order to limit the amount of hosts that are allowed to access the network through a single port. As the default violation mode is shutdown, when the number of MAC addresses exceeds two, the interface is put into err-disabled state.

For the MAC restriction, the immediate reaction to this task is typically to use an extended MAC address access-list to deny traffic from this MAC address from entering interfaces Fa0/7 or Fa0/8. However, MAC address access-lists only affect non-IP traffic. Therefore, assuming that hosts on VLAN 17 are running IP (a fair assumption), using a MAC access-list to filter this host will have no effect.

As an alternative, traffic from this host has been effectively black holed by creating a static MAC address table (CAM table) entry for its MAC address. Much like static IP routing, a static MAC entry in the CAM table takes precedence over any dynamically learned reachability information.

Task 1.1 Verification

```
Rack1SW1#show port-security interface fa0/7
Port Security              : Enabled
Port Status                : Secure-down
Violation Mode             : Shutdown
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 2
Total MAC Addresses        : 0
Configured MAC Addresses   : 0
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 0000.0000.0000:0
Security Violation Count   : 0

Rack1SW1#show port-security interface fa0/8
Port Security              : Enabled
Port Status                : Secure-down
Violation Mode             : Shutdown
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 2
Total MAC Addresses        : 0
Configured MAC Addresses   : 0
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 0000.0000.0000:0
Security Violation Count   : 0
```

An additional MAC address is heard on the port and a violation occurs

↓ ↓ ↓

```
Rack1SW1#
%PM-4-ERR_DISABLE: psecure-violation error detected on Fa0/7, putting
Fa0/7 in err-disable state

Rack1SW1#
%PORT_SECURITY-2-PSECURE_VIOLATION: Security violation occurred, caused
by MAC address 00d0.586e.b930 on port FastEthernet0/7

Rack1SW1#
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/7,
changed state to down
```



```
Rack1SW1#show port-security interface fa0/7
Port Security              : Enabled
Port Status                : Secure-shutdown ← port disabled
Violation Mode             : Shutdown
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 2
Total MAC Addresses        : 0
Configured MAC Addresses   : 0
Sticky MAC Addresses       : 0
Last Source Address        : 00d0.586e.b930
Security Violation Count   : 1

Rack1SW1#show interface status

Port      Name               Status       Vlan     Duplex  Speed Type
Fa0/7                        err-disabled 17        auto   auto 10/100BaseTX
                            ↑    ↑    ↑
                       err-disabled state

Rack1SW1#show errdisable recovery
ErrDisable Reason    Timer Status
-----------------    --------------
udld                 Disabled
bpduguard            Disabled
security-violatio    Disabled
channel-misconfig    Disabled
vmps                 Disabled
pagp-flap            Disabled
dtp-flap             Disabled
link-flap            Disabled
l2ptguard            Disabled
psecure-violation    Enabled
gbic-invalid         Disabled
dhcp-rate-limit      Disabled
unicast-flood        Disabled
storm-control        Disabled
arp-inspection       Disabled
loopback             Disabled

Timer interval: 60 seconds

Interfaces that will be enabled at the next timeout:

Rack1SW1#show mac-address-table vlan 17 | inc Drop|Vlan|--
-----------------------------------------
Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
  17    0030.1369.87a0    STATIC      Drop
```


Task 1.2 Solution

SW2:

```
interface FastEthernet0/2
 storm-control unicast level 3.00
```

Task 1.2 Breakdown

Storm control limits the amount of unicast, multicast, or broadcast traffic that is received in a layer 2 switchport. When the threshold of unicast or broadcast traffic is exceeded, traffic in excess of the threshold is dropped. When the multicast threshold is exceeded, all unicast, multicast, or broadcast traffic above the threshold is dropped. To configure storm-control, issue the **storm-control [unicast | broadcast | multicast] level [level]** interface level command.

Task 1.2 Verification

```
Rack1SW1#show storm-control unicast
Interface  Filter State   Level     Current ← shows real-time level
---------  -------------  -------   -------
Fa0/1      inactive       100.00%   N/A
Fa0/2      Forwarding     3.00%     0.00%
Fa0/3      inactive       100.00%   N/A
```

Pitfall

The storm-control command takes the level argument as a percentage of interface bandwidth. If you are asked to suppress traffic based on an absolute bandwidth level, such as 2Mbps, ensure to take into account whether the interface is running in 10Mbps or 100Mbps mode.

Task 1.3 Solution

SW1:

```
interface FastEthernet0/7
 switchport protected
!
interface FastEthernet0/8
 switchport protected
```

Task 1.3 Breakdown

Port protection prevents hosts that are in the same broadcast domain from directly communicating with each other at layer 2. This feature is especially useful when devices are placed in the same VLAN that would not normally be communicating with each other, such as web servers in a DMZ. Since there is typically not a valid case in which one server would initiate a connection to another server, this feature is very useful.

Task 1.3 Verification

```
Rack1SW1#show interfaces fastEthernet 0/7 switchport  | include
Protected
Protected: true

Rack1SW1#show interfaces fastEthernet 0/8 switchport  | include
Protected
Protected: true
```

Task 1.4 Solution

R4:

```
interface Serial0/0/0.54 point-to-point
 frame-relay interface-dlci 405
  class EEK
!
map-class frame-relay EEK
 frame-relay end-to-end keepalive mode bidirectional
 frame-relay end-to-end keepalive timer send 15
```

R5:

```
interface Serial0/0/0.54 point-to-point
 frame-relay interface-dlci 504
  class EEK
!
map-class frame-relay EEK
 frame-relay end-to-end keepalive mode bidirectional
 frame-relay end-to-end keepalive timer send 15
```

Task 1.4 Breakdown

When problems occur in the provider cloud, the end devices of the Frame Relay cloud may not detect a problem, as LMI communication with the local Frame Relay switch continues without interruption. For this reason, the DLCI may appear to be *active*, however, in reality no user traffic can be sent across the PVC. Frame Relay end-to-end keepalives can be used to detect this problem.

By participating in active request/response polling, Frame Relay end-to-end keepalives behave much like the hello packets in IGP. If a response is not heard back within the configured timer, the DLCI is brought to inactive state.

Task 1.4 Verification

```
Rack1R5#show frame-relay map
Serial0/0/0.54 (up): point-to-point dlci, dlci 504(0x1F8,0x7C80),
broadcast
          status defined, active

Rack1R5#ping 129.1.54.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 129.1.54.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/60 ms

Rack1R5#show frame-relay end-to-end keepalive

End-to-end Keepalive Statistics for Interface Serial0/0/0 (Frame Relay
DTE)

DLCI = 504, DLCI USAGE = LOCAL, VC STATUS = ACTIVE (EEK UP)

SEND SIDE STATISTICS

Send Sequence Number: 20,       Receive Sequence Number: 21
Configured Event Window: 3,     Configured Error Threshold: 2
Total Observed Events: 23,      Total Observed Errors: 0
Monitored Events: 3,            Monitored Errors: 0
Successive Successes: 3,        End-to-end VC Status: UP

RECEIVE SIDE STATISTICS

Send Sequence Number: 20,       Receive Sequence Number: 19
Configured Event Window: 3,     Configured Error Threshold: 2
Total Observed Events: 22,      Total Observed Errors: 0
Monitored Events: 3,            Monitored Errors: 0
Successive Successes: 3,        End-to-end VC Status: UP
```

Task 2.1 Solution

SW3 and SW4:

```
interface Port-channel34
 ip ospf network point-to-point
```


Task 2.1 Breakdown

With an OSPF network type of broadcast, you will see both net link states and summary net link states for the area. Since a network type of point-to-point treats the local network slightly different, it will not have a net link entry for the area. Alternatively, you could also use the network type of point-to-multipoint.

Task 2.1 Verification

Check the output of the "**show**" command before the configuration is applied:

```
Rack1SW4#show ip ospf database | include Net Link States \(Area 34\)
                Net Link States (Area 34)
                Summary Net Link States (Area 34)
```

Check the output of the "**show**" command after the configuration is applied:

```
Rack1SW4#show ip ospf database | include Net Link States \(Area 34\)
                Summary Net Link States (Area 34)
```

Task 2.2 Solution

R1:
```
router bgp 200
 neighbor 129.1.17.7 route-reflector-client
```

R3:
```
router bgp 200
 neighbor 129.1.23.2 route-reflector-client
```

R4:
```
router bgp 100
 neighbor 129.1.46.6 route-reflector-client
```

R5:
```
router bgp 100
 neighbor 129.1.58.8 route-reflector-client
```


Task 2.2 Verification

R3 has R2 as its RR client; R3 has iBGP peering with both R1 and R2. Verify that when routes learned by R3 from R2 are advertised to R1, the Originator ID and the Cluster List attributes are being added:

```
Rack1R3#show ip bgp 205.90.31.0
BGP routing table entry for 205.90.31.0/24, version 20
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  254, (Received from a RR-client)
    129.1.23.2 from 129.1.23.2 (150.1.2.2)
      Origin incomplete, metric 0, localpref 100, valid, internal, best

Rack1R1#show ip bgp 205.90.31.0
BGP routing table entry for 205.90.31.0/24, version 22
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           3
  254
    129.1.23.2 (metric 21024000) from 129.1.13.3 (150.1.3.3)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Originator: 150.1.2.2, Cluster list: 150.1.3.3
```

R5 has SW2 as its RR client; R5 has iBGP peering with both R4 and SW2. Verify that when routes learned by R5 from SW2 are advertised to R4, the Originator ID and the Cluster List attributes are being added:

```
Rack1R5#show ip bgp 220.20.3.0
BGP routing table entry for 220.20.3.0/24, version 38
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
        2
  200 254, (Received from a RR-client)
    129.1.58.8 from 129.1.58.8 (150.1.8.8)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
  200 254
    150.1.4.4 (metric 2) from 150.1.4.4 (150.1.4.4)
      Origin incomplete, metric 0, localpref 100, valid, internal
```



```
Rack1R4#show ip bgp 220.20.3.0
BGP routing table entry for 220.20.3.0/24, version 3
Paths: (3 available, best #3, table Default-IP-Routing-Table)
  Advertised to update-groups:
        1    2    3
  200 254
    129.1.58.8 (metric 2) from 150.1.5.5 (150.1.5.5)
      Origin incomplete, metric 0, localpref 100, valid, internal
      Originator: 150.1.8.8, Cluster list: 150.1.5.5
  200 254
    129.1.124.1 from 129.1.124.1 (150.1.1.1)
      Origin incomplete, localpref 100, valid, external
  200 254
    129.1.124.2 from 129.1.124.2 (150.1.2.2)
      Origin incomplete, localpref 100, valid, external, best

Rack1R1#show ip bgp quote-regexp ^254 | begin Netw
   Network          Next Hop            Metric LocPrf Weight Path
*>i205.90.31.0      129.1.23.2               0    100      0 254 ?
*>i220.20.3.0       129.1.23.2               0    100      0 254 ?
*>i222.22.2.0       129.1.23.2               0    100      0 254 ?

Rack1R1#show ip bgp quote-regexp ^100 | begin Netw
   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   129.1.124.4                           0 100 54 i
* i                 129.1.17.7               0    100      0 100 54 i
*> 28.119.17.0/24   129.1.124.4                           0 100 54 i
* i                 129.1.17.7               0    100      0 100 54 i
*> 112.0.0.0        129.1.124.4                           0 100 54 50 60 i
* i                 129.1.17.7               0    100      0 100 54 50 60 i
*> 113.0.0.0        129.1.124.4                           0 100 54 50 60 i
* i                 129.1.17.7               0    100      0 100 54 50 60 i
*> 114.0.0.0        129.1.124.4                           0 100 54 i
* i                 129.1.17.7               0    100      0 100 54 i
*> 115.0.0.0        129.1.124.4                           0 100 54 i
* i                 129.1.17.7               0    100      0 100 54 i
<output omitted>

Rack1R5#show ip bgp quote-regexp ^54 | begin Netw
   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   129.1.58.8               0    100      0 54 i
* i                 129.1.46.6               0    100      0 54 i
*>i28.119.17.0/24   129.1.58.8               0    100      0 54 i
* i                 129.1.46.6               0    100      0 54 i
*>i112.0.0.0        129.1.58.8               0    100      0 54 50 60 i
* i                 129.1.46.6               0    100      0 54 50 60 i
*>i113.0.0.0        129.1.58.8               0    100      0 54 50 60 i
* i                 129.1.46.6               0    100      0 54 50 60 i
*>i114.0.0.0        129.1.58.8               0    100      0 54 i
* i                 129.1.46.6               0    100      0 54 i
*>i115.0.0.0        129.1.58.8               0    100      0 54 i
* i                 129.1.46.6               0    100      0 54 i
<output omitted>
```



```
Rack1R4#show ip bgp quote-regexp ^200 | beg Netw
   Network          Next Hop            Metric LocPrf Weight Path
* i205.90.31.0      129.1.58.8               0    100      0 200 254 ?
*                   129.1.124.1                           0 200 254 ?
*>                  129.1.124.2                           0 200 254 ?
* i220.20.3.0       129.1.58.8               0    100      0 200 254 ?
*                   129.1.124.1                           0 200 254 ?
*>                  129.1.124.2                           0 200 254 ?
* i222.22.2.0       129.1.58.8               0    100      0 200 254 ?
*                   129.1.124.1                           0 200 254 ?
*>                  129.1.124.2                           0 200 254 ?
```

Task 2.3 Solution

R1:

```
router bgp 200
 network 129.1.17.0 mask 255.255.255.0
```

R3:

```
router bgp 200
 network 129.1.3.0 mask 255.255.255.128
 network 129.1.3.128 mask 255.255.255.128
```

R4:

```
router bgp 100
 network 129.1.45.0 mask 255.255.255.248
 network 129.1.46.0 mask 255.255.255.0
```

SW2:

```
router bgp 100
 network 129.1.58.0 mask 255.255.255.0
```


Task 2.3 Verification

Verify BGP prefix origination

```
Rack1SW2#show ip bgp quote-regexp ^$
BGP table version is 23, local router ID is 150.1.8.8
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
r>i129.1.45.0/29    150.1.4.4                0    100      0 i
r>i129.1.46.0/24    150.1.4.4                0    100      0 i
*> 129.1.58.0/24    0.0.0.0                  0         32768 i

Rack1SW1#show ip bgp quote-regexp ^$
BGP table version is 23, local router ID is 150.1.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
r>i129.1.3.0/25     129.1.13.3               0    100      0 i
r>i129.1.3.128/25   129.1.13.3               0    100      0 i
r>i129.1.17.0/24    129.1.17.1               0    100      0 i

Rack1R3#show ip bgp quote-regexp ^$
BGP table version is 50, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 129.1.3.0/25     0.0.0.0                  0         32768 i
*> 129.1.3.128/25   0.0.0.0                  0         32768 i
r>i129.1.17.0/24    129.1.13.1               0    100      0 i

Rack1R2#show ip bgp quote-regexp ^$
BGP table version is 56, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
r>i129.1.3.0/25     129.1.23.3               0    100      0 i
r>i129.1.3.128/25   129.1.23.3               0    100      0 i
r>i129.1.17.0/24    129.1.13.1               0    100      0 i
```


These devices show RIB failure (r), which is not something to be worried about in this case. Here, it just means that even though the route made it through the best path selection process for BGP, the route was not installed in the routing table. Here, it is due to a better route. In earlier IOS versions, networks with a RIB failure were not advertised to BGP peers, but that is no longer the case. Other items that could cause a RIB failure include memory issues or restrictions on the number of routes.

Task 2.4 Solution

R1:

```
router bgp 200
 neighbor 129.1.124.4 route-map BGP_OUT_TO_R4 out
!
ip prefix-list VLAN_3 seq 5 permit 129.1.3.0/25
!
ip prefix-list VLAN_33 seq 5 permit 129.1.3.128/25
!
route-map BGP_OUT_TO_R4 permit 10
 match ip address prefix-list VLAN_3
 set metric 20
!
route-map BGP_OUT_TO_R4 permit 20
 match ip address prefix-list VLAN_33
 set metric 10
!
route-map BGP_OUT_TO_R4 permit 1000
```

R2:

```
router bgp 200
 neighbor 129.1.124.4 route-map BGP_OUT_TO_R4 out
!
ip prefix-list VLANs_3_&_33 seq 5 permit 129.1.3.0/24 ge 25 le 25
!
route-map BGP_OUT_TO_R4 deny 10
 match ip address prefix-list VLANs_3_&_33
!
route-map BGP_OUT_TO_R4 permit 1000
```

SW1:

```
router bgp 200
 neighbor 129.1.78.8 route-map BGP_OUT_TO_SW2 out
!
ip prefix-list VLAN_3 seq 5 permit 129.1.3.0/25
!
ip prefix-list VLAN_33 seq 5 permit 129.1.3.128/25
!
route-map BGP_OUT_TO_SW2 permit 10
 match ip address prefix-list VLAN_3
 set metric 10
!
route-map BGP_OUT_TO_SW2 permit 20
 match ip address prefix-list VLAN_33
 set metric 20
!
route-map BGP_OUT_TO_SW2 permit 1000
```


Task 2.4 Breakdown

Recall how to influence the BGP best path selection process:

Attribute	Direction Applied	Traffic Flow Affected
Weight	Inbound	Outbound
Local-Preference	Inbound	Outbound
AS-Path	Outbound	Inbound
MED	Outbound	Inbound

In the above task, traffic engineering is applied on traffic destined for VLANs 3 and 33. AS 200 wants to affect how traffic is entering its AS that is destined for these VLANs. In order to effect an inbound traffic flow, either the MED or AS-Path attributes should be modified on outbound BGP updates. In the above solutions, MED has been used to influence the selection path. However, AS-Path could have been used in the same manner.

Traffic for VLAN 3 is preferred to come in the link between SW1 and SW2. This has been accomplished by advertising VLAN 3 with a more preferable (lower) MED value to SW2 than that which has been advertised to R4.

Additionally, traffic for VLAN 33 has a preferred entry point of the link between R1 and R4. This has been similarly accomplished by advertising VLAN 33 with a more preferable (lower) MED value to R4 than that which has been advertised to SW2.

Lastly, this requirement states that the link between R2 and R4 can not be used by AS 100 to get to VLAN 3 or VLAN 33. This is simply accomplished by filtering the advertisement of these networks from R2 to R4. Specifically, this has been configured by creating a prefix-list which matches both VLAN 3 and 33. Next, a route-map is configured that will be applied outbound from R2 to R4. The first sequence of the route-map is a deny sequence in which the previously created prefix-list is matched. This effectively stops the advertisement of VLANs 3 and 33 to R4.

Pitfall

When changing BGP attributes through a route-map, don't forget to add an explicit permit sequence of the route-map at the end. If you leave the explicit permit out, all other prefixes not matched in the route-map will be denied.


```
Rack1R4#show ip bgp
BGP table version is 19, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   129.1.46.6                100      0 54 i
*>i28.119.17.0/24   129.1.46.6                100      0 54 i
*>i112.0.0.0        129.1.46.6             0  100      0 54 50 60 i
*>i113.0.0.0        129.1.46.6             0  100      0 54 50 60 i
*>i114.0.0.0        129.1.46.6             0  100      0 54 i
*>i115.0.0.0        129.1.46.6             0  100      0 54 i
*>i116.0.0.0        129.1.46.6             0  100      0 54 i
*>i117.0.0.0        129.1.46.6             0  100      0 54 i
*>i118.0.0.0        129.1.46.6             0  100      0 54 i
*>i119.0.0.0        129.1.46.6             0  100      0 54 i

The > denotes the best path                          1. weight both 0
 ↓                                                          ↓
*>i129.1.3.0/25     129.1.58.8               10     100      0 200 i
*                   129.1.124.1              20              0 200 i
<snip>

Rack1R4#show ip bgp 129.1.3.0 255.255.255.128
BGP routing table entry for 129.1.3.0/25, version 19
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to non peer-group peers:
  129.1.46.6 129.1.124.1 129.1.124.2
  200 ← 3. AS-Path both 1 AS long
    129.1.58.8 (metric 74) from 150.1.5.5 (150.1.5.5)
4. Origin both IGP  5. MED is tiebreaker  ↙ 2. local-preference both
100
      Origin IGP, metric 10, localpref 100, valid, internal, best
      Originator: 150.1.8.8, Cluster list: 150.1.5.5
  200 ← 3. AS-Path both 1 AS long
```


Task 2.4 Verification

Verify that traffic towards VLAN3 prefers the Ethernet segment between SW1 and SW2:

```
Rack1R4#show ip bgp 129.1.3.0 255.255.255.128 bestpath
BGP routing table entry for 129.1.3.0/25, version 48
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
        1    2
  200
    129.1.58.8 (metric 2) from 150.1.5.5 (150.1.5.5)
      Origin IGP, metric 10, localpref 100, valid, internal, best
      Originator: 150.1.8.8, Cluster list: 150.1.5.5

Rack1SW2#show ip bgp 129.1.3.0 255.255.255.128 bestpath
BGP routing table entry for 129.1.3.0/25, version 24
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1          3
  200
    129.1.78.7 from 129.1.78.7 (150.1.7.7)
      Origin IGP, metric 10, localpref 100, valid, external, best
```

Verify that traffic towards VLAN33 prefers the Frame-relay segment between R1 and R4:

```
Rack1SW2#show ip bgp 129.1.3.128 255.255.255.128 bestpath
BGP routing table entry for 129.1.3.128/25, version 26
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1          2
  200
    150.1.4.4 (metric 3) from 129.1.58.5 (150.1.5.5)
      Origin IGP, metric 10, localpref 100, valid, internal, best
      Originator: 150.1.4.4, Cluster list: 150.1.5.5

Rack1R4#show ip bgp 129.1.3.128 255.255.255.128 bestpath
BGP routing table entry for 129.1.3.128/25, version 49
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
        1    2    3
  200
    129.1.124.1 from 129.1.124.1 (150.1.1.1)
      Origin IGP, metric 10, localpref 100, valid, external, best
```


Verify that R2 does NOT advertise to R4 VLAN3 and VLAN33 NLRI:

```
Rack1R2#show ip bgp neighbors 129.1.124.4 advertised-routes
BGP table version is 56, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
r>i129.1.17.0/24    129.1.13.1               0    100      0 i
*> 205.90.31.0      192.10.1.254             0             0 254 ?
*> 220.20.3.0       192.10.1.254             0             0 254 ?
*> 222.22.2.0       192.10.1.254             0             0 254 ?

Total number of prefixes 4
```

Task 2.5 Solution

R1:

```
ip as-path access-list 1 permit ^254$
!
route-map BGP_OUT_TO_R4 deny 30
 match as-path 1
```

SW1:

```
ip as-path access-list 1 permit ^254$
!
route-map BGP_OUT_TO_SW2 deny 30
 match as-path 1
```


Task 2.5 Breakdown

By filtering the advertisement of prefixes learned from AS 254 to AS 100, AS 100 is forced to use the path between R2 and R4 to reach these prefixes. This has been accomplished by creating an AS-Path access-list which matches prefixes that are from AS 254. Next, this AS-Path access-list is added to a new deny sequence of the route-map previously defined on R1 and SW1.

Task 2 5 Verification

```
Rack1R4#show ip bgp quote-regexp _254_ | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      129.1.124.2                          0 200 254 ?
*> 220.20.3.0       129.1.124.2                          0 200 254 ?
*> 222.22.2.0       129.1.124.2                          0 200 254
```

Task 2.5 Verification

Before applying the configuration, verify that R4 receives NLRI for AS 254 from the following devices in AS 200: R1, R2 and SW1:

```
Rack1R4#show ip bgp quote-regexp _254
BGP table version is 49, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i205.90.31.0      129.1.58.8               0    100      0 200 254 ?
*                   129.1.124.1                           0 200 254 ?
*>                  129.1.124.2                           0 200 254 ?
* i220.20.3.0       129.1.58.8               0    100      0 200 254 ?
*                   129.1.124.1                           0 200 254 ?
*>                  129.1.124.2                           0 200 254 ?
* i222.22.2.0       129.1.58.8               0    100      0 200 254 ?
*                   129.1.124.1                           0 200 254 ?
*>                  129.1.124.2                           0 200 254 ?
```

After, only R2 advertised NLRI for AS 254 into AS 100:

```
Rack1R4#show ip bgp quote-regexp _254_ | begin Network
   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      129.1.124.2                          0 200 254 ?
*> 220.20.3.0       129.1.124.2                          0 200 254 ?
*> 222.22.2.0       129.1.124.2                          0 200 254 ?
```


Task 2.6 Solution

R4:

```
router bgp 100
 neighbor 129.1.124.1 default-originate
 neighbor 129.1.124.2 default-originate
```

SW2:

```
router bgp 100
 neighbor 129.1.78.7 default-originate
```

Task 2.6 Verification

```
Rack1SW1#show ip bgp 0.0.0.0
BGP routing table entry for 0.0.0.0/0, version 27
Paths: (2 available, best #1, table Default-IP-Routing-Table)
Flag: 0x1860
  Advertised to update-groups:
     2
  100
    129.1.78.8 from 129.1.78.8 (150.1.8.8)
      Origin IGP, localpref 100, valid, external, best
  100
    129.1.17.1 from 129.1.17.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal

Rack1R1#show ip bgp 0.0.0.0
BGP routing table entry for 0.0.0.0/0, version 56
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
     2          3
  100, (Received from a RR-client)
    129.1.17.7 from 129.1.17.7 (150.1.7.7)
      Origin IGP, metric 0, localpref 100, valid, internal
  100
    129.1.124.4 from 129.1.124.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, external, best

Rack1R2#show ip bgp 0.0.0.0
BGP routing table entry for 0.0.0.0/0, version 58
Paths: (2 available, best #1, table Default-IP-Routing-Table)
Flag: 0x960
  Advertised to update-groups:
     1          3
  100
    129.1.124.4 from 129.1.124.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, external, best
  100
    129.1.13.1 (metric 21024000) from 129.1.23.3 (150.1.3.3)
      Origin IGP, metric 0, localpref 100, valid, internal
      Originator: 150.1.1.1, Cluster list: 150.1.3.3
```


Task 2.7 Solution

SW1:

```
router bgp 200
 neighbor 129.1.78.8 route-map BGP_IN_FROM_SW2 in
!
ip prefix-list DEFAULT seq 5 permit 0.0.0.0/0
!
route-map BGP_IN_FROM_SW2 permit 10
 match ip address prefix-list DEFAULT
 set local-preference 200
```

Task 2.7 Breakdown

In the above task, it is asked that SW1 be configured as the most preferable default exit point from AS 200. Since it is also stated that this configuration must be done on SW1, either local-preference or weight are candidates to affect the BGP best path selection. However, as weight is only locally significant, it is not a valid attribute to impact how the entire AS chooses the best path. Therefore, local-preference must be used to affect the selection.

In the above configuration, an IP prefix-list has been created which matches a default route. Next, a route-map is created that matches this prefix-list and sets the local-preference. As the default local-preference value is 100, any value above 100 would accomplish the desired goal.

Task 2.7 Verification

Verify that SW1 filters any NLRI received from SW2 except the default route

```
Rack1SW1#show ip bgp neighbors 129.1.78.8 routes
BGP table version is 39, local router ID is 150.1.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 0.0.0.0          129.1.78.8               0    200      0 100 i

Total number of prefixes 1
```



```
Rack1R1#show ip bgp
BGP table version is 75, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i0.0.0.0          129.1.17.7               0    100      0 100 i
*>                  129.1.124.4              0             0 100 i
<output omitted>
```

Verify that R1 and R2 install a default route into RT through SW1:

```
Rack1R1#show ip route 0.0.0.0
Routing entry for 0.0.0.0/0, supernet
  Known via "bgp 200", distance 200, metric 0, candidate default path
  Tag 100, type internal
  Last update from 129.1.17.7 00:01:08 ago
  Routing Descriptor Blocks:
  * 129.1.17.7, from 129.1.17.7, 00:01:08 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
      Route tag 100
```

Shutdown the link to SW2 and verify the default routing again:

```
Rack1R1#show ip route 0.0.0.0
Routing entry for 0.0.0.0/0, supernet
  Known via "bgp 200", distance 20, metric 0, candidate default path
  Tag 100, type external
  Last update from 129.1.124.4 00:00:36 ago
  Routing Descriptor Blocks:
  * 129.1.124.4, from 129.1.124.4, 00:00:36 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
```

Task 2.8 Solution

R2:

```
ip as-path access-list 1 permit ^100(_[0-9]+)?$
!
router bgp 200
 neighbor 129.1.124.4 filter-list 1 in
```

Task 2.8 Breakdown

Recall the special characters used in regular expressions:

Character	Meaning
^	Start of string

$	End of string
[]	Range of characters
-	Used to specify range (i.e. [0-9])
( )	Logical grouping
.	Any single character
*	Zero or more instances
+	One or more instance
?	Zero or one instance
_ (underscore)	Comma, open or close brace, open or close parentheses, start or end of string, or space

The above task requires that R2 only accept prefixes that have been originated in its directly connected provider's AS, as well as the provider's directly connected customers. This is a common view of the BGP table to take, since it is usually a safe assumption that your provider will have the best path to a destination if they are directly peering with that destination's AS.

The easiest way to create a regular expression is to think logically about what you are first try to match, and to write out all possibilities of these matches. For example, R2's directly connected AS is AS 100. Therefore, we can assume that there may be paths that have been originated inside AS 100. This is the first possibility we must match:

^100$

The ^ means that the path begins, the 100 matches AS 100, and the $ means that the path ends.

Next, be must also match the condition in which prefixes are originated from AS 100's directly connected ASs. However, we do not know which explicit AS numbers these are. Therefore, for the time being we will use the placeholder X. The second possibility is therefore as follows:

^100_X$

The ^ means that the path begins, the 100 matches AS 100, the _ matches a space, the X is our place holder for any single AS, and the $ means that the path ends.

Next let's reason out what X can represent. Since X is only one single AS, there will be no spaces, commas, parentheses, or any other special type characters. In other words, X must be a combination of integers. However, since we don't know what the exact path is, we must take into account that X may be more than one integer (i.e. 10 is two integers, 123 is three integers). The character used to match one or more instances is the plus sign. Therefore our second path is now:

^100_X+$

Where X is any single integer. Next we should define X. Again since we do not know what specific number or combination of numbers X will be, we can reason that it can be any combination of any number from zero to nine. This can be denoted as a the range from 0 to 9 by using brackets. Therefore our second choice is now:

^100_[0-9]+$

This will match all of AS 100's directly connected customers. Now we can stop where we are, and list both of these combinations in an as-path access-list, or we can try to combine them into one single line. To combine them, first let us compare what is different between them.

^100$
^100_[0-9]+$

From looking at the expressions, it is evident that the sequence _[0-9]+ is the difference. For the time being let us represent this sequence with the variable A. In the first case, A does not exist in the expression. In the second case, A does exist in the expression. In other words, A is either true or false. True or false (0 or 1) is represented by the character ?

Therefore we can reduce our expression to:

^100A?$

However, if we simply write the expression as ^100_[0-9]+?$, the question mark will apply to the plus sign. Instead, we want the question mark to apply to the string _[0-9]+ as a whole. Therefore, this string can be grouped together using parentheses. Parentheses are used in regular expressions as simply a logical grouping. Therefore, our final expression reduces to:

^100(_[0-9]+)?$

In order to meet the requirement of still being eligible as a default exit point, make sure to verify that the policy does not block the default 0.0.0.0 route from R4.

Note

To match a question mark in IOS, the escape sequence CTRL-V or ESC-Q must be entered first.

Task 2.8 Verification

```
Rack1R2#show ip bgp neighbors 129.1.124.4 routes
BGP table version is 106, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  0.0.0.0          129.1.124.4              0             0 100 i
*> 28.119.16.0/24   129.1.124.4                            0 100 54 i
*> 28.119.17.0/24   129.1.124.4                            0 100 54 i
*> 114.0.0.0        129.1.124.4                            0 100 54 i
*> 115.0.0.0        129.1.124.4                            0 100 54 i
*> 116.0.0.0        129.1.124.4                            0 100 54 i
*> 117.0.0.0        129.1.124.4                            0 100 54 i
*> 118.0.0.0        129.1.124.4                            0 100 54 i
*> 119.0.0.0        129.1.124.4                            0 100 54 i
*> 129.1.45.0/29    129.1.124.4              0             0 100 i
*> 129.1.46.0/24    129.1.124.4              0             0 100 i
*> 129.1.58.0/24    129.1.124.4                            0 100 i
```

Verify paths for non-direct customers of AS100:

```
Rack1R2#show ip bgp quote-regexp ^100_[0-9]+(_[0-9]+)+$
BGP table version is 106, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop        Metric LocPrf Weight Path
*>i112.0.0.0        129.1.13.1           0    100      0 100 54 50 60 i
*>i113.0.0.0        129.1.13.1           0    100      0 100 54 50 60 i
```

Task 2.9 Solution

R1:

```
ip prefix-list DEFAULT seq 5 permit 0.0.0.0/0
!
route-map BGP_IN_FROM_R4 permit 10
 match ip address prefix-list DEFAULT
 set local-preference 50
!
route-map BGP_IN_FROM_R4 permit 1000
!
router bgp 200
  neighbor 129.1.124.4 route-map BGP_IN_FROM_R4 in
```


Task 2.9 Breakdown

Similar to task 6.17, the local-preference of the default route learned from AS 100 has been modified in order to affect how traffic leaves AS 200. In this case, R1 is configured as the least preferred exit point by setting the local-preference lower than the other two values of 100 and 200.

Task 2.9 Verification

Verify the default routing in AS200. Look for the most preferred default route when all links to AS100 are up:

```
Rack1R3#show ip bgp 0.0.0.0
BGP routing table entry for 0.0.0.0/0, version 132
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     2
  100
    129.1.17.7 (metric 20514560) from 129.1.13.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 200, valid, internal, best
      Originator: 150.1.7.7, Cluster list: 150.1.1.1
```

Next, shutdown the link between SW1 and SW2. Then, verify the BGP default route again:

```
Rack1R3#show ip bgp 0.0.0.0
BGP routing table entry for 0.0.0.0/0, version 134
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Flag: 0x840
  Advertised to update-groups:
     1
  100, (Received from a RR-client)
    129.1.23.2 from 129.1.23.2 (150.1.2.2)
      Origin IGP, metric 0, localpref 100, valid, internal, best
```

Finally, shut down the serial interface on R2 and verify the BGP routes again:

```
Rack1R3#show ip bgp 0.0.0.0
BGP routing table entry for 0.0.0.0/0, version 160
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
     2
  100
    129.1.13.1 from 129.1.13.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 50, valid, internal, best
```


Task 2.10 Solution

R2:

```
router bgp 200
 aggregate-address 129.1.0.0 255.255.0.0
 aggregate-address 150.1.0.0 255.255.240.0
 neighbor 129.1.23.3 route-map BGP_OUT_TO_R3 out
!
ip prefix-list AGGREGATE seq 5 permit 129.1.0.0/16
ip prefix-list AGGREGATE seq 10 permit 150.1.0.0/20
!
route-map BGP_OUT_TO_R4 deny 20
 match ip address prefix-list AGGREGATE
!
route-map BGP_OUT_TO_R3 deny 10
 match ip address prefix-list AGGREGATE
!
route-map BGP_OUT_TO_R3 permit 1000
```

R6:

```
router bgp 100
 aggregate-address 129.1.0.0 255.255.0.0
 aggregate-address 150.1.0.0 255.255.240.0
 neighbor 129.1.46.4 route-map BGP_OUT_TO_R4 out
!
ip prefix-list AGGREGATE seq 5 permit 129.1.0.0/16
ip prefix-list AGGREGATE seq 10 permit 150.1.0.0/20
!
route-map BGP_OUT_TO_R4 deny 10
 match ip address prefix-list AGGREGATE
!
route-map BGP_OUT_TO_R4 permit 1000
```

SW2:

```
router bgp 100
 aggregate-address 129.1.0.0 255.255.0.0
 aggregate-address 150.1.0.0 255.255.240.0
 neighbor 129.1.78.7 route-map BGP_OUT out
 neighbor 129.1.58.5 route-map BGP_OUT out
!
ip prefix-list AGGREGATE seq 5 permit 129.1.0.0/16
ip prefix-list AGGREGATE seq 10 permit 150.1.0.0/20
!
route-map BGP_OUT deny 10
 match ip address prefix-list AGGREGATE
!
route-map BGP_OUT permit 1000
```


Task 2.10 Breakdown

The above task illustrates a straightforward aggregation configuration, in which the border routers of the network are advertising an aggregate block of the internal address space to the backbones. In addition to this, the aggregate block is denied from being advertised to the internal routers by matching it in a prefix-list, and denying it in a route-map applied to the iBGP neighbors.

Task 2.10 Verification

Verify the summary prefix generation. For example on SW2:

```
Rack1SW2#show ip bgp 129.1.0.0
BGP routing table entry for 129.1.0.0/16, version 59
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     2
  Local, (aggregated by 100 150.1.8.8)
    0.0.0.0 from 0.0.0.0 (150.1.8.8)
      Origin IGP, localpref 100, weight 32768, valid, aggregated,
local, atomic-aggregate, best
```

Confirm that SW2 does not send summary to internal routers:

```
Rack1SW2#show ip bgp neigh 129.1.58.5 advertised-routes | inc 129.1.0.0
Rack1SW2#

Rack1SW2#show ip bgp neigh 129.1.78.7 advertised-routes | inc 129.1.0.0
Rack1SW2#
```

Task 3.1 Solution

R1, R2, R3, R4 and R6:

```
ipv6 unicast-routing
```

R1:

```
interface FastEthernet0/0
 ipv6 address 2001:CC1E:1:1::1/64
```

R2:

```
interface Serial0/1
 ipv6 address 2001:CC1E:1:23::2/64
```

R3:

```
interface FastEthernet0/0
 ipv6 address 2001:CC1E:1:3::3/64
!
interface Serial1/3
 ipv6 address 2001:CC1E:1:23::3/64
```


R4:
```
interface FastEthernet0/1
 ipv6 address 2001:CC1E:1:46::4/64
```

R6:
```
interface FastEthernet0/0
 ipv6 address 2001:CC1E:1:46::6/64
```

Task 3.1 Verification

```
Rack1R1#show ipv6 interface brief fastEthernet 0/0
FastEthernet0/0            [up/up]
    FE80::213:80FF:FEAD:7A80
    2001:CC1E:1:1::1

Rack1R2#show ipv6 interface brief serial 0/1
Serial0/1                  [up/up]
    FE80::211:20FF:FEFD:6E00
    2001:CC1E:1:23::2

Rack1R3#show ipv6 interface brief fastEthernet 0/0
FastEthernet0/0            [up/up]
    FE80::211:20FF:FE93:E560
    2001:CC1E:1:3::3

Rack1R3#show ipv6 interface brief serial 1/3
Serial1/3                  [up/up]
    FE80::211:20FF:FE93:E560
    2001:CC1E:1:23::3

Rack1R4#show ipv6 interface brief fastEthernet 0/1
FastEthernet0/1            [up/up]
    FE80::215:62FF:FE41:FD9D
    2001:CC1E:1:46::4

Rack1R6#show ipv6 interface brief fastEthernet 0/0
FastEthernet0/0            [up/up]
    FE80::21A:2FFF:FE78:4678
    2001:CC1E:1:46::6
```


Task 3.2 Solution

R1:

```
interface Serial0/0
 ipv6 address 2001:CC1E:1:124::1/64
 ipv6 address FE80::1 link-local
 frame-relay map ipv6 FE80::2 104
 frame-relay map ipv6 FE80::4 104 broadcast
 frame-relay map ipv6 2001:CC1E:1:124::2 104
 frame-relay map ipv6 2001:CC1E:1:124::4 104
```

R2:

```
interface Serial0/0
 ipv6 address 2001:CC1E:1:124::2/64
 ipv6 address FE80::2 link-local
 frame-relay map ipv6 FE80::4 204 broadcast
 frame-relay map ipv6 2001:CC1E:1:124::1 204
 frame-relay map ipv6 2001:CC1E:1:124::4 204
 frame-relay map ipv6 FE80::1 204
```

R4:

```
interface Serial0/0/0.124 multipoint
 ipv6 address 2001:CC1E:1:124::4/64
 ipv6 address FE80::4 link-local
 frame-relay map ipv6 FE80::2 402 broadcast
 frame-relay map ipv6 2001:CC1E:1:124::1 401
 frame-relay map ipv6 2001:CC1E:1:124::2 402
 frame-relay map ipv6 FE80::1 401 broadcast
```


Task 3.2 Verification

```
Rack1R4#show frame-relay map 401
Serial0/0/0.124 (up): ipv6 FE80::1 dlci 401(0x191,0x6410), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0.124 (up): ipv6 2001:CC1E:1:124::1 dlci 401(0x191,0x6410),
static,
              CISCO, status defined, active
Serial0/0/0.124 (up): ip 129.1.124.1 dlci 401(0x191,0x6410), static,
              broadcast,
              CISCO, status defined, active

Rack1R4#show frame-relay map 402
Serial0/0/0.124 (up): ipv6 2001:CC1E:1:124::2 dlci 402(0x192,0x6420),
static,
              CISCO, status defined, active
Serial0/0/0.124 (up): ipv6 FE80::2 dlci 402(0x192,0x6420), static,
              broadcast,
              CISCO, status defined, active
Serial0/0/0.124 (up): ip 129.1.124.2 dlci 402(0x192,0x6420), static,
              broadcast,
              CISCO, status defined, active

Rack1R2#show frame-relay map
Serial0/0 (up): ipv6 FE80::4 dlci 204(0xCC,0x30C0), static,
              broadcast,
              CISCO, status defined, active
Serial0/0 (up): ip 129.1.124.4 dlci 204(0xCC,0x30C0), static,
              broadcast,
              CISCO, status defined, active
Serial0/0 (up): ipv6 2001:CC1E:1:124::1 dlci 204(0xCC,0x30C0), static,
              CISCO, status defined, active
Serial0/0 (up): ipv6 2001:CC1E:1:124::4 dlci 204(0xCC,0x30C0), static,
              CISCO, status defined, active
Serial0/0 (up): ipv6 FE80::1 dlci 204(0xCC,0x30C0), static,
              CISCO, status defined, active
Serial0/0 (up): ip 129.1.124.1 dlci 204(0xCC,0x30C0), static,
              CISCO, status defined, active

Rack1R1#show frame-relay map
Serial0/0 (up): ipv6 FE80::2 dlci 104(0x68,0x1880), static,
              CISCO, status defined, active
Serial0/0 (up): ip 129.1.124.2 dlci 104(0x68,0x1880), static,
              CISCO, status defined, active
Serial0/0 (up): ipv6 FE80::4 dlci 104(0x68,0x1880), static,
              broadcast,
              CISCO, status defined, active
Serial0/0 (up): ip 129.1.124.4 dlci 104(0x68,0x1880), static,
              broadcast,
              CISCO, status defined, active
Serial0/0 (up): ipv6 2001:CC1E:1:124::2 dlci 104(0x68,0x1880), static,
              CISCO, status defined, active
Serial0/0 (up): ipv6 2001:CC1E:1:124::4 dlci 104(0x68,0x1880), static,
              CISCO, status defined, active
```


Test basic connectivity:

```
Rack1R1#ping 2001:CC1E:1:124::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:124::2, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/112/112
ms

Rack1R1#ping 2001:CC1E:1:124::4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:124::4, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/32 ms

Rack1R4#ping ipv6 2001:CC1E:1:46::6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:46::6, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/1/4 ms

Rack1R2#ping 2001:CC1E:1:23::3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:23::3, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/29/32 ms
```


Task 3.3 Solution

R4:

```
ipv6 router eigrp 46
 no shut
!
interface fastEtherent 0/1
 ipv6 eigrp 46
!
! This prefix list is to be used later, in this section
!
ipv6 prefix-list TEST permit 0::0/0 le 64
```

R6:

```
interface fastEthernet 0/0
 ipv6 eigrp 46
!
interface loopback601
 ipv6 address 2001:205:90:31::1/48
 ipv6 eigrp 46
!
interface loopback602
 ipv6 address 2001:220:20:3::1/64
 ipv6 eigrp 46
!
interface loopback603
 ipv6 address 2001:222:22:2::1/80
 ipv6 eigrp 46
!
ipv6 router eigrp 46
 no shut
```


Task 3.3 Verification

```
Rack1R4#show ipv6 route eigrp
IPv6 Routing Table - Default - 8 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
D   2001:205:90::/48 [90/156160]
     via FE80::21A:2FFF:FE78:4678, FastEthernet0/1
D   2001:220:20:3::/64 [90/156160]
     via FE80::21A:2FFF:FE78:4678, FastEthernet0/1
D   2001:222:22:2::/80 [90/156160]
     via FE80::21A:2FFF:FE78:4678, FastEthernet0/1
```

For now, we will just configure the prefix list since there is not currently any advertisements going to R2 or R3. Note: Some IOS versions may be missing part of the context sensitive help for the command. Try typing in the entire command.

```
Rack1R4#ping 2001:205:90:31::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:205:90:31::1, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/4 ms

Rack1R4#ping 2001:220:20:3::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:220:20:3::1, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/4 ms

Rack1R4#ping 2001:222:22:2::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:222:22:2::1, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/0 ms
```


Task 3.4 Solution

R4:

```
interface serial 0/0/0.124
 ipv6 ospf 1 area 0
 ipv6 ospf network point-to-multipoint
```

R2:

```
interface Serial0/1
ipv6 ospf 1 area 0
ipv6 ospf network point-to-point

interface Serial0/0
ipv6 ospf 1 area 0
ipv6 ospf network point-to-multipoint
```

R1:

```
interface Serial0/0
ipv6 ospf 1 area 0
ipv6 ospf network point-to-multipoint

interface FastEthernet0/0
ipv6 ospf 1 area 0
```

R3:

```
interface Serial1/3
 ipv6 ospf 1 area 0
 ipv6 ospf network point-to-point

interface FastEthernet0/0
 ipv6 ospf 1 area 0
```


Task 3.4 Verification

Verify OSPFv3 neighbors and routes:

```
Rack1R4#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
150.1.1.1         1   FULL/  -        00:01:34    5               Serial0/0/0.124
150.1.2.2         1   FULL/  -        00:01:46    5               Serial0/0/0.124

Rack1R4#show ipv6 route ospf
IPv6 Routing Table - Default - 12 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:CC1E:1:1::/64 [110/65]
     via FE80::1, Serial0/0/0.124
O   2001:CC1E:1:3::/64 [110/129]
     via FE80::2, Serial0/0/0.124
O   2001:CC1E:1:23::/64 [110/128]
     via FE80::2, Serial0/0/0.124
O   2001:CC1E:1:124::1/128 [110/64]
     via FE80::1, Serial0/0/0.124
O   2001:CC1E:1:124::2/128 [110/64]
     via FE80::2, Serial0/0/0.124

Rack1R2#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
150.1.4.4         1   FULL/  -        00:01:36    15              Serial0/0
150.1.3.3         1   FULL/  -        00:00:39    9               Serial0/1

Rack1R2#show ipv6 route ospf
IPv6 Routing Table - 10 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:CC1E:1:1::/64 [110/129]
     via FE80::4, Serial0/0
O   2001:CC1E:1:3::/64 [110/65]
     via FE80::211:20FF:FE93:E560, Serial0/1
O   2001:CC1E:1:124::1/128 [110/128]
     via FE80::4, Serial0/0
O   2001:CC1E:1:124::4/128 [110/64]
     via FE80::4, Serial0/0
```


Task 3.5 Solution

R4:

```
!
ipv6 router eigrp 46
 redistribute ospf 1 include-connected

 default-metric 10000 10 255 1 1500
!
ipv6 router ospf 1
 redistribute eigrp 46 route-map NO65 include-connected
!
route-map NO65
 match ipv6 address prefix TEST
!
interface FastEthernet0/1
 ipv6 summary-address eigrp 46 2001:222:22:2::/64
```

Task 3.5 Verification

Look at your routing tables on R6 and R3, and verify that both show all the networks. To restrict to prefixes with a mask of 64 bits or less, you can add the prefix list configured earlier to a route map with the redistribution. In order to still have reachability to the loopback on R6, a summary needs to be configured with a mask length less than 64 bits.

Check that R4 OSPF learned routes are re-distributed into EIGRP:

```
Rack1R4#show ipv6 route ospf
IPv6 Routing Table - Default - 14 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:CC1E:1:1::/64 [110/65]
     via FE80::1, Serial0/0/0.124
O   2001:CC1E:1:3::/64 [110/129]
     via FE80::2, Serial0/0/0.124
O   2001:CC1E:1:23::/64 [110/128]
     via FE80::2, Serial0/0/0.124
O   2001:CC1E:1:124::1/128 [110/64]
     via FE80::1, Serial0/0/0.124
O   2001:CC1E:1:124::2/128 [110/64]
     via FE80::2, Serial0/0/0.124

Rack1R6#show ipv6 route eigrp
IPv6 Routing Table - Default - 16 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
D   2001:222:22:2::/64 [90/158720]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0
EX  2001:CC1E:1:1::/64 [170/261120]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0
EX  2001:CC1E:1:3::/64 [170/261120]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0
EX  2001:CC1E:1:23::/64 [170/261120]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0
EX  2001:CC1E:1:124::/64 [170/261120]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0
EX  2001:CC1E:1:124::1/128 [170/261120]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0
EX  2001:CC1E:1:124::2/128 [170/261120]
     via FE80::215:62FF:FE41:FD9D, FastEthernet0/0
```


Check that R4 EIGRP learned routes are re-distributed into OSPF:

```
Rack1R4#show ipv6 route eigrp
IPv6 Routing Table - Default - 14 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
D   2001:205:90::/48 [90/156160]
     via FE80::21A:2FFF:FE78:4678, FastEthernet0/1
D   2001:220:20:3::/64 [90/156160]
     via FE80::21A:2FFF:FE78:4678, FastEthernet0/1
D   2001:222:22:2::/64 [5/156160]
     via Null0, directly connected
D   2001:222:22:2::/80 [90/156160]
     via FE80::21A:2FFF:FE78:4678, FastEthernet0/1
```

Also see bellow that the /80 prefix is NOT advertised as per task 3.3 requirements:

```
Rack1R1#show ipv6 route ospf
IPv6 Routing Table - 14 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
OE2  2001:205:90::/48 [110/20]
     via FE80::4, Serial0/0
OE2  2001:220:20:3::/64 [110/20]
     via FE80::4, Serial0/0
OE2  2001:222:22:2::/64 [110/20]
     via FE80::4, Serial0/0
O   2001:CC1E:1:3::/64 [110/193]
     via FE80::4, Serial0/0
O   2001:CC1E:1:23::/64 [110/192]
     via FE80::4, Serial0/0
OE2  2001:CC1E:1:46::/64 [110/20]
     via FE80::4, Serial0/0
O   2001:CC1E:1:124::2/128 [110/128]
     via FE80::4, Serial0/0
O   2001:CC1E:1:124::4/128 [110/64]
     via FE80::4, Serial0/0
```


Task 4.1 Solution

R6:

```
!
interface FastEthernet0/1
 no shutdown
```

SW4:

```
sdm prefer extended-match

ip vrf TEST
 rd 44:44

interface FastEthernet0/6
 ip vrf forwarding TEST
 no switchport
 ip address 10.0.0.10 255.255.255.0


router ospf 129 vrf TEST
 network 10.0.0.10 0.0.0.0 area 0
```


Task 4.1 Breakdown

Configuring a VRF on the switch may require a change to the SDM profile for 3550 switches.

Task 4.1 Verification

```
Rack1SW4#show ip vrf TEST
  Name                              Default RD          Interfaces
  TEST                              44:44               Fa0/6
Rack1SW4#

Rack1R6#ping vrf VPNB 10.0.0.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.10, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
Rack1R6#
```

Task 4.2 Solution

R6:

```
interface FastEthernet0/0
 mpls ip
```

R4:

```
Interface Serial0/0/0.54
mpls ip
!
Interface FastEthernet0/0
 mpls ip
!
interface FastEthernet0/1
 mpls ip
```

R5:

```
interface Serial0/0/0.54
 mpls ip
!
interface FastEthernet0/1
 mpls ip
```


Task 4.2 Breakdown

LDP is the default label protocol, so all that is needed is to enable MPLS on the interfaces.

Task 4.2 Verification

Verify that the neighbor adjacencies form, and check the output of "**show mpls ldp neighbor**" and "**show mpls ldp discovery**".

```
Rack1R4#show mpls ldp neighbor
    Peer LDP Ident: 150.1.6.6:0; Local LDP Ident 150.1.4.4:0
        TCP connection: 150.1.6.6.24124 - 150.1.4.4.646
        State: Oper; Msgs sent/rcvd: 16/16; Downstream
        Up time: 00:01:13
        LDP discovery sources:
          FastEthernet0/1, Src IP addr: 129.1.46.6
        Addresses bound to peer LDP Ident:
          129.1.46.6      54.1.1.6        150.1.6.6
    Peer LDP Ident: 150.1.5.5:0; Local LDP Ident 150.1.4.4:0
        TCP connection: 150.1.5.5.25621 - 150.1.4.4.646
        State: Oper; Msgs sent/rcvd: 16/15; Downstream
        Up time: 00:00:59
        LDP discovery sources:
          Serial0/0/0.54, Src IP addr: 129.1.54.5
          FastEthernet0/0, Src IP addr: 129.1.45.5
        Addresses bound to peer LDP Ident:
          129.1.58.5      129.1.45.5      129.1.54.5      150.1.5.5

Rack1R4#show mpls ldp discovery
 Local LDP Identifier:
    150.1.4.4:0
    Discovery Sources:
    Interfaces:
        FastEthernet0/0 (ldp): xmit/recv
            LDP Id: 150.1.5.5:0
        FastEthernet0/1 (ldp): xmit/recv
            LDP Id: 150.1.6.6:0
        Serial0/0/0.54 (ldp): xmit/recv
            LDP Id: 150.1.5.5:0

Rack1R5#show mpls ldp neighbor
    Peer LDP Ident: 150.1.4.4:0; Local LDP Ident 150.1.5.5:0
        TCP connection: 150.1.4.4.646 - 150.1.5.5.25621
        State: Oper; Msgs sent/rcvd: 16/17; Downstream
        Up time: 00:02:11
        LDP discovery sources:
          FastEthernet0/1, Src IP addr: 129.1.45.4
          Serial0/0/0.54, Src IP addr: 129.1.54.4
        Addresses bound to peer LDP Ident:
          129.1.45.4      129.1.46.4      129.1.54.4      129.1.124.4
          150.1.4.4
```



```
Rack1R5#show mpls ldp discovery
 Local LDP Identifier:
    150.1.5.5:0
    Discovery Sources:
    Interfaces:
        FastEthernet0/1 (ldp): xmit/recv
            LDP Id: 150.1.4.4:0
        Serial0/0/0.54 (ldp): xmit/recv
            LDP Id: 150.1.4.4:0

Rack1R6#show mpls ldp discovery
 Local LDP Identifier:
    150.1.6.6:0
    Discovery Sources:
    Interfaces:
        FastEthernet0/0 (ldp): xmit/recv
            LDP Id: 150.1.4.4:0
Rack1R6#show mpls ldp neighbor
    Peer LDP Ident: 150.1.4.4:0; Local LDP Ident 150.1.6.6:0
        TCP connection: 150.1.4.4.646 - 150.1.6.6.24124
        State: Oper; Msgs sent/rcvd: 20/20; Downstream
        Up time: 00:04:22
        LDP discovery sources:
          FastEthernet0/0, Src IP addr: 129.1.46.4
        Addresses bound to peer LDP Ident:
          129.1.45.4      129.1.46.4      129.1.54.4      129.1.124.4
          150.1.4.4
```

For testing, you can also ping from R6 to R5, and verify that you see the counters increment in the output of show mpls forwarding.

```
Rack1R6#ping 150.1.5.5 repeat 10

Type escape sequence to abort.
Sending 10, 100-byte ICMP Echos to 150.1.5.5, timeout is 2 seconds:
!!!!!!!!!!
Success rate is 100 percent (10/10), round-trip min/avg/max = 1/1/4 ms

Rack1R4#show mpls forw 150.1.5.5
Local  Outgoing      Prefix          Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id    Switched      interface
18     Pop Label     150.1.5.5/32    1140          Fa0/0      129.1.45.5
```


Task 4.3 Solution

R5:

```
router bgp 100
  no bgp default ipv4-unicast
  neighbor 150.1.6.6 remote-as 100
  neighbor 150.1.6.6 update-source lo0
 address-family vpnv4 uni
  neighbor 150.1.6.6 activate
  address-family ipv4 vrf VPNA
  redistribute connected
```

R6:

```
router bgp 100
no bgp default ipv4-unicast
 neighbor 150.1.5.5 remote-as 100
 neighbor 150.1.5.5 upd lo0
address-family vpnv4 uni
 neighbor 150.1.5.5 activate
address-family ipv4 vrf VPNB
 redistribute connected

router ospf 12 vrf VPNB
 redist bgp 100 subnets
```

Task 4.3 Breakdown

Here, we have the neighbors added to BGP for the address families, in addition to redistribution for the VRFs. When redistributing into BGP for the VRFs on the endpoints, normally you would redistribute based on the VRF routing protocols. Since R6 only has the connected network in OSPF, redistribute connected is sufficient for the reachability for this section.

Task 4.3 Verification

Verify that R5 and R6 show the routes.

```
Rack1R5#show ip bgp vpnv4 all
BGP table version is 7, 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: 4:4 (default for vrf VPNA)
*>i10.0.0.0/24      150.1.6.6                0    100      0 ?
*> 50.0.0.1/32      0.0.0.0                  0         32768 ?
*> 51.0.0.1/32      0.0.0.0                  0         32768 ?
Route Distinguisher: 6:6
*>i10.0.0.0/24      150.1.6.6                0    100      0 ?
```



```
Rack1R6#show ip bgp vpnv4 all
BGP table version is 7, 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: 4:4
*>i50.0.0.1/32      150.1.5.5                0    100      0 ?
*>i51.0.0.1/32      150.1.5.5                0    100      0 ?
Route Distinguisher: 6:6 (default for vrf VPNB)
*> 10.0.0.0/24      0.0.0.0                  0         32768 ?
*>i50.0.0.1/32      150.1.5.5                0    100      0 ?
*>i51.0.0.1/32      150.1.5.5                0    100      0 ?
```

```
Rack1R5#show ip route vrf VPNA | beg Gate
Gateway of last resort is not set

     51.0.0.0/32 is subnetted, 1 subnets
C       51.0.0.1 is directly connected, Loopback51
     50.0.0.0/32 is subnetted, 1 subnets
C       50.0.0.1 is directly connected, Loopback50
     10.0.0.0/24 is subnetted, 1 subnets
B       10.0.0.0 [200/0] via 150.1.6.6, 00:06:50
```

Looking at the mpls forwarding table on R4, you can see entries for R5 and R6's loopbacks with a pop tag.

```
Rack1R4#show mpls forwarding 150.1.5.5
Local  Outgoing      Prefix          Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id    Switched      interface
18     Pop Label     150.1.5.5/32    4293          Fa0/0      129.1.45.5

Rack1R4#show mpls forw 150.1.6.6
Local  Outgoing      Prefix          Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id    Switched      interface
19     Pop Label     150.1.6.6/32    6227          Fa0/1      129.1.46.6
```

Next, take a look at the traffic flow. Starting on R5, look at the CEF entry for the destination network.


```
Rack1R5#show ip cef vrf VPNA 10.0.0.0/24
10.0.0.0/24
  nexthop 129.1.45.4 FastEthernet0/1 label 19 25
```

The CEF table gives us the label information, which can be traced through R5 to R6.

```
Rack1R5#show mpls forwarding label 19
Local  Outgoing      Prefix          Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id    Switched      interface
19     19            150.1.6.6/32    0             Fa0/1      129.1.45.4

Rack1R4#show mpls forwarding label 19
Local  Outgoing      Prefix          Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id    Switched      interface
19     Pop Label     150.1.6.6/32    6805          Fa0/1      129.1.46.6

Rack1R6#show mpls forwarding label 26
Local  Outgoing      Prefix          Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id    Switched      interface
26     No Label      10.0.0.0/24[V]  1140          aggregate/VPNB
```

Check that SW4 can ping the VPNA loopbacks on R5:

```
Rack1SW4#show ip route vrf TEST ospf
     51.0.0.0/32 is subnetted, 1 subnets
O E2    51.0.0.1 [110/1] via 10.0.0.6, 00:06:15, FastEthernet0/6
     50.0.0.0/32 is subnetted, 1 subnets
O E2    50.0.0.1 [110/1] via 10.0.0.6, 00:06:15, FastEthernet0/6

Rack1SW4#ping vrf TEST 51.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 51.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms

Rack1SW4#ping vrf TEST 50.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 50.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/8 ms
```


Task 5.1 Solution

R3:

```
interface Serial1/2
 ip multicast helper-map 225.25.25.25 129.1.23.255 111
!
interface Serial1/3
 ip directed-broadcast
!
access-list 111 permit udp any any eq 31337
!
ip forward-protocol udp 31337
```

R2:

```
interface Serial0/1
 ip multicast helper-map broadcast 225.25.25.25 111
!
access-list 111 permit udp any any eq 31337
!
ip forward-protocol udp 31337
```


Task 5.1 Verification

In order to test the above configuration, a router configured with the IP SLA monitor feature in VLAN 17 will be designated as the multicast server, while another router in VLAN 22 will be the multicast client:

```
SW1:
rtr 1
 type udpEcho  dest-ipaddr 225.25.25.25 dest-port 31337 source-ipaddr
129.1.17.7 source-port 31337 control disable
 timeout 1
 frequency 5
rtr schedule 1 start-time now
!
ip multicast-routing distributed
!
interface Vlan 17
 ip pim dense-mode
```

Make sure to remove the PIM mode when done testing:

```
R1:
!
Rack1R1(config)#interface fastethernet 0/0
Rack1R1(config-if)#no ip mroute-cache
                   ↑       ↑       ↑
            multicast fast switching disabled on
              the incoming interface so debug
                  output can be seen

Rack1R1#show ip mroute
<snip>

(*, 225.25.25.25), 00:08:28/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/1, Forward/Dense, 00:08:28/00:00:00

(129.1.17.7, 225.25.25.25), 00:08:28/00:02:50, flags: T
  Incoming interface: FastEthernet0/0, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/1, Forward/Dense, 00:08:28/00:00:00
                              ↑     ↑     ↑
                Indicates a multicast feed destined for 225.25.25.25
                  is being received from 129.1.17.7 in interface
              FastEthernet0/0, and is forwarded out interface Serial0/1

Rack1R1#debug ip mpacket
IP multicast packets debugging is on
Rack1R1#
IP(0): s=129.1.17.7 (FastEthernet0/0) d=225.25.25.25 (Serial0/1) id=0,
prot=17, len=44(44), mforward
Rack1R1#
```



```
IP(0): s=129.1.17.7 (FastEthernet0/0) d=225.25.25.25 (Serial0/1) id=0,
prot=17, len=44(44), mforward
Rack1R1#
IP(0): s=129.1.17.7 (FastEthernet0/0) d=225.25.25.25 (Serial0/1) id=0,
prot=17, len=44(44), mforward
↑       ↑       ↑
packets generated by SLA are received by R1
in the Ethernet interface connecting
to VLAN 17 and are forwarded out
interface Serial 0/1 to R3

Rack1R3#show ip mroute
<snip>

(*, 225.25.25.25), 00:18:53/stopped, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial1/2, Forward/Dense, 00:18:53/00:00:00

(129.1.17.7, 225.25.25.25), 00:12:52/00:02:57, flags: PLTX
  Incoming interface: Serial1/2, RPF nbr 129.1.13.1
  Outgoing interface list: Null
  ↑             ↑             ↑
  Feed is received in Serial1/2
  but it is not forwarded anywhere

Rack1R2#debug ip packet detail 111
IP packet debugging is on (detailed) for access list 111
↑               ↑                 ↑
Previously defined access-list 111
used to filter debug output
e
Rack1R2#
IP: s=129.1.17.7 (Serial0/1), d=255.255.255.255, len 44, rcvd 2
    UDP src=31337, dst=31337
Rack1R2#
IP: s=129.1.17.7 (Serial0/1), d=255.255.255.255, len 44, rcvd 2
    UDP src=31337, dst=31337
↑               ↑                 ↑
R2 received the feed as an IP broadcast

Rack1R2#show access-lists
Extended IP access list 111
    10 permit udp any any eq 31337 (319 matches)
      ↑               ↑                 ↑
      Broadcast feed hits the helper-map and
      is translated back into a multicast feed
```

Testing only, remove when done

SW4:

```
ip multicast-routing distributed
ip mroute 129.1.17.7 255.255.255.255 192.10.1.2
!
vlan 22
!
interface vlan 22
 ip address 192.10.1.10 255.255.255.0
 ip pim dense
 no ip mroute-cache
```

SW3:

```
vlan 22

IP(0): s=129.1.17.7 (FastEthernet0/0) d=225.25.25.25 id=0, prot=17,
len=60(44), mroute olist null
```

```
Rack1SW4#
IP(0): s=129.1.17.7 (FastEthernet0/0) d=225.25.25.25 id=0, prot=17,
len=60(44), mroute olist null
```

Client receives transmission as a multicast
Broadcast conversion is transparent to the client

```
Rack1SW4#show ip mroute | beg \(129
(129.1.17.7, 225.25.25.25), 00:01:43/00:02:56, flags: PT
  Incoming interface: Vlan22, RPF nbr 192.10.1.2, Mroute
  Outgoing interface list: Null

Rack1R2#show ip mroute | beg \(129
(129.1.17.7, 225.25.25.25), 00:08:34/00:02:57, flags: T
  Incoming interface: Serial0/1, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0, Forward/Dense, 00:01:26/00:00:00
```

A few notes on testing:

Without a PIM neighbor on R2's FastEthernet segment, you may see "Null" for the outgoing interface list in the output of show ip mroute.

In the testing / verification shown, SW1 and SW4 had PIM modes configured. Our section did explicitly state to not add PIM on additional interfaces for the traffic to pass, so make sure that you remove the PIM statements from the interfaces on SW1 and SW4.

Task 5.2 Solution

R4 and R5:

```
!
ip multicast-routing
!
interface Loopback1
 ip address 150.1.0.255 255.255.255.255
!
interface FastEthernet0/0
 ip pim sparse-mode
!
interface FastEthernet0/1
 ip pim sparse-mode
!
router ospf 1
 network 150.1.0.255 0.0.0.0 area 0
!
ip pim rp-address 150.1.0.255
```

R4:

```
!
ip msdp peer 150.1.5.5 connect-source Loopback0
ip msdp originator-id loopback 0
```

R5:

```
!
ip msdp peer 150.1.4.4 connect-source Loopback0
ip msdp originator-id loopback 0
```

R6:

```
!
ip multicast-routing
!
ip pim rp-address 150.1.0.255
!
interface FastEthernet0/0
 ip pim sparse-mode
```

SW2:

```
!
ip multicast-routing distributed
!
ip pim rp-address 150.1.0.255
!
interface Vlan58
 ip pim sparse-mode
```

📖 Further Reading

Anycast RP

Task 5.3 Verification

```
Rack1R6#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s): 224.0.0.0/4, Static
    RP: 150.1.0.255 (?)

Rack1R4#show ip msdp peer
MSDP Peer 150.1.5.5 (?), AS 100
  Connection status:
    State: Up, Resets: 0, Connection source: Loopback0 (150.1.4.4)
    Uptime(Downtime): 00:48:18, Messages sent/received: 48/53
    Output messages discarded: 0
    Connection and counters cleared 00:48:18 ago
  SA Filtering:
    Input (S,G) filter: none, route-map: none
    Input RP filter: none, route-map: none
    Output (S,G) filter: none, route-map: none
    Output RP filter: none, route-map: none
  SA-Requests:
    Input filter: none
  Peer ttl threshold: 0
  SAs learned from this peer: 0
    Input queue size: 0, Output queue size: 0
  MD5 signature protection on MSDP TCP connection: not enabled
  Message counters:
    RPF Failure count: 0
    SA Messages in/out: 13/0
    SA Requests in: 0
    SA Responses out: 0
    Data Packets in/out: 3/0

Rack1R5#show ip msdp peer
MSDP Peer 150.1.4.4 (?), AS 100
  Connection status:
    State: Up, Resets: 0, Connection source: Loopback0 (150.1.5.5)
    Uptime(Downtime): 00:49:28, Messages sent/received: 54/49
    Output messages discarded: 0
    Connection and counters cleared 00:49:35 ago
  SA Filtering:
    Input (S,G) filter: none, route-map: none
    Input RP filter: none, route-map: none
    Output (S,G) filter: none, route-map: none
    Output RP filter: none, route-map: none
  SA-Requests:
    Input filter: none
  Peer ttl threshold: 0
  SAs learned from this peer: 0
    Input queue size: 0, Output queue size: 0
  MD5 signature protection on MSDP TCP connection: not enabled
  Message counters:
    RPF Failure count: 0
    SA Messages in/out: 0/10
    SA Requests in: 0
    SA Responses out: 0
    Data Packets in/out: 0/3
```


For testing purposes, we will have R6's Loopback0 join multicast group 226.26.26.26

```
R6:
!
interface Loopback0
  ip igmp join-group 226.26.26.26
 ip pim sparse-mode

Rack1SW2#ping 226.26.26.26 repeat 5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 226.26.26.26, timeout is 2 seconds:

Reply to request 0 from 129.1.46.6, 9 ms
Reply to request 1 from 129.1.46.6, 1 ms
Reply to request 2 from 129.1.46.6, 1 ms
Reply to request 3 from 129.1.46.6, 1 ms
Reply to request 4 from 129.1.46.6, 1 ms

Rack1R4#show ip msdp sa-cache
MSDP Source-Active Cache - 1 entries
(129.1.58.8, 226.26.26.26), RP 150.1.5.5, BGP/AS 0, 00:01:10/00:05:04,
Peer 150.1.5.5

R6:
!
interface Loopback0
 no ip igmp join-group 226.26.26.26
 no ip pim sparse-mode
```

Task 6.1 Solution

```
R6:
access-list 100 permit tcp host 129.1.46.100 any eq telnet
access-list 100 deny   tcp any any eq telnet log
!
line vty 0 988
 access-class 100 in
```

Task 6.1 Verification

```
Rack1R6#telnet 150.1.6.6
Trying 150.1.6.6 ...
% Connection refused by remote host

Rack1R6#
%SEC-6-IPACCESSLOGP: list 100 denied tcp 150.1.6.6(44049) ->
0.0.0.0(23), 1 packet
```


Task 6.2 Solution

R2:

```
!
access-list 22 permit 129.1.0.0 0.0.255.255
!
login block-for 300 attempts 10 within 60
login quiet-mode access-class 22
!
username cisco password cisco
login on-failure
line vty 0 181
 login local
```

Task 6.2 Verification

```
Rack1R2#show login
     A default login delay of 1 seconds is applied.
     Quiet-Mode access list 22 is applied.
     All failed login is logged and generate SNMP traps.

     Router enabled to watch for login Attacks.
     If more than 10 login failures occur in 60 seconds or less,
     logins will be disabled for 300 seconds.

     Router presently in Normal-Mode.
     Current Watch Window
         Time remaining: 42 seconds.
         Login failures for current window: 0.
     Total login failures: 0.
```

For testing, start with lower values for attempts:

```
Rack1R2(config)#login block 300 attempts 2 within 600
Rack1R2(config)#end
Rack1R2#telnet 150.1.2.2 /source lo0
Trying 150.1.2.2 ... Open


User Access Verification

Username: c
Password:
% Login invalid

%SEC_LOGIN-4-LOGIN_FAILED: Login failed [user: c] [Source: 150.1.2.2]
[localport: 23] [Reason: Login Authentication Failed - BadUser] at
23:51:09 PST Sat Mar 2 2002

Username: c
Password:
% Login invalid
```



```
%SEC_LOGIN-4-LOGIN_FAILED: Login failed [user: c] [Source: 150.1.2.2]
[localport: 23] [Reason: Login Authentication Failed - BadUser] at
23:51:19 PST Sat Mar 2 2002

%SEC_LOGIN-1-QUIET_MODE_ON: Still timeleft for watching failures is 574
secs, [user: c] [Source: 150.1.2.2] [localport: 23] [Reason: Login
Authentication Failed - BadUser] [ACL: 22] at 23:51:19 PST Sat Mar 2
2002
[Connection to 150.1.2.2 closed by foreign host]
```

Adjust to the values specified in the section:

R2:

```
!
login block-for 300 attempts 10 within 60
```

Task 6.2 Breakdown

Security enhancements allow conditional blocking to prevent the router from being impacted by a denial of service or brute force attack. The login block-for command allows you to set a threshold time period, such that if a certain number of failed attempts are received, access will be blocked. The quiet-mode ACL allows you to specify which hosts are allowed to access the device, even if the block threshold is exceeded. The login on-failure command, although not mandated by the section, will allow you to see the failed attempts logged locally. If you are just using a password on the line, it will not trigger the feature, so username and password are configured, along with login local on the VTY lines.

Further Reading

Cisco IOS Login Enhancements

Task 7.1 Solution

R6:

```
logging host ipv6 2001:CC1E:1:1::100
!
ip access-list log-update threshold 10
```


Task 7.1 Breakdown

This task is very straightforward. Configure the logging destination and adjust the threshold. Make sure that your logging level is informational or debugging in order to get hits for ACL entries. By default, the logging severity level is high enough, but it is possible that a lower level could have been set in the initial configuration.

```
Rack1R6#show logging | beg Trap
    Trap logging: level informational, 95 message lines logged
        Logging to 2001:CC1E:1:1::100  (udp port 514,  audit disabled,
              authentication disabled, encryption disabled, link up),
              4 message lines logged,
```

Task 7.2 Solution

R1, R2, SW1:

```
ntp server 150.1.3.3
```

R3, R6:

```
ntp master
```

R4, R5, SW2:

```
ntp server 150.1.6.6
```

R1, R2, R3, SW1:

```
!
clock timezone PST -8
clock summer-time PDT recurring
```

R4, R5, R6, SW2:

```
clock timezone CST -6
clock summer-time CDT recurring
```

SW3 and SW4:

```
ntp server 150.1.6.6
```


Task 7.2 Verification

Verify that the clocks are synchronized. For instance on R1:

```
Rack1R1#show ntp status
Clock is synchronized, stratum 9, reference is 150.1.3.3
nominal freq is 249.5901 Hz, actual freq is 249.5902 Hz, precision is
2**18
reference time is C043C87E.A5BC8624 (18:48:30.647 PST Wed Mar 20 2002)
clock offset is -7.5216 msec, root delay is 25.09 msec
root dispersion is 10.45 msec, peer dispersion is 2.90 msec
```

R6 is in Chicago (UTC -6), while R2 is in Reno (UTC -8):

```
Rack1R6#show clock
12:27:53.275 CDT Mon May 17 2010

Rack1R6#show ntp status
Clock is synchronized, stratum 8, reference is 127.127.1.1
nominal freq is 250.0000 Hz, actual freq is 249.9958 Hz, precision is
2**24
reference time is CF9BFB42.8C493FC4 (12:28:34.547 CDT Mon May 17 2010)
clock offset is 0.0000 msec, root delay is 0.00 msec
root dispersion is 0.00 msec, peer dispersion is 0.00 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is
0.000016483 s/s
system poll interval is 16, last update was 2 sec ago.

Rack1R2#show clock
18:50:34.471 PST Wed Mar 20 2002


Rack1R2#show ntp status
Clock is synchronized, stratum 9, reference is 150.1.3.3
nominal freq is 249.5901 Hz, actual freq is 249.5902 Hz, precision is
2**18
reference time is C043C8FF.9B019B05 (18:50:39.605 PST Wed Mar 20 2002)
clock offset is -4.5501 msec, root delay is 25.12 msec
root dispersion is 5.63 msec, peer dispersion is 1.04 msec

Rack1SW3#show version | include started
System restarted at 06:33:06 UTC Mon May 17 2010
```


Note

When NTP is configured, the device will also timestamp the last configuration change and the last time the configuration was saved to NVRAM in the configuration itself.

```
Rack1SW3#show running-config | include Last|NVRAM
! Last configuration change at 08:00:33 UTC Sun Jan 15 2010
! NVRAM config last updated at 08:06:55 UTC Sun Jan 15 2010
```

Task 7.2 Breakdown

NTP advertisements are always sent in Coordinated Universal Time (UTC), also commonly known as Greenwich Mean Time (GMT). In order to avoid log inconsistencies due to devices being located in different time zones, it is common practice to leave the local time in UTC. However, the time zone of the router's local clock can be adjusted by issuing the **clock timezone [*timezone*] [*offset*]** global configuration command. Additionally, daylight savings time can be configured with the **clock summer-time [*daylight timezone*] recurring** command. Time zone configuration is always locally significant, and is never propagated via NTP.

Task 7.3 Solution

R1, R2, SW1:

```
ip domain-lookup
ip name-server 150.1.3.3
```

R3:

```
ip dns server
ip domain-lookup
!
ip host Rack1R1 150.1.1.1
ip host Rack1R2 150.1.2.2
ip host Rack1SW1 150.1.7.7
```

Task 7.3 Verification

Verify the new domain server:

```
Rack1R1#ping Rack1R2
Translating "Rack1R2"...domain server (150.1.3.3)

Translating "Rack1R2"...domain server (150.1.3.3) [OK]

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 = 28/29/32 ms

Rack1R1#ping Rack1SW1
Translating "Rack1SW1"...domain server (150.1.3.3)

Translating "Rack1SW1"...domain server (150.1.3.3) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.7.7, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```


Task 7.4 Solution

R4:

```
interface FastEthernet0/0
 glbp 1 ip 129.1.45.6
 glbp 1 preempt
 glbp 1 weighting 30
 glbp 1 load-balancing weighted
```

R5:

```
interface FastEthernet0/1
 glbp 1 ip 129.1.45.6
 glbp 1 priority 50
 glbp 1 preempt
 glbp 1 weighting 70
 glbp 1 load-balancing weighted
```

Task 7.4 Verification

```
Rack1R4#show glbp brief
Interface   Grp  Fwd Pri State    Address  Active router Standby router
Fa0/0       1    -   100 Active   129.1.45.6    local         129.1.45.5
Fa0/0       1    1   -   Active   0007.b400.0101  local           -
Fa0/0       1    2   -   Listen   0007.b400.0102  129.1.45.5      -
```



```
Rack1R4#show glbp
FastEthernet0/0 - Group 1
  State is Active
    2 state changes, last state change 00:00:57
  Virtual IP address is 129.1.45.6
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 2.367 secs
  Redirect time 600 sec, forwarder time-out 14400 sec
  Preemption enabled, min delay 0 sec
  Active is local
  Standby is 129.1.45.5, priority 50 (expires in 8.361 sec)
  Priority 100 (default)
  Weighting 30 (configured 30), thresholds: lower 1, upper 30
  Load balancing: weighted
  Group members:
    000f.90fa.ed60 (129.1.45.4) local
    000f.90fb.0a21 (129.1.45.5)
  There are 2 forwarders (1 active)
  Forwarder 1
    State is Active
      1 state change, last state change 00:00:47
    MAC address is 0007.b400.0101 (default)
    Owner ID is 000f.90fa.ed60
    Redirection enabled
    Preemption enabled, min delay 30 sec
    Active is local, weighting 30
  Forwarder 2
    State is Listen
    MAC address is 0007.b400.0102 (learnt)
    Owner ID is 000f.90fb.0a21
    Redirection enabled, 597.572 sec remaining (maximum 600 sec)
    Time to live: 14397.572 sec (maximum 14400 sec)
    Preemption enabled, min delay 30 sec
  Active is 129.1.45.5 (primary), weighting 70 (expires in 7.568 sec)
```


Task 8.1 Solution

R2:

```
interface Serial0/0
 frame-relay traffic-shaping
 frame-relay class DLCI_204
!
map-class frame-relay DLCI_204
 frame-relay cir 512000
 frame-relay bc 5120
 frame-relay be 0
 frame-relay fragment 640
```

R4:

```
interface Serial0/0/0
 frame-relay traffic-shaping
!
interface Serial0/0/0.124 multipoint
 frame-relay interface-dlci 401
  class DLCI_401
 frame-relay interface-dlci 402
  class DLCI_402
!
interface Serial0/0/0.54 point-to-point
 frame-relay interface-dlci 405
  class EEK
!
map-class frame-relay EEK
 frame-relay cir 512000
 frame-relay bc 5120
 frame-relay be 0
 frame-relay fragment 640
!
map-class frame-relay DLCI_401
 frame-relay cir 512000
 frame-relay bc 5120
 frame-relay be 0
 frame-relay fragment 640
!
map-class frame-relay DLCI_402
 frame-relay cir 512000
 frame-relay bc 5120
 frame-relay be 0
 frame-relay fragment 640
```

Quick Note
Previously applied.

Quick Note
Previously applied.

Task 8.1 Breakdown

The smaller the Frame Relay Traffic Shaping interval (Tc), the less time traffic is delayed in the output queue as it is waiting to exit to the transmit ring. This in turn equates to less delay, and better performance, for low bandwidth delay sensitive traffic such as VoIP. However, lowering the shaping interval does not accomplish anything when the MTU of a packet exceeds the Bc value.

Suppose that the MTU of the interface is 1500 bytes, and that in each Tc the FRTS algorithm has allotted 5120 bits of committed burst. This means that it will take a minimum of three intervals (30ms in this case) in order to clock this packet onto the interface. Depending on the serialization delay of the interface (dependent on the hardware clocking speed), this delay in sending the packet can result in unacceptable delay for real time traffic, even if it is prioritized. This is due to the fact that even if a packet is in the low latency queue, it must wait for whatever packet is on the transmit ring to exit the interface.

In order to further reduce the delay of real time traffic as it exits the output queue, Frame Relay fragmentation can be used to reduce the MTU of packets transmitted out the interface. By reducing the maximum fragment size to Bc (in bytes), a real time packet such as VoIP is guaranteed that the worst case scenario delay that will be incurred in the output queue is one single Tc (10ms in this case).

Task 8.1 Verification

```
Rack1R4#show frame-relay pvc 402

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

DLCI = 402, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0/0.124

  input pkts 2623          output pkts 2717         in bytes 364700
  out bytes 380229         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 1208      out bcast bytes 289541
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 11:39:51, last time pvc status changed 11:34:56
  fragment type end-to-end fragment size 640
  cir 512000    bc 5120      be 0         byte limit 640    interval 10
  mincir 256000    byte increment 640   BECN response no  IF_CONG no
  frags 2         bytes 151       frags delayed 0         bytes delayed
0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: weighted fair
  Current fair queue configuration:
   Discard     Dynamic      Reserved
   threshold   queue count  queue count
    64          16           0
  Output queue size 0/max total 600/drops 0
```



```
Rack1R2#show frame-relay pvc 204

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

DLCI = 204, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0

  input pkts 2714          output pkts 2624         in bytes 378812
  out bytes 364279         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 1153      out bcast bytes 279976
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 11:36:11, last time pvc status changed 11:35:32
  fragment type end-to-end fragment size 640
  cir 512000    bc   5120      be 0         limit 640    interval 10
  mincir 256000    byte increment 640   BECN response no  IF_CONG no
  frags 12        bytes 1746      frags delayed 0         bytes delayed
0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: weighted fair
  Current fair queue configuration:
   Discard     Dynamic      Reserved
   threshold   queue count  queue count
    64          16           0
  Output queue size 0/max total 600/drops 0
```


Task 8.2 Solution

R2:

```
class-map match-all VoIP
  match access-group name VoIP
!
 policy-map LLQ
  class VoIP
   priority 192
!
map-class frame-relay DLCI_204
 service-policy output LLQ
!
ip access-list extended VoIP
 permit udp any 129.1.46.0 0.0.0.255 range 16384 32767
```

R4:

```
class-map match-all VoIP
  match access-group name VoIP
!
 policy-map LLQ
  class VoIP
   priority 192
!
map-class frame-relay DLCI_402
 service-policy output LLQ
!
ip access-list extended VOIP
 permit udp 129.1.46.0 0.0.0.255 any range 16384 32767
```

Task 8.2 Breakdown

By putting VoIP traffic in the low latency queue by using the **priority** keyword under the MQC policy-map, VoIP traffic is always guaranteed to be dequeued first on the Frame Relay circuit between R2 and R4 up to 192Kbps. When VoIP traffic exceeds 192Kbps of the output queue, it is not guaranteed low latency, but may be transmitted. When VoIP traffic exceeds 192Kbps of the output queue, and there is congestion in the queue, VoIP in excess of 192Kbps will be dropped.

Task 8.2 Verification

```
Rack1R4#show frame-relay pvc 402

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

DLCI = 402, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0/0.124

  input pkts 2648          output pkts 2742         in bytes 367995
  out bytes 383584         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 1223      out bcast bytes 292361
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 11:44:45, last time pvc status changed 11:39:50
  fragment type end-to-end fragment size 640
  cir 512000    bc 5120      be 0         byte limit 640    interval 10
  mincir 256000    byte increment 640   BECN response no  IF_CONG no
  frags 27        bytes 3506      frags delayed 0       bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  service policy LLQ
 Serial0/0/0.124: DLCI 402 -

  Service-policy output: LLQ

    queue stats for all priority classes:

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

    Class-map: VoIP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name VoIP
      Priority: 192 kbps, burst bytes 4800, b/w exceed drops: 0


    Class-map: class-default (match-any)
      2 packets, 151 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 0/0
  Output queue 0/192, 0 drop, 0 dequeued
```



```
Rack1R2#show frame-relay pvc 204

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

DLCI = 204, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0

  input pkts 2741          output pkts 2650         in bytes 382643
  out bytes 367658         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 1169      out bcast bytes 282820
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 11:41:37, last time pvc status changed 11:40:57
  fragment type end-to-end fragment size 640
  cir 512000    bc   5120      be 0         limit 640    interval 10
  mincir 256000    byte increment 640   BECN response no  IF_CONG no
  frags 38        bytes 5125      frags delayed 0       bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  service policy LLQ
 Serial0/0: DLCI 204 -

  Service-policy output: LLQ

    Class-map: VoIP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name VoIP
      Queueing
        Strict Priority
        Output Queue: Conversation 40
        Bandwidth 192 (kbps) Burst 4800 (Bytes)
        (pkts matched/bytes matched) 0/0
        (total drops/bytes drops) 0/0

    Class-map: class-default (match-any)
      10 packets, 1318 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
  Output queue size 0/max total 600/drops 0
```


Task 8.3 Solution

SW3:

```
mls qos
!
ip access-list extended HTTP_REPLIES
 permit tcp any eq 80 any
!
ip access-list extended SMTP_REPLIES
 permit tcp any eq 25 any
!
class-map HTTP_REPLIES
 match access-group name HTTP_REPLIES
!
class-map SMTP_REPLIES
 match access-group name SMTP_REPLIES
!
mls qos aggregate-policer POLICE_2M 2000000 128000 exceed-action drop
!
policy-map MARK_AND_POLICE
 class HTTP_REPLIES
  set dscp af21
  police aggregate POLICE_2M
 class SMTP_REPLIES
  set dscp af23
  police aggregate POLICE_2M
!
interface FastEthernet 0/5
 service-policy input MARK_AND_POLICE
```


Task 8.3 Verification

```
Rack1SW3#show policy-map interface fastEthernet 0/5
 FastEthernet0/5

  Service-policy input: MARK_AND_POLICE

    Class-map: HTTP_REPLIES (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name HTTP_REPLIES

    Class-map: SMTP_REPLIES (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name SMTP_REPLIES

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

Rack1SW3#show mls qos aggregate-policer
aggregate-policer POLICE_2M 2000000 128000 exceed-action drop

Rack1SW3#show mls qos interface FastEthernet 0/5 ?
  buffers     Show buffer information
  policers    Show policers information
  queueing    Show queueing information
  statistics  Show statistics
  |           Output modifiers
  <cr>

Rack1SW3#show mls qos interface FastEthernet 0/5 policers
FastEthernet0/5
policymap=MARK_AND_POLICE

type=Shared, id=1 name=POLICE_2M
```


Lab 13 Solutions

Task 1.1 Solution

SW1:

```
interface FastEthernet0/22
 switchport voice vlan dot1p
```

Task 1.1 Verification

```
Rack1SW1#show interfaces fa0/22 switchport | include Voice
Voice VLAN: dot1p
```

Task 1.2 Solution

R4:

```
interface Serial0/1/0
 ip address negotiated
 encapsulation ppp
 clockrate 64000
 no shutdown
```

R5:

```
interface Serial0/1/0
 encapsulation ppp
 peer default ip address dhcp
 clockrate 64000
 no shutdown
!
ip address-pool dhcp-proxy-client
ip dhcp-server 139.1.11.100
```


Tasks 1.2 & 7.3 Verification

This task should be verified in conjunction with Task 7.3. Apply Task 7.3 solution in order to perform complete verification. The preferred option at this point of the lab would be to temporarily hardcode R4's IP address. Then, after full IP reachability has been obtained, R4's IP address can be learned dynamically. If you use this option, be sure to write down what workaround you have put in place so that later in the lab you will be sure to come back to solve the task correctly.

Enable debugging:

```
Rack1R4#debug ppp negotiation
PPP protocol negotiation debugging is on

Rack1R5#debug dhcp
DHCP client activity debugging is on

Rack1R1#debug ip dhcp server events
Rack1R4(config)#interface s0/1/0
Rack1R4(config-if)#shutdown
Rack1R4(config-if)#no shutdown

Se0/1/0 PPP: Phase is ESTABLISHING, Active Open
Se0/1/0 LCP: O CONFREQ [Closed] id 6 len 10
Se0/1/0 LCP:    MagicNumber 0x30A1E593 (0x050630A1E593)
Se0/1/0 LCP: I CONFREQ [REQsent] id 6 len 10
Se0/1/0 LCP:    MagicNumber 0x07F9584E (0x050607F9584E)
Se0/1/0 LCP: O CONFACK [REQsent] id 6 len 10
Se0/1/0 LCP:    MagicNumber 0x07F9584E (0x050607F9584E)
Se0/1/0 LCP: I CONFACK [ACKsent] id 6 len 10
Se0/1/0 LCP:    MagicNumber 0x30A1E593 (0x050630A1E593)
Se0/1/0 LCP: State is Open
Se0/1/0 PPP: Phase is FORWARDING, Attempting Forward
Se0/1/0 PPP: Phase is ESTABLISHING, Finish LCP
Se0/1/0 PPP: Phase is UP
Se0/1/0 IPCP: O CONFREQ [Closed] id 1 len 10
Se0/1/0 IPCP:    Address 0.0.0.0 (0x030600000000)
Se0/1/0 CDPCP: O CONFREQ [Closed] id 1 len 4
Se0/1/0 PPP: Process pending ncp packets
Se0/1/0 IPCP: I CONFREQ [REQsent] id 1 len 10
Se0/1/0 IPCP:    Address 139.1.45.5 (0x03068B012D05)
Se0/1/0 IPCP: O CONFACK [REQsent] id 1 len 10
Se0/1/0 IPCP:    Address 139.1.45.5 (0x03068B012D05)
Se0/1/0 CDPCP: I CONFREQ [REQsent] id 1 len 4
Se0/1/0 CDPCP: O CONFACK [REQsent] id 1 len 4
Se0/1/0 CDPCP: I CONFACK [ACKsent] id 1 len 4
Se0/1/0 CDPCP: State is Open
Se0/1/0 IPCP: I CONFREQ [ACKsent] id 2 len 10
Se0/1/0 IPCP:    Address 139.1.45.5 (0x03068B012D05)
Se0/1/0 IPCP: O CONFACK [ACKsent] id 2 len 10
Se0/1/0 IPCP:    Address 139.1.45.5 (0x03068B012D05)
Se0/1/0 IPCP: TIMEout: State ACKsent
```



```
Se0/1/0 IPCP: O CONFREQ [ACKsent] id 2 len 10
Se0/1/0 IPCP:    Address 0.0.0.0 (0x030600000000)
Se0/1/0 IPCP: I CONFNAK [ACKsent] id 1 len 10
Se0/1/0 IPCP:    Address 139.1.45.4 (0x03068B012D04)
Se0/1/0 IPCP: ID 1 didn't match 2, discarding packet
Se0/1/0 IPCP: I CONFNAK [ACKsent] id 2 len 10
Se0/1/0 IPCP:    Address 139.1.45.4 (0x03068B012D04)
Se0/1/0 IPCP: O CONFREQ [ACKsent] id 3 len 10
Se0/1/0 IPCP:    Address 139.1.45.4 (0x03068B012D04)
Se0/1/0 IPCP: I CONFACK [ACKsent] id 3 len 10
Se0/1/0 IPCP:    Address 139.1.45.4 (0x03068B012D04)
Se0/1/0 IPCP: State is Open
Se0/1/0 IPCP: Install negotiated IP interface address 139.1.45.4
Se0/1/0 IPCP: Install route to 139.1.45.5
Se0/1/0 IPCP: Add link info for cef entry 139.1.45.5


Rack1R4#show ip interface s0/1/0
Serial0/1/0 is up, line protocol is up
  Internet address is 139.1.45.4/32
  Broadcast address is 255.255.255.255
  Address determined by IPCP
  Peer address is 139.1.45.5
<output omitted>

Rack1R5#
DHCP: proxy allocate request
DHCP: new entry. add to queue, interface
DHCP: SDiscover attempt # 1 for entry:
DHCP: SDiscover: sending 292 byte length DHCP packet
DHCP: SDiscover 292 bytes
DHCP: XID MATCH in dhcpc_for_us()
DHCP: Received a BOOTREP pkt
DHCP: offer received from 139.1.15.1
DHCP: SRequest attempt # 1 for entry:
DHCP: SRequest- Server ID option: 139.1.15.1
DHCP: SRequest- Requested IP addr option: 139.1.45.4
DHCP: SRequest placed lease len option: 86400
DHCP: SRequest: 310 bytes
DHCP: SRequest: 310 bytes
DHCP: SRequest attempt # 2 for entry:
DHCP: SRequest- Server ID option: 139.1.15.1
DHCP: SRequest- Requested IP addr option: 139.1.45.4
DHCP: SRequest placed lease len option: 86400
DHCP: SRequest: 310 bytes
DHCP: SRequest: 310 bytes
DHCP: XID MATCH in dhcpc_for_us()
DHCP: Received a BOOTREP pkt
DHCP Proxy Client Pooling: ***Allocated IP address: 139.1.45.4
```



```
Rack1R1#
DHCPD: assigned IP address 139.1.45.4 to client
0063.6973.636f.2d31.3339.2e31.2e34.352e.352d.5365.7269.616c.302f.31.

Rack1R1#show ip dhcp binding
Bindings from all pools not associated with VRF:
IP address          Client-ID/              Lease expiration
Type
                    Hardware address/
                    User name
139.1.45.4          0063.6973.636f.2d31.    Mar 02 1993 01:24 AM
Automatic
                    3339.2e31.2e34.352e.
                    352d.5365.7269.616c.
                    302f.31
```

Task 2.1 Solution

R3:
```
key chain RIP
 key 1
  key-string CISCO
!
interface FastEthernet0/1
 ip rip authentication mode md5
 ip rip authentication key-chain RIP
!
router rip
 version 2
 network 192.10.1.0
 no auto-summary
```

R4:
```
router rip
 version 2
 no validate-update-source
 redistribute connected metric 1 route-map CONNECTED_TO_RIP
 network 139.1.0.0
 network 150.1.0.0
 no auto-summary
!
route-map CONNECTED_TO_RIP permit 10
 match interface FastEthernet0/0
```

R5:
```
router rip
 version 2
 network 139.1.0.0
 network 150.1.0.0
 no auto-summary
```

SW2:
```
ip routing
!
router rip
 version 2
 network 139.1.0.0
 network 150.1.0.0
 no auto-summary
```

R4:
```
router rip
 offset-list 0 in 1 Serial0/1/0
```

R5:
```
router rip
 default-information originate
!
ip route 0.0.0.0 0.0.0.0 null0
```

R4, R5, and SW2:
```
router rip
 timers basic 3 18 18 24
```


Task 2.1 Breakdown

On R4, the redistribution will allow the Fa0/0 network to be advertised into RIP. Using a network statement with the passive interface command would still accept updates on that interface, which would break the section requirements. Due to the negotiated PPP connection being seen as a /32 locally, the addition of the "no validate-update-source" will prevent the error shown below:

```
RIP: ignored v2 update from bad source 139.1.45.5 on Serial0/1/0
```

RIP goes by hop count for path selection. The routes learned via SW2 will have a hop count that is one higher. By incrementing the routes learned via the serial link, both paths will have the same metric. With RIP, offset list 0 will match all routes without creating an access list.

RIP convergence time is dependent on the update and flush timers. The lower the flush timer is, the sooner the route will be removed out of the table if an update has not been received about it. Under normal circumstances, the age of a prefix will be reset every update timer. In this case, the flush time for the prefix should never be reached. When an update is not received, it is typically due to a lost routing path. In this case, the route is cleared out of the table when the age reaches the flush.

To change these timers, issue the **`timers basic`** RIP process subcommand. The default RIP timers are hello 30, invalid 180, hold down 180, and flush 240. To view these timer values, issue the **`show ip protocols`** command.

Note: Newer IOS versions also have a configuration option for a sleep timer, but there is not a fixed default value configured.

Task 2.1 Verification

Verify RIP configuration:

```
Rack1R3#show ip protocols
Routing Protocol is "rip"
  Sending updates every 30 seconds
  Invalid after 180 seconds, hold down 180, flushed after 240
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Redistributing: rip
  Default version control: send version 2, receive version 2
    Interface             Send  Recv  Triggered RIP  Key-chain
    FastEthernet0/1           2     2                    RIP
  Automatic network summarization is in effect
  Maximum path: 4
  Routing for Networks:
    192.10.1.0
  Routing Information Sources:
    Gateway         Distance      Last Update
    192.10.1.254         120      00:00:09
  Distance: (default is 120)
```

Verify RIP routes:

```
Rack1R3#show ip route rip
R    222.22.2.0/24 [120/7] via 192.10.1.254, 00:00:06, FastEthernet0/1
R    220.20.3.0/24 [120/7] via 192.10.1.254, 00:00:06, FastEthernet0/1
R    205.90.31.0/24 [120/7] via 192.10.1.254, 00:00:06, FastEthernet0/1

Rack1R5#show ip route rip
R    204.12.1.0/24 [120/1] via 139.1.45.4, 00:00:28, Serial0/1/0
     139.1.0.0/16 is variably subnetted, 7 subnets, 2 masks
R       139.1.48.0/24 [120/1] via 139.1.58.8, 00:00:20, FastEthernet0/1
                      [120/1] via 139.1.45.4, 00:00:28, Serial0/1/0
     150.1.0.0/24 is subnetted, 3 subnets
R       150.1.4.0 [120/1] via 139.1.45.4, 00:00:28, Serial0/1/0
R       150.1.8.0 [120/1] via 139.1.58.8, 00:00:20, FastEthernet0/1
```


Verify the RIP routes on R4 before the offset-list has been applied:

```
Rack1R4#show ip route rip
     139.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
R       139.1.15.0/24 [120/1] via 139.1.45.5, 00:00:26
R       139.1.5.0/24 [120/1] via 139.1.45.5, 00:00:26
R       139.1.25.0/24 [120/1] via 139.1.45.5, 00:00:26
R       139.1.45.0/24 [120/2] via 139.1.48.8, 00:00:19, FastEthernet0/1
R       139.1.58.0/24 [120/1] via 139.1.48.8, 00:00:19, FastEthernet0/1
                      [120/1] via 139.1.45.5, 00:00:26
     150.1.0.0/24 is subnetted, 3 subnets
R       150.1.5.0 [120/1] via 139.1.45.5, 00:00:26
R       150.1.8.0 [120/1] via 139.1.48.8, 00:00:19, FastEthernet0/1
R*   0.0.0.0/0 [120/1] via 139.1.45.5, 00:00:26
```

Apply offset list and verify the routes again:

```
Rack1R4#show ip route rip
     139.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
R       139.1.15.0/24 [120/2] via 139.1.48.8, 00:00:15, FastEthernet0/1
                      [120/2] via 139.1.45.5, 00:00:26
R       139.1.5.0/24 [120/2] via 139.1.48.8, 00:00:15, FastEthernet0/1
                     [120/2] via 139.1.45.5, 00:00:26
R       139.1.25.0/24 [120/2] via 139.1.48.8, 00:00:15, FastEthernet0/1
                      [120/2] via 139.1.45.5, 00:00:26
R       139.1.45.0/24 [120/2] via 139.1.48.8, 00:00:15, FastEthernet0/1
R       139.1.58.0/24 [120/1] via 139.1.48.8, 00:00:15, FastEthernet0/1
     150.1.0.0/24 is subnetted, 3 subnets
R       150.1.5.0 [120/2] via 139.1.48.8, 00:00:15, FastEthernet0/1
                  [120/2] via 139.1.45.5, 00:00:26
R       150.1.8.0 [120/1] via 139.1.48.8, 00:00:15, FastEthernet0/1
R*   0.0.0.0/0 [120/2] via 139.1.48.8, 00:00:15, FastEthernet0/1
               [120/2] via 139.1.45.5, 00:00:26
```

Before and after configuration, check timers with show ip protocols.

```
Rack1SW2# show ip protocols | include Sending|Invalid
Routing Protocol is "rip"
  Sending updates every 30 seconds, next due in 27 seconds
  Invalid after 180 seconds, hold down 180, flushed after 240

Rack1SW2#show ip protocols | include Sending|Invalid
  Sending updates every 3 seconds, next due in 1 seconds
  Invalid after 18 seconds, hold down 18, flushed after 24
```


Task 2.2 Solution

R2:

```
router ospf 1
 area 0 range 139.1.0.0 255.255.240.0
```

Task 2.2 Breakdown

By advertising a summary, R2 will be the less preferred path, since R5 will have a more specific route via R1. If the connection to R1 fails, the summary will be the route used, since R5 will no longer have a more specific route.

Task 2.2 Verification

Check that R5 prefers R1 to reach VLAN 2, 6, 7, 11, 367.

```
Rack1R5#show ip route ospf
     139.1.0.0/16 is variably subnetted, 15 subnets, 3 masks
O IA    139.1.11.0/24 [110/65] via 139.1.15.1, 00:02:49, Serial0/0.501
O IA    139.1.13.0/24 [110/128] via 139.1.15.1, 00:02:49, Serial0/0.501
O IA    139.1.2.0/24 [110/910] via 139.1.15.1, 00:02:49, Serial0/0.501
O IA    139.1.0.0/24 [110/129] via 139.1.15.1, 00:02:49, Serial0/0.501
IA    139.1.0.0/20 [110/65] via 139.1.25.2, 00:00:11, Serial0/0/0.502
O IA    139.1.6.0/24 [110/130] via 139.1.15.1, 00:02:49, Serial0/0.501
O IA    139.1.7.0/24 [110/130] via 139.1.15.1, 00:02:49, Serial0/0.501
O IA    139.1.23.0/24 [110/128] via 139.1.25.2, 00:02:49, Serial0/0.502
     150.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
O IA    150.1.7.7/32 [110/130] via 139.1.25.2, 00:02:49, Serial0/0.502
                     [110/130] via 139.1.15.1, 00:02:49, Serial0/0.501
O IA    150.1.6.6/32 [110/130] via 139.1.25.2, 00:02:49, Serial0/0.502
                     [110/130] via 139.1.15.1, 00:02:49, Serial0/0.501
O IA    150.1.3.3/32 [110/129] via 139.1.25.2, 00:02:50, Serial0/0.502
                     [110/129] via 139.1.15.1, 00:02:50, Serial0/0.501
O       150.1.2.2/32 [110/65] via 139.1.25.2, 00:02:50, Serial0/0.502
O       150.1.1.1/32 [110/65] via 139.1.15.1, 00:02:50, Serial0/0.501
```


Verify that R5 installs the summary in its RT:

```
Rack1R5#show ip route ospf | section 139.1.0.0/20
O IA    139.1.0.0/20 [110/65] via 139.1.25.2, 00:07:12, Serial0/0/0.502
```

Check the backup path:

```
Rack1R5(config)#interface s0/0.501
Rack1R5(config-subif)#shutdown
%OSPF-5-ADJCHG: Process 1, Nbr 150.1.1.1 on Serial0/0.501 from FULL to
DOWN, Neighbor Down: Interface down or detached

Rack1R5(config-subif)#do sh ip route ospf
     139.1.0.0/16 is variably subnetted, 8 subnets, 3 masks
O IA    139.1.0.0/20 [110/65] via 139.1.25.2, 00:05:15, Serial0/0.502
O IA    139.1.23.0/24 [110/128] via 139.1.25.2, 00:05:15, Serial0/0.502
     150.1.0.0/16 is variably subnetted, 7 subnets, 2 masks
O IA    150.1.7.7/32 [110/130] via 139.1.25.2, 00:05:15, Serial0/0.502
O IA    150.1.6.6/32 [110/130] via 139.1.25.2, 00:05:15, Serial0/0.502
O IA    150.1.3.3/32 [110/129] via 139.1.25.2, 00:05:15, Serial0/0.502
O       150.1.2.2/32 [110/65] via 139.1.25.2, 00:05:15, Serial0/0.502
```


Task 2.3 Solution

R3:
```
router ospf 1
 redistribute rip subnets
!
router rip
redistribute ospf 1 metric 1
auto-summary
```

R5:
```
router ospf 1
 redistribute rip subnets
```

SW3:
```
ip routing
!
router ospf 1
 network 139.1.11.254 0.0.0.0 area 0
```

SW4:
```
ip routing
!
router ospf 1
 network 139.1.2.22 0.0.0.0 area 0
```

Task 2.3 Breakdown

With RIP, auto-summarization is on by default, and will summarize to classful boundaries. If you disabled it during earlier RIP configuration, you can disable it for this step, so that R3 only sends the necessary routes. Since it is the default, "auto-summary" will not show up in the configuration under the RIP process.

Task 2.3 Verification

Verify that R3 sends the minimum required routing information to BB2:

```
Rack1R3#debug ip rip
RIP protocol debugging is on
Rack1R3#
RIP: sending v2 update to 224.0.0.9 via FastEthernet0/1 (192.10.1.3)
RIP: build update entries
   139.1.0.0/16 via 0.0.0.0, metric 1, tag 0
   150.1.0.0/16 via 0.0.0.0, metric 1, tag 0
   204.12.1.0/24 via 0.0.0.0, metric 1, tag 0
```

Verify that RIP routes are re-distributed into OSPF:

```
Rack1R1#show ip route ospf | i E2
O E2 222.22.2.0/24 [110/20] via 139.1.13.3, 00:06:04, Serial0/1
O E2 204.12.1.0/24 [110/20] via 139.1.15.5, 00:06:04, Serial0/0
O E2 220.20.3.0/24 [110/20] via 139.1.13.3, 00:06:04, Serial0/1
O E2    139.1.5.0/24 [110/20] via 139.1.15.5, 00:06:04, Serial0/0
O E2    139.1.45.4/32 [110/20] via 139.1.15.5, 00:06:04, Serial0/0
O E2    139.1.45.0/24 [110/20] via 139.1.15.5, 00:06:04, Serial0/0
O E2    139.1.58.0/24 [110/20] via 139.1.15.5, 00:06:04, Serial0/0
O E2    139.1.48.0/24 [110/20] via 139.1.15.5, 00:06:04, Serial0/0
O E2 192.10.1.0/24 [110/20] via 139.1.13.3, 00:06:04, Serial0/1
O E2    150.1.5.0/24 [110/20] via 139.1.15.5, 00:06:04, Serial0/0
O E2    150.1.4.0/24 [110/20] via 139.1.15.5, 00:06:04, Serial0/0
O E2    150.1.8.0/24 [110/20] via 139.1.15.5, 00:06:04, Serial0/0
O E2 205.90.31.0/24 [110/20] via 139.1.13.3, 00:06:04, Serial0/1
```

Check that SW3 and SW4 form OSPF neighbors:

```
Rack1SW3#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.1.1         1   FULL/DR         00:00:35    139.1.11.1      Vlan11

Rack1SW4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.2.2         1   FULL/DR         00:00:30    139.1.2.2       Vlan2
```


Finally, to ensure you have full internal connectivity run the following TCL script:

```
foreach i {
139.1.2.2
139.1.25.2
150.1.2.2
139.1.23.2
139.1.13.3
139.1.0.3
150.1.3.3
139.1.23.3
192.10.1.3
150.1.4.4
139.1.45.4
139.1.48.4
139.1.15.5
139.1.5.5
139.1.25.5
150.1.5.5
139.1.45.5
139.1.58.5
139.1.6.6
139.1.0.6
150.1.6.6
139.1.7.7
139.1.0.7
150.1.7.7
150.1.8.8
139.1.48.8
139.1.58.8
139.1.11.254
139.1.2.22

} { ping $i  }
```

Note that the Frame Relay link between R6 and BB1 is omitted from connectivity test.

Task 2.4 Solution

R4:

```
router bgp 100
 network 139.1.5.0 mask 255.255.255.0
 aggregate-address 139.1.0.0 255.255.0.0 summary-only
 neighbor 204.12.1.254 unsuppress-map UNSUPPRESS
 distribute-list prefix DENY_AGGREGATE in
!
ip prefix-list DENY_AGGREGATE seq 5 deny 139.1.0.0/16
ip prefix-list DENY_AGGREGATE seq 10 permit 0.0.0.0/0 le 32
!
ip prefix-list VLAN_5 seq 5 permit 139.1.5.0/24
!
route-map UNSUPPRESS permit 10
 match ip address prefix-list VLAN_5
```

R6:

```
router bgp 100
 network 139.1.6.0 mask 255.255.255.0
 aggregate-address 139.1.0.0 255.255.0.0 summary-only
 distribute-list prefix DENY_AGGREGATE in
!
ip prefix-list DENY_AGGREGATE seq 5 deny 139.1.0.0/16
ip prefix-list DENY_AGGREGATE seq 10 permit 0.0.0.0/0 le 32
```

R4:

```
router rip
 redistribute bgp 100 metric 1 route-map PERMIT_ODD
!
router bgp 100
 bgp router-id 150.1.5.5
 neighbor 204.12.1.254 route-map PERMIT_ODD in
!
ip access-list standard ODD
 permit 1.0.0.0 254.255.255.255
!
route-map PERMIT_ODD permit 10
 match ip address ODD
```


R5:

```
router rip
 redistribute ospf 1 metric 1 route-map OSPF_TO_RIP
!
route-map OSPF_TO_RIP permit 10
 match tag 6
```

R6:

```
router ospf 1
 redistribute bgp 100 subnets tag 6 route-map PERMIT_EVEN
!
router bgp 100
 neighbor 54.1.2.254 route-map PERMIT_EVEN in
!
ip access-list standard EVEN
 permit 0.0.0.0 254.255.255.255
!
route-map PERMIT_EVEN permit 10
 match ip address EVEN
```


Task 2.4 Breakdown

Start by adding a network to BGP and then configuring a summary on R4 and R6. In order for the more specific route for VLAN 5 to be sent, an unsuppress map is used along with the summary-only keyword on the aggregate, so that the more specific route is unsuppressed before sending to the backbone.

Additionally, if you are sending prefixes out to the backbones at multiple locations, you may want to consider filtering routes inbound, so that you do not learn the same route from another location. Filtering is implemented both on R4 and R6, so advertised routes are not injected back into the topology.

The BGP synchronization rule states that all iBGP learned routes must have a match in the IGP table in order to be considered for BGP best path selection. Although the BGP synchronization rule is rarely enabled in a production BGP environment, and is effectively considered legacy now, the problem that it was designed to prevent is still valid.

BGP synchronization is designed to prevent the case when non BGP speaking devices are in the transit path of the iBGP network. Since these transit devices are not running BGP, they must have an IGP route in order to send traffic to the final destination. Therefore, the BGP synchronization process first checks the IGP table to see if there is a match for all iBGP learned prefixes. If there are equal IGP matches in the IP routing table, synchronization has occurred, and the iBGP learned prefix can be considered for best path selection. However, if there is no matching IGP prefix for the iBGP prefix, synchronization has not occurred, and the iBGP learned prefix cannot be considered for best path selection.

In the above scenario, BGP synchronization is enabled on R4. Therefore any iBGP learned prefixes on R4 must have matching IGP routes in order to be considered valid. Therefore, BGP prefixes must be injected into the IGP domain in order for this case to occur.

There is an additional issue with OSPF. When you turn synchronization on, and redistribute BGP prefixes into OSPF, you should make sure that OSPF ASBR Router ID matches originating BGP Router ID. This is why we set Router ID of R4 to 150.1.5.5.

Task 2.4 Verification

Check routes that R4 and R6 advertise to BB3 and BB1:

```
Rack1R4#show ip bgp neighbors 204.12.1.254 advertised-routes
BGP table version is 15, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
r> 139.1.0.0        0.0.0.0                            32768 i
s> 139.1.5.0/24     139.1.45.5               2         32768 ?

Rack1R6#show ip bgp neighbors 54.1.2.254 advertised-routes
BGP table version is 14, 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
*> 139.1.0.0        0.0.0.0                            32768 i
```

Verify that R4 accepts only odd first octet prefixes from BB3:

```
Rack1R4#show ip bgp neighbors 204.12.1.254 routes
BGP table version is 21, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 113.0.0.0        204.12.1.254                           0 54 50 60 i
*> 115.0.0.0        204.12.1.254                           0 54 i
*> 117.0.0.0        204.12.1.254                           0 54 i
*> 119.0.0.0        204.12.1.254                           0 54 i
```


Confirm that R6 accepts only prefixes with even first octet from BB1:

```
Rack1R6#show ip bgp neighbors 54.1.2.254 routes
BGP table version is 18, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   54.1.2.254                             0 54 i
*> 28.119.17.0/24   54.1.2.254                             0 54 i
*> 112.0.0.0        54.1.2.254                0            0 54 50 60 i
*> 114.0.0.0        54.1.2.254                0            0 54 i
*> 116.0.0.0        54.1.2.254                0            0 54 i
*> 118.0.0.0        54.1.2.254                0            0 54 i
```

Next, verify the BGP redistribution:

```
Rack1R4#show ip route rip
R    118.0.0.0/8 [120/2] via 139.1.48.8, 00:00:01, FastEthernet0/1
                 [120/2] via 139.1.45.5, 00:00:00
R    116.0.0.0/8 [120/2] via 139.1.48.8, 00:00:01, FastEthernet0/1
                 [120/2] via 139.1.45.5, 00:00:00
     139.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
R       139.1.15.0/24 [120/2] via 139.1.48.8, 00:00:01, FastEthernet0/1
                      [120/2] via 139.1.45.5, 00:00:00
R       139.1.5.0/24 [120/2] via 139.1.48.8, 00:00:01, FastEthernet0/1
                     [120/2] via 139.1.45.5, 00:00:00
R       139.1.25.0/24 [120/2] via 139.1.48.8, 00:00:01, FastEthernet0/1
                      [120/2] via 139.1.45.5, 00:00:00
R       139.1.45.0/24 [120/2] via 139.1.48.8, 00:00:01, FastEthernet0/1
R       139.1.58.0/24 [120/1] via 139.1.48.8, 00:00:01, FastEthernet0/1
R    114.0.0.0/8 [120/2] via 139.1.48.8, 00:00:01, FastEthernet0/1
                 [120/2] via 139.1.45.5, 00:00:00
R    112.0.0.0/8 [120/2] via 139.1.48.8, 00:00:01, FastEthernet0/1
                 [120/2] via 139.1.45.5, 00:00:00
     28.0.0.0/24 is subnetted, 2 subnets
R       28.119.17.0 [120/2] via 139.1.48.8, 00:00:02, FastEthernet0/1
                    [120/2] via 139.1.45.5, 00:00:01
R       28.119.16.0 [120/2] via 139.1.48.8, 00:00:02, FastEthernet0/1
                    [120/2] via 139.1.45.5, 00:00:01
     150.1.0.0/24 is subnetted, 3 subnets
R       150.1.5.0 [120/2] via 139.1.48.8, 00:00:01, FastEthernet0/1
                  [120/2] via 139.1.45.5, 00:00:00
R       150.1.8.0 [120/1] via 139.1.48.8, 00:00:01, FastEthernet0/1
R*   0.0.0.0/0 [120/2] via 139.1.48.8, 00:00:01, FastEthernet0/1
               [120/2] via 139.1.45.5, 00:00:00
```



```
Rack1R6#show ip route ospf | include E2
O E2 119.0.0.0/8 [110/20] via 139.1.0.3, 00:04:58, FastEthernet0/0
O E2 222.22.2.0/24 [110/20] via 139.1.0.3, 00:05:01, FastEthernet0/0
O E2 204.12.1.0/24 [110/20] via 139.1.0.3, 00:05:01, FastEthernet0/0
O E2 117.0.0.0/8 [110/20] via 139.1.0.3, 00:04:58, FastEthernet0/0
O E2 220.20.3.0/24 [110/20] via 139.1.0.3, 00:05:01, FastEthernet0/0
O E2    139.1.5.0/24 [110/20] via 139.1.0.3, 00:05:01, FastEthernet0/0
O E2    139.1.45.4/32 [110/20] via 139.1.0.3, 00:05:01, FastEthernet0/0
O E2    139.1.45.0/24 [110/20] via 139.1.0.3, 00:05:01, FastEthernet0/0
O E2    139.1.58.0/24 [110/20] via 139.1.0.3, 00:05:01, FastEthernet0/0
O E2    139.1.48.0/24 [110/20] via 139.1.0.3, 00:05:01, FastEthernet0/0
O E2 115.0.0.0/8 [110/20] via 139.1.0.3, 00:04:58, FastEthernet0/0
O E2 113.0.0.0/8 [110/20] via 139.1.0.3, 00:04:58, FastEthernet0/0
O E2 192.10.1.0/24 [110/20] via 139.1.0.3, 00:05:01, FastEthernet0/0
O E2    150.1.5.0/24 [110/20] via 139.1.0.3, 00:05:01, FastEthernet0/0
O E2    150.1.4.0/24 [110/20] via 139.1.0.3, 00:05:03, FastEthernet0/0
O E2    150.1.8.0/24 [110/20] via 139.1.0.3, 00:05:03, FastEthernet0/0
O E2 205.90.31.0/24 [110/20] via 139.1.0.3, 00:05:03, FastEthernet0/0
```

Verify BGP synchronization:

```
Rack1R6#show ip bgp 115.0.0.0
BGP routing table entry for 115.0.0.0/8, version 22
Paths: (1 available, best #1, table Default-IP-Routing-Table, RIB-
failure(17))
  Advertised to update-groups:
     2
  54
    150.1.4.4 (metric 20) from 150.1.4.4 (150.1.5.5)
      Origin IGP, metric 0, localpref 100, valid, internal,
synchronized, best
```

```
Rack1R4#show ip bgp 116.0.0.0
BGP routing table entry for 116.0.0.0/8, version 16
Paths: (1 available, best #1, table Default-IP-Routing-Table, RIB-
failure(17))
  Advertised to update-groups:
     1
  54
    150.1.6.6 (metric 2) from 150.1.6.6 (150.1.6.6)
      Origin IGP, metric 0, localpref 100, valid, internal,
synchronized, best
```


Make a final verification by tracerouting to even numbered routes from R4 and odd from R6:

```
Rack1R4#traceroute 116.0.0.1

Type escape sequence to abort.
Tracing the route to 116.0.0.1

  1 139.1.48.8 4 msec
    139.1.45.5 16 msec
    139.1.48.8 8 msec
  2 139.1.25.2 28 msec
    139.1.58.5 12 msec
    139.1.25.2 32 msec
  3 139.1.25.2 24 msec
    139.1.23.3 44 msec
    139.1.25.2 28 msec
  4 139.1.0.6 44 msec
    139.1.23.3 36 msec
    139.1.0.6 40 msec
  5 139.1.0.6 40 msec
    54.1.2.254 60 msec
    139.1.0.6 40 msec
```

```
Rack1R6#traceroute 115.0.0.1

Type escape sequence to abort.
Tracing the route to 115.0.0.1

  1 139.1.0.3 4 msec 0 msec 0 msec
  2 139.1.23.2 16 msec 16 msec 12 msec
  3 139.1.25.5 32 msec 32 msec 28 msec
  4 139.1.45.4 44 msec 40 msec 44 msec
  5 204.12.1.254 44 msec 44 msec 44 msec
  6 172.16.4.1 36 msec *  32 msec
```

Task 2.5 Solution

R4:

```
router bgp 100
 neighbor 204.12.1.254 maximum-prefix 150000 90
```

R6:

```
router bgp 100
 neighbor 54.1.2.254 maximum-prefix 150000 90
```


Task 2.5 Breakdown

Large fluctuations in the BGP table can cause devices with limited amounts of memory to crash. These fluctuations usually occur either due to a misconfiguration, or a malicious attack on the BGP table. In order to prevent such a fluctuation from occurring, the **maximum-prefix** option on the BGP neighbor statement can be used to configure a threshold of received routes at which a BGP session will be reset.

Task 2.5 Verification

```
Rack1R6#show ip bgp neighbors 54.1.2.254 | begin Maximum prefixes
  Maximum prefixes allowed 150000
  Threshold for warning message 90%
  Number of NLRIs in the update sent: max 3, min 0
<output omitted>

Rack1R4#show ip bgp neighbors 204.12.1.254 | begin Maximum prefixes
  Maximum prefixes allowed 150000
  Threshold for warning message 90%
  Number of NLRIs in the update sent: max 0, min 0
<output omitted>
```

Task 3.1 Solution

R2:

```
interface FastEthernet0/0
 ipv6 ospf 1 area 1
!
interface Serial0/1
 ipv6 ospf 1 area 0

ipv6 router ospf 1
area 1 range 2001:CC1E:1:0::/62
```

R3:

```
interface FastEthernet0/0
 ipv6 ospf 1 area 0
!
!
interface Serial1/3
 ipv6 ospf 1 area 0
```

R6:

```
interface FastEthernet0/0
 ipv6 ospf 1 area 1
!
interface FastEthernet0/1
 ipv6 ospf 1 area 0

ipv6 router ospf 1
area 1 range 2001:CC1E:1:4::/62
```


Task 3.1 Verification

Configuring a summary will prevent R2 and R6 from seeing the original routes for each other's Fa0/0 interfaces. Verify the routes on R6, R3 and R2:

```
Rack1R2#show ipv6 route ospf
IPv6 Routing Table - 9 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:CC1E:1::/62 [110/0]
     via ::, Null0
O   2001:CC1E:1::/64 [110/65]
     via FE80::3, Serial0/1
OI  2001:CC1E:1:4::/62 [110/66]
     via FE80::3, Serial0/1

Rack1R3#show ipv6 route ospf
IPv6 Routing Table - 8 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
OI  2001:CC1E:1::/62 [110/782]
     via FE80::2, Serial1/3
OI  2001:CC1E:1:4::/62 [110/2]
     via FE80::6, FastEthernet0/0

Rack1R6#show ipv6 route ospf
IPv6 Routing Table - Default - 8 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
OI  2001:CC1E:1::/62 [110/783]
     via FE80::3, FastEthernet0/1
O   2001:CC1E:1:4::/62 [110/0]
     via Null0, directly connected
O   2001:CC1E:1:23::2/127 [110/782]
     via FE80::3, FastEthernet0/1
```


Verify IPv6 connectivity between R2 VLAN2 and R6 VLAN6:

```
Rack1R2#ping 2001:CC1E:1:6::6 source 2001:CC1E:1:2::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:6::6, timeout is 2
seconds:
Packet sent with a source address of 2001:CC1E:1:2::2
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/29/32 ms
```

Task 3.2 Solution

R6:

```
interface FastEthernet0/0
 ipv6 address 2001:CC1E:1:6::/64 eui-64
 ipv6 nd ra-interval 60
 ipv6 nd ra-lifetime 180
```

Task 3.2 Verification

Verify IPv6 ND RA configuration:

```
Rack1R6#show ipv6 interface FastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::6
  No Virtual link-local address(es):
  Global unicast address(es):
    2001:CC1E:1:6::6, subnet is 2001:CC1E:1:6::/64 [EUI]
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::5
    FF02::6
    FF02::1:FF00:6
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ICMP unreachables are sent
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds (using 43942)
  ND advertised reachable time is 0 (unspecified)
  ND advertised retransmit interval is 0 (unspecified)
  ND router advertisements are sent every 60 seconds
  ND router advertisements live for 180 seconds
  ND advertised default router preference is Medium
  Hosts use stateless autoconfig for addresses.
```


Task 3.3 Solution

R3:

```
interface Tunnel345
 ipv6 address 2001:CC1E:1:345::/64 eui-64
 tunnel source Loopback0
 tunnel mode ipv6ip isatap
!
interface Loopback100
 ipv6 address 2001:CC1E:1:3::3/64
!
! The next hop IPv6 address could be found using the command
! show ipv6 interface tunnel345 on R4 and R5
! they are based on the respective tunnel's IPv4 source addresses
!
ipv6 route 2001:CC1E:1:4::/64 2001:CC1E:1:345:0:5EFE:9601:404
ipv6 route 2001:CC1E:1:5::/64 2001:CC1E:1:345:0:5EFE:9601:505
!
ipv6 router ospf 1
 redistribute static
```

R4:

```
ipv6 unicast-routing
!
interface Tunnel345
 ipv6 address 2001:CC1E:1:345::/64 eui-64
 tunnel source Loopback0
 tunnel mode ipv6ip isatap
!
interface Loopback100
 ipv6 address 2001:CC1E:1:4::4/64
!
ipv6 route ::/0 2001:CC1E:1:345:0:5EFE:9601:303
!
```

R5:

```
ipv6 unicast-routing
!
interface Tunnel345
 ipv6 address 2001:CC1E:1:345::/64 eui-64
 tunnel source Loopback0
 tunnel mode ipv6ip isatap
!
interface Loopback100
 ipv6 address 2001:CC1E:1:5::5/64
 ipv6 ospf 1 area 1
 ipv6 ospf network point-to-point
!
ipv6 route ::/0 2001:CC1E:1:345:0:5EFE:9601:303
```


Task 3.3 Verification

Check R6's and R2's routing table to see if static ipv6 routes are re-distributed:

```
Rack1R6#show ipv6 route ospf
IPv6 Routing Table - Default - 10 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
OI  2001:CC1E:1::/62 [110/783]
     via FE80::3, FastEthernet0/1
O   2001:CC1E:1:4::/62 [110/0]
     via Null0, directly connected
OE2 2001:CC1E:1:4::/64 [110/20]
     via FE80::3, FastEthernet0/1
OE2 2001:CC1E:1:5::/64 [110/20]
     via FE80::3, FastEthernet0/1
O   2001:CC1E:1:23::2/127 [110/782]
     via FE80::3, FastEthernet0/1

Rack1R6#ping 2001:CC1E:1:5::5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:5::5, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 104/104/108
ms

Rack1R6#ping 2001:CC1E:1:4::4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:4::4, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 104/104/104
```



```
Rack1R2#show ipv6 route ospf
IPv6 Routing Table - 11 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:CC1E:1::/62 [110/0]
     via ::, Null0
O   2001:CC1E:1::/64 [110/65]
     via FE80::3, Serial0/1
OI  2001:CC1E:1:4::/62 [110/66]
     via FE80::3, Serial0/1
OE2  2001:CC1E:1:4::/64 [110/20]
     via FE80::3, Serial0/1
OE2  2001:CC1E:1:5::/64 [110/20]
     via FE80::3, Serial0/1

Rack1R2#ping 2001:CC1E:1:5::5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:5::5, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 128/129/132
ms

Rack1R2#ping 2001:CC1E:1:4::4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:4::4, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 128/131/133
ms
```


Task 5.1 Solution

R3:
```
interface Tunnel35
 ip unnumbered FastEthernet0/0
 ip pim dense-mode
 tunnel source Loopback0
 tunnel destination 150.1.5.5
```

R5:
```
interface Tunnel35
 ip unnumbered FastEthernet0/0
 ip pim dense-mode
 tunnel source Loopback0
 tunnel destination 150.1.3.3
!
ip mroute 0.0.0.0 0.0.0.0 Tunnel35
```

Task 5.1 Breakdown

The above scenario uses a GRE tunnel to tunnel multicast traffic across non-PIM speaking neighbors. As the tunnel interface is based on the loopback interfaces of R3 and R5, R1 (the non-PIM speaking device) only sees unicast GRE traffic between these loopback interfaces. Therefore, as long as the transit devices have unicast reachability throughout the network, they can be used to transport multicast traffic.

Task 5.1 Verification

Join multicast groups 239.2.2.2 with R2 FastEthernet0/0 and 239.5.5.5 with R5 FastEthernet 0/0:

R2:
```
!
interface FastEthernet0/0
 ip igmp join-group 239.2.2.2
```

R5:
```
!
interface FastEthernet0/0
 ip igmp join-group 239.5.5.5
```

Enable mpacket debugging at R3:

```
Rack1R3#debug ip mpacket
IP multicast packets debugging is on

Rack1R3(config)#interface FastEthernet0/0
Rack1R3(config-if)#no ip mroute-cache
```


Simulate multicast traffic from R6 to 239.2.2.2, add the Fa0/1 interface on R6 as a PIM dense mode interface to test.

```
Rack1R6#ping 239.2.2.2 repeat 6

Type escape sequence to abort.
Sending 6, 100-byte ICMP Echos to 239.2.2.2, timeout is 2 seconds:

Reply to request 0 from 139.1.23.2, 32 ms
Reply to request 1 from 139.1.23.2, 32 ms
Reply to request 2 from 139.1.23.2, 32 ms
Reply to request 3 from 139.1.23.2, 32 ms
Reply to request 4 from 139.1.23.2, 32 ms
Reply to request 5 from 139.1.23.2, 36 ms
```

Look at R3's debugging output:

```
IP(0): s=139.1.0.6 (FastEthernet0/0) d=239.2.2.2 (Serial1/3) id=22,
ttl=254, prot=1, len=100(100), mforward
Rack1R3#
IP(0): s=139.1.0.6 (FastEthernet0/0) d=239.2.2.2 (Serial1/3) id=23,
ttl=254, prot=1, len=100(100), mforward
Rack1R3#
IP(0): s=139.1.0.6 (FastEthernet0/0) d=239.2.2.2 (Serial1/3) id=24,
ttl=254, prot=1, len=100(100), mforward
Rack1R3#
IP(0): s=139.1.0.6 (FastEthernet0/0) d=239.2.2.2 (Serial1/3) id=25,
ttl=254, prot=1, len=100(100), mforward
Rack1R3#
IP(0): s=139.1.0.6 (FastEthernet0/0) d=239.2.2.2 (Serial1/3) id=26,
ttl=254, prot=1, len=100(100), mforward
Rack1R3#
IP(0): s=139.1.0.6 (FastEthernet0/0) d=239.2.2.2 (Serial1/3) id=27,
ttl=254, prot=1, len=100(100), mforward

Rack1R3#show ip mroute 239.2.2.2
IP Multicast Routing Table
<snip>

(*, 239.2.2.2), 00:00:20/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Tunnel35, Forward/Dense, 00:00:20/00:00:00
    Serial1/3, Forward/Dense, 00:00:20/00:00:00

(139.1.0.6, 239.2.2.2), 00:00:20/00:02:58, flags: T
  Incoming interface: FastEthernet0/0, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial1/3, Forward/Dense, 00:00:21/00:00:00
    Tunnel35, Prune/Dense, 00:00:20/00:02:39
```


Next, enable additional debugging at R3, and send multicast traffic from R6 to 239.5.5.5:

```
Rack1R6#ping 239.5.5.5 repeat 6

Type escape sequence to abort.
Sending 6, 100-byte ICMP Echos to 239.5.5.5, timeout is 2 seconds:

Reply to request 0 from 139.1.5.5, 68 ms
Reply to request 1 from 139.1.5.5, 68 ms
Reply to request 2 from 139.1.5.5, 80 ms
Reply to request 3 from 139.1.5.5, 68 ms
Reply to request 4 from 139.1.5.5, 68 ms
Reply to request 5 from 139.1.5.5, 88 ms

Rack1R3#debug ip packet detail 100
IP packet debugging is on (detailed) for access list 100
```

Note how GRE traffic is load balanced. There are two debugs running on R3: debug ip mpacket and debug ip packet detail for the GRE traffic.

```
Rack1R3#
IP(0): s=139.1.0.6 (FastEthernet0/0) d=239.5.5.5 (Tunnel35) id=46,
ttl=254, prot=1, len=100(100), mforward
IP: s=150.1.3.3 (Tunnel35), d=150.1.5.5 (Serial1/2), len 124, sending,
proto=47
IP(0): s=139.1.0.6 (FastEthernet0/0) d=239.5.5.5 (Tunnel35) id=47,
ttl=254, prot=1, len=100(100), mforward
IP: s=150.1.3.3 (Tunnel35), d=150.1.5.5 (Serial1/2), len 124, sending,
proto=47
IP(0): s=139.1.0.6 (FastEthernet0/0) d=239.5.5.5 (Tunnel35) id=48,
ttl=254, prot=1, len=100(100), mforward
IP: s=150.1.3.3 (Tunnel35), d=150.1.5.5 (Serial1/3), len 124, sending,
proto=47
IP(0): s=139.1.0.6 (FastEthernet0/0) d=239.5.5.5 (Tunnel35) id=49,
ttl=254, prot=1, len=100(100), mforward
IP: s=150.1.3.3 (Tunnel35), d=150.1.5.5 (Serial1/2), len 124, sending,
proto=47
IP(0): s=139.1.0.6 (FastEthernet0/0) d=239.5.5.5 (Tunnel35) id=50,
ttl=254, prot=1, len=100(100), mforward
IP: s=150.1.3.3 (Tunnel35), d=150.1.5.5 (Serial1/3), len 124, sending,
proto=47
IP: s=150.1.3.3 (Tunnel35), d=150.1.5.5 (Serial1/2), len 78, sending,
proto=47
IP(0): s=139.1.0.6 (FastEthernet0/0) d=239.5.5.5 (Tunnel35) id=51,
ttl=254, prot=1, len=100(100), mforward
IP: s=150.1.3.3 (Tunnel35), d=150.1.5.5 (Serial1/3), len 124, sending,
proto=47
```


R2:
```
interface FastEthernet0/0
 no ip igmp join-group 239.2.2.2
 ip mroute-cache
```

R5:
```
interface FastEthernet0/0
 no ip igmp join-group 239.5.5.5
```

R3:
```
undebug all
no access-list 100
```

Task 5.2 Solution

R1, R2:
```
ip multicast rpf backoff 10 1000
ip multicast route-limit 100
```

Task 5.2 Breakdown

Here, we are just modifying some miscellaneous settings for R1 and R2. We aren't given a minimum value, so you can pick something arbitrarily for the RPF backoff.

Task 5.2 Verification

```
Rack1R2#show ip rpf events | section RPF back|RPF max
RPF backoff delay: 10 msec
RPF maximum delay: 1 sec
```

```
Rack1R1#show ip rpf events | section RPF back|RPF max
RPF backoff delay: 10 msec
RPF maximum delay: 1 sec
```


Task 6.1 Solution

R3:

```
interface FastEthernet0/1
 ip access-group FILTER_IN in
 ip access-group FILTER_OUT out
 no ip unreachables
!
ip access-list extended FILTER_IN
 deny   icmp any any echo log
 permit ip any any
!
ip access-list extended FILTER_OUT
 deny   icmp any any time-exceeded log
 deny   icmp any any port-unreachable log
 permit ip any any
```

R4:

```
interface FastEthernet0/0
 ip access-group FILTER_IN in
 ip access-group FILTER_OUT out
 no ip unreachables
!
ip access-list extended FILTER_IN
 deny   icmp any any echo log
 permit ip any any
!
ip access-list extended FILTER_OUT
 deny   icmp any any time-exceeded log
 deny   icmp any any port-unreachable log
 permit ip any any
```

Task 6.1 Breakdown

Double check the ACL, and make sure that you have a “permit any” at the end, so that you are not dropping any legitimate traffic. Blocking the ICMP echo traffic will affect ping testing for connectivity. If you are checking connectivity at the end of the lab, make sure to take note of any situations like this where you are specifically asked to block the traffic.

Task 6.1 Verification

```
Rack1R3#show ip interface fastEthernet 0/1 | section
Inbound|Outgoing|ICMP
  Outgoing access list is FILTER_OUT
  Inbound  access list is FILTER_IN
  ICMP redirects are always sent
  ICMP unreachables are never sent
  ICMP mask replies are never sent

Rack1R3#show access-lists
Extended IP access list FILTER_IN
    10 deny icmp any any echo log
    20 permit ip any any (27 matches)
Extended IP access list FILTER_OUT
    10 deny icmp any any time-exceeded log
    20 deny icmp any any port-unreachable log
    30 permit ip any any
```

Task 6.2 Solution

```
R5:
ip inspect tcp synwait-time 10
ip inspect name INTERCEPT tcp
!
interface FastEthernet 0/0
 ip inspect INTERCEPT out
```

Task 6.2 Verification

```
Rack1R5#show ip inspect all
Session audit trail is disabled
Session alert is enabled
one-minute (sampling period) thresholds are [400:500] connections
max-incomplete sessions thresholds are [400:500]
max-incomplete tcp connections per host is 50. Block-time 0 minute.
tcp synwait-time is 10 sec -- tcp finwait-time is 5 sec
tcp idle-time is 3600 sec -- udp idle-time is 30 sec
dns-timeout is 5 sec
Inspection Rule Configuration
 Inspection name INTERCEPT
    tcp alert is on audit-trail is off timeout 3600

Interface Configuration
 Interface FastEthernet0/0
  Inbound inspection rule is not set
  Outgoing inspection rule is INTERCEPT
    tcp alert is on audit-trail is off timeout 3600
  Inbound access list is not set
  Outgoing access list is not set
```


Task 6.3 Solution

R5:

```
ip inspect max-incomplete low 81
ip inspect max-incomplete high 100
!
ip inspect one-minute low 40
ip inspect one-minute high 60
!
ip inspect tcp max-incomplete host 20 block-time 2
ip inspect tcp finwait-time 2
```

Task 6.3 Verification

```
Rack1R5#show ip inspect config
Session audit trail is disabled
Session alert is enabled
one-minute (sampling period) thresholds are [40:60] connections
max-incomplete sessions thresholds are [81:100]
max-incomplete tcp connections per host is 20. Block-time 2 minutes.
tcp synwait-time is 10 sec -- tcp finwait-time is 2 sec
tcp idle-time is 3600 sec -- udp idle-time is 30 sec
dns-timeout is 5 sec
Inspection Rule Configuration
 Inspection name INTERCEPT
    tcp alert is on audit-trail is off timeout 3600
```

Task 6.3 Breakdown

Watch your thresholds carefully. Thresholds need to be crossed. For the rising thresholds 100 and 60, the wording in the section is exceeds and above. For the one minute falling, the section says below. For the incomplete threshold, the section states “reaches 80”. Since the threshold of 80 would not be crossed until it dropped below 80, setting the threshold to 81 will allow the clamping to stop when that threshold is crossed, and the number of connections falls to 80.

Task 6.4 Solution

SW1:

```
ip dhcp snooping vlan 367
ip dhcp snooping
!
interface FastEthernet 0/3
 ip dhcp snooping trust
```

R3:

```
ip dhcp relay information policy keep
!
interface fastEthernet 0/0
 ip dhcp relay information trusted
```

Task 6.4 Breakdown

This task needs to be correlated with other DHCP-related scenarios in this lab, specifically Task 7.4. As soon as DHCP snooping is enabled in SW1, it will insert information option and empty "giaddr" field in all relayed DHCP requests. In order for R3, which will be configured as a DHCP relay later, to accept this information, the command **ip dhcp relay information trusted** is configured on R3's VLAN 367 interface.

At the same time, it is required that R3 keeps relay information inserted by SW1. The default IOS behavior is to replace it when forwarding DHCP messages. This is why **ip dhcp relay information policy keep** is configured globally on R3.

Task 6.4 Verification

```
Rack1SW1#show ip dhcp snooping
Switch DHCP snooping is enabled
DHCP snooping is configured on following VLANs:
367
Insertion of option 82 is enabled
   circuit-id format: vlan-mod-port
    remote-id format: MAC
Option 82 on untrusted port is not allowed
Verification of hwaddr field is enabled
Interface                    Trusted     Rate limit (pps)
------------------------     -------     ----------------
FastEthernet0/3              yes         unlimited

Rack1R1#show ip dhcp relay info trust
All interfaces are trusted source of relay agent information option
```

Note: With the earlier configuration as shown, the helper address is tied to the active HSRP device. For testing, you can create an access list to filter debugging as shown below:

```
R3:
ip access-list 102 permit udp any any range 67 68

Rack1R3#debug ip packet 102 detail
Rack1R3#debug ip dhcp server
```

First, take a look at the output when R3 is not active. It receives the DHCP request, but does not forward.

```
19:03:39.982: IP: s=0.0.0.0 (FastEthernet0/0), d=255.255.255.255, len
344, rcvd 2
 19:03:39.982:     UDP src=68, dst=67
 19:03:39.982: DHCPD: message is from trusted interface FastEthernet0/0
```

Next, take a look at how the output changes when R3 is active for the HSRP group. For this test, R6's FastEthernet interface has been shut down, and R3 has been given time to take over for the HSRP group.


```
19:35:42.642: IP: s=0.0.0.0 (FastEthernet0/0), d=255.255.255.255, len
362, rcvd 2
 19:35:42.642:     UDP src=68, dst=67
 19:35:42.646: DHCPD: message is from trusted interface FastEthernet0/0
 19:35:42.646: DHCPD: Finding a relay for client
0063.6973.636f.2d30.3031.322e.3030.6630.2e62.3861.302d.4661.302f.30 on
interface FastEthernet0/0.
 19:35:42.646: DHCPD: setting giaddr to 139.1.0.3.
 19:35:42.650: IP: tableid=0, s=139.1.0.3 (local), d=139.1.13.1
(Serial1/2), routed via FIB
 19:35:42.650: IP: s=139.1.0.3 (local), d=139.1.13.1 (Serial1/2), len
362, sending
 19:35:42.650:     UDP src=67, dst=67
 19:35:42.650: DHCPD: BOOTREQUEST from
0063.6973.636f.2d30.3031.322e.3030.6630.2e62.3861.302d.4661.302f.30
forwarded to 139.1.13.1.
 19:35:42.758: IP: tableid=0, s=139.1.13.1 (Serial1/2), d=139.1.0.3
(FastEthernet0/0), routed via RIB
 19:35:42.758: IP: s=139.1.13.1 (Serial1/2), d=139.1.0.3, len 385, rcvd
4
 19:35:42.758:     UDP src=67, dst=67
 19:35:42.758: DHCPD: forwarding BOOTREPLY to client 0012.00f0.b8a0.
 19:35:42.762: DHCPD: broadcasting BOOTREPLY to client 0012.00f0.b8a0.
 19:35:42.762: IP: s=139.1.0.3 (local), d=255.255.255.255
(FastEthernet0/0), len 385, sending broad/multicast
 19:35:42.762:     UDP src=67, dst=68
Rack1R3#
```


Task 6.5 Solution

R5:

```
ip domain-name INE.com
username cisco password cisco
crypto key generate rsa modulus 1024
!
object-group network TELSSH
 150.1.1.1 /32
 150.1.2.2 /32
 150.1.3.3 /32
 150.1.4.4 /32
 150.1.7.7 /32
 150.1.8.8 /32
!
access-list 105 permit tcp obj TELSSH any range 22 23
!
line vty 0 807
 access-class 105 in
 login local
```

Task 6.5 Verification

Try to telnet from various addresses. Attempting from R6's lo0 should be blocked, as well as from R4 when not sourcing from the loopback0 interface.

```
Rack1R6#telnet 150.1.6.6 /source-interface loopback 0
Trying 150.1.6.6 ...
% Connection refused by remote host

Rack1R1#telnet 150.1.6.6
Trying 150.1.6.6 ...
% Connection refused by remote host


Rack1R1#telnet 150.1.6.6 /source-interface loopback 0
Trying 150.1.6.6 ... Open


User Access Verification

Username: cisco
Password:
Rack1R6>
```


Task 7.1 Solution

R6:

```
snmp-server enable traps bgp
snmp-server host 139.1.2.100 CISCOBGP
```

R3 and R4:

```
logging 139.1.5.100
logging facility local6
```

Task 7.1 Verification

```
Rack1R3#show logging | section Trap
    Trap logging: level informational, 85 message lines logged
      Logging to 139.1.5.100 (udp port 514, audit disabled, link up), 2
      message lines logged, xml disabled,
                filtering disabled

Rack1R4#show loggin | section Trap
    Trap logging: level informational, 86 message lines logged
        Logging to 139.1.5.100  (udp port 514,  audit disabled,
              authentication disabled, encryption disabled, link up),
              2 message lines logged,
              0 message lines rate-limited,
              0 message lines dropped-by-MD,
              xml disabled, sequence number disabled
              filtering disabled

Rack1R6#show snmp host
Notification host: 139.1.2.100  udp-port: 162   type: trap
user: CISCOBGP  security model: v1
```

Task 7.2 Solution

R6:

```
interface FastEthernet0/0
 ip nbar protocol-discovery
```

R5:

```
flow monitor TEST
 statistics packet protocol
 statistics packet size
 record netflow ipv4 protocol-port-tos

interface fastEthernet 0/1
 ip flow monitor TEST output
 ip accounting output-packets
```


Task 7.2 Verification

To see how NBAR collects statistics enable NBAR on interface FastEthernet 0/0:

```
Rack1R6#show ip nbar protocol-discovery interface Fa0/0 top-n 3

 FastEthernet0/0

 Last clearing of "show ip nbar protocol-discovery" counters 00:00:44

                         Input                     Output
                         -----                     ------
   Protocol              Packet Count              Packet Count
                         Byte Count                Byte Count
                         5min Bit Rate (bps)       5min Bit Rate (bps)
                         5min Max Bit Rate (bps)   5min Max Bit Rate (bps)
   ----------------------- ------------------------ ------------------
   ospf                        0                         5
                               0                         450
                               0                         0
                               0                         0
   bgp                         0                         0
                               0                         0
                               0                         0
                               0                         0
   bittorrent                  0                         0
                               0                         0
                               0                         0
                               0                         0
   unknown                     0                         0
                               0                         0
                               0                         0
                               0                         0
   Total                       0                         5
                               0                         450
                               0                         0
                               0                         0
```


Alternatively, IP accounting and Netflow can also be used to gather traffic statistics, as shown on R4's configuration. Generate some transit traffic to test.

```
Rack1R4#show flow mon TEST statistics
  Cache type:                               Normal
  Cache size:                                 4096
  Current entries:                               0
  High Watermark:                                1

  Flows added:                                   1
  Flows aged:                                    1
    - Active timeout    (  1800 secs)            0
    - Inactive timeout (    15 secs)            1
    - Event aged                                 0
    - Watermark aged                             0
    - Emergency aged                             0

Packet size distribution (14 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416
   .000 .000 .000 1.00 .000 .000 .000 .000 .000 .000 .000 .000 .000

    448  480  512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

Protocol     Total    Flows   Packets Bytes Packets Active(Sec) Idle(Sec)
--------     Flows     /Sec     /Flow  /Pkt    /Sec       /Flow     /Flow
ICMP             1      0.0        14   100      0.0        2.3      15.3
Total:           1      0.0        14   100      0.0        2.3      15.3

Rack1R4#show ip accounting
   Source            Destination              Packets                Bytes
 204.12.1.254      139.1.48.8                      15                 1500

Accounting data age is 3
```

Task 7.3 Solution

R1:

```
ip dhcp excluded-address 139.1.45.0 139.1.45.3
ip dhcp excluded-address 139.1.45.5 139.1.45.255
!
ip dhcp pool R4
  network 139.1.45.0 255.255.255.0
!
ip route 139.1.45.5 255.255.255.255 139.1.15.5
```

R5:

```
no ip dhcp-server 139.1.11.100
ip dhcp-server 139.1.15.1
```

Quick Note
Task states that installed server is not valid. Use R1 instead.

Task 7.3 Breakdown

Verification for this task is shown with Task 1.2. Make sure to exclude the addresses before defining the address pool.

Task 7.4 Solution

R1:

```
ip dhcp excluded-address 139.1.0.0 139.1.0.99
ip dhcp excluded-address 139.1.0.201 139.1.0.255
!
ip dhcp pool VLAN_367
 network 139.1.0.0 255.255.255.0
 default-router 139.1.0.1
 domain-name InternetworkExpert.com
 lease infinite
!
```

R3:

```
!
interface FastEthernet0/0
 standby 1 name HSRP
 ip helper-address 139.1.13.1 redundancy HSRP
 standby 1 ip 139.1.0.1
 standby 1 preempt
```

R6:

```
interface FastEthernet0/1
 standby 1 name HSRP
 ip helper-address 139.1.13.1 redundancy HSRP
 standby 1 ip 139.1.0.1
 standby 1 priority 101
 standby 1 preempt
```

Task 7.4 Verification

Verify the standby configuration:

```
Rack1R6#show standby
FastEthernet0/1 - Group 1
  State is Active
    2 state changes, last state change 00:00:34
  Virtual IP address is 139.1.0.1
  Active virtual MAC address is 0000.0c07.ac01
    Local virtual MAC address is 0000.0c07.ac01 (v1 default)
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 0.480 secs
  Preemption enabled
  Active router is local
  Standby router is 139.1.0.3, priority 100 (expires in 9.008 sec)
  Priority 101 (configured 101)
  Group name is "HSRP" (cfgd)
```


Verify DHCP address assignment and the redundancy configuration:

Use SW2 to simulate a host in VLAN367:

```
Rack1SW2(config)#interface vl367
%LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan367, changed state
to up

Rack1SW2(config-if)#ip address dhcp
Rack1SW2(config-if)#
DHCP: DHCP client process started: 10
RAC: Starting DHCP discover on Vlan367
DHCP: Try 1 to acquire address for Vlan367
DHCP: allocate request
DHCP: new entry. add to queue
DHCP: SDiscover attempt # 1 for entry:
DHCP: SDiscover: sending 300 byte length DHCP packet
DHCP: SDiscover 300 bytes
           B'cast on Vlan367 interface from 0.0.0.0
DHCP: SDiscover attempt # 2 for entry:
DHCP: SDiscover: sending 300 byte length DHCP packet
DHCP: SDiscover 300 bytes
           B'cast on Vlan367 interface from 0.0.0.0
DHCP: Received a BOOTREP pkt
DHCP: offer received from 139.1.13.1
DHCP: SRequest attempt # 1 for entry:
DHCP: SRequest- Server ID option: 139.1.13.1
DHCP: SRequest- Requested IP addr option: 139.1.0.100
DHCP: SRequest placed lease len option: 4294967295
DHCP: SRequest: 318 bytes
DHCP: SRequest: 318 bytes
           B'cast on Vlan367 interface from 0.0.0.0
DHCP: Received a BOOTREP pkt
DHCP: offer received from 139.1.13.1
DHCP: offer received in bad state: Requesting  punt
DHCP: Received a BOOTREP pkt
DHCP: offer received from 139.1.13.1
DHCP: offer received in bad state: Requesting  punt
DHCP: Received a BOOTREP pkt
DHCP: offer received from 139.1.13.1
DHCP: offer received in bad state: Requesting  punt
DHCP: Received a BOOTREP pkt
Interface Vlan367 assigned DHCP address 139.1.0.100, mask 255.255.255.0

DHCP Client Pooling: ***Allocated IP address: 139.1.0.100
DHCP: Received a BOOTREP pkt
DHCP: rcv ack in Bound state: punt
Allocated IP address = 139.1.0.100  255.255.255.0
```



```
Rack1R1#show ip dhcp binding
Bindings from all pools not associated with VRF:
IP address          Client-ID/              Lease expiration
Type
                    Hardware address/
                    User name
139.1.0.100         0063.6973.636f.2d30.    Infinite
Automatic
                    3030.662e.3866.6232.
                    2e65.3830.302d.566c.
                    3336.37
139.1.45.4          0063.6973.636f.2d31.    Mar 02 1993 01:24 AM
Automatic
                    3339.2e31.2e34.352e.
                    352d.5365.7269.616c.
                    302f.31

Rack1R6(config)#interface Fa0/1
Rack1R6(config-if)#shutdown

Rack1R3#show standby
FastEthernet0/0 - Group 1
  State is Active
    5 state changes, last state change 00:00:18
  Virtual IP address is 139.1.0.1
  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 2.412 secs
  Preemption enabled
  Active router is local
  Standby router is unknown
  Priority 100 (default 100)
  IP redundancy name is "HSRP" (cfgd)

Rack1SW2#ping 139.1.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 139.1.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms

R6:
interface Fa0/1
 no shutdown

SW2:
no interface vl367
```


Task 7.4 Breakdown

R1 is supposed to hand out addresses for VLAN367, but is not directly connected. R3 and R6 can forward the traffic by using a helper address. By tying the helper address to the HSRP group name with the redundancy keyword, only the active HSRP device will forward the traffic. Configuring HSRP will allow one device to take over for the other and act as the gateway. Make sure to have R1 configured with the HSRP address as the gateway. The section also states to not rely on client specific methods. If that was not a restriction, two methods that could be used would be specifying multiple addresses for the default router option on the DHCP scope or IRDP. With IRDP, the end devices need to be IRDP-aware. With multiple default routers specified, the clients need to determine that the first one is unreachable and decide to use the next one.

Task 7.5 Solution

SW1 and SW2:

```
logging file flash:log.txt informational
```

Task 7.5 Verification

```
Rack1SW2#show logging
Syslog logging: enabled (0 messages dropped, 3 messages rate-limited, 0
flushes, 0 overruns, xml disabled, filtering disabled)
    Console logging: level debugging, 58 messages logged, xml disabled,
                     filtering disabled
    Monitor logging: level debugging, 0 messages logged, xml disabled,
                     filtering disabled
    Buffer logging: level debugging, 60 messages logged, xml disabled,
                    filtering disabled
    Exception Logging: size (4096 bytes)
    Count and timestamp logging messages: disabled
    File logging: file flash:log.txt,
        max size 0, min size 0,
        level informational, 1 messages logged
    Trap logging: level informational, 63 message lines logged
<output omitted>
```


Task 7.6 Solution

R3:

```
interface FastEthernet 0/0
 ip irdp
 ip irdp address 139.1.0.3 1000
 ip irdp preference 1000
 ip irdp maxadvertinterval 30
 ip irdp minadvertinterval 10
```

R6:

```
!
interface FastEthernet 0/1
 ip irdp
 ip irdp address 139.1.0.6 500
 ip irdp preference 500
 ip irdp maxadvertinterval 30
 ip irdp minadvertinterval 10
```

SW1:

```
!
interface Vlan 367
 ip irdp
 ip irdp address 139.1.0.7 500
 ip irdp preference 500
 ip irdp maxadvertinterval 30
 ip irdp minadvertinterval 10
```


Task 7.6 Breakdown

IRDP works as following:

- Broadcast periodic IRDP advertisements
- Default minimum interval between advertisements is 450 seconds
- Default maximum interval between advertisements is 600 seconds
- Default preference is 0

We need to change the advertisements intervals and the preference value for R3 to make it preferred; higher value means higher preference. The preference value can be changed in two ways:

- Using **ip irdp preference <value>** command. The router will advertise its own IP address with the configured preference value
- Using **ip irdp address x.x.x.x <value>** command. The router will advertise the IP address with the configured preference value and its own IP address with the default preference value of zero.

Task 7.6 Verification

```
Rack1SW1#show ip irdp vlan 367
Vlan367 has router discovery enabled

Advertisements will occur between every 10 and 30 seconds.
Advertisements are sent with broadcasts.
Advertisements are valid for 90 seconds.
Default preference will be 0.
Proxy for 139.1.0.7 with preference 500.

Rack1R3#show ip irdp fastEthernet 0/0
FastEthernet0/0 has router discovery enabled

Advertisements will occur between every 10 and 30 seconds.
Advertisements are sent with broadcasts.
Advertisements are valid for 90 seconds.
Default preference will be 0.
Proxy for 139.1.0.3 with preference 1000.

Rack1R6#show ip irdp fastEthernet 0/1
FastEthernet0/1 has router discovery enabled

Advertisements will occur between every 10 and 30 seconds.
Advertisements are sent with broadcasts.
Advertisements are valid for 90 seconds.
Default preference will be 0.
Proxy for 139.1.0.6 with preference 500.
```


Task 8.1 Solution

R2:

```
access-list 101 permit udp any any
access-list 102 permit tcp any any
!
 class-map match-all ICMP
  match protocol icmp
!
 class-map match-all UDP
  match access-group 101
!
class-map match-all TCP
  match access-group 102
!
 policy-map MQC_CAR
  class ICMP
   drop
  class UDP
   police cir 128000 bc 2000
     conform-action transmit
     exceed-action set-prec-transmit 0
  class TCP
   police cir 256000 bc 4000
     conform-action transmit
     exceed-action set-prec-transmit 0
!
interface FastEthernet0/0
 service-policy input MQC_CAR
```


Task 8.1 Verification

Verify the policy map application on the interface. For ICMP, you can match with the “match protocol ICMP” rather than by using an access list. Since both the conform action and exceed actions are both drop, you can use the MQC ‘drop’ keyword for the traffic in that class.

```
Rack1R2#show policy-map int fa0/0
 FastEthernet0/0

  Service-policy input: MQC_CAR

    Class-map: ICMP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol icmp
      drop

    Class-map: UDP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group 101
      police:
          cir 128000 bps, bc 2000 bytes
        conformed 0 packets, 0 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          set-prec-transmit 0
        conformed 0 bps, exceed 0 bps

    Class-map: TCP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group 102
      police:
          cir 256000 bps, bc 4000 bytes
        conformed 0 packets, 0 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          set-prec-transmit 0
        conformed 0 bps, exceed 0 bps

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
Rack1R2#
```


Task 8.2 Solution

R5:

```
class-map match-all HTTP_RESPONSES
match access-group name HTTP_RESPONSES
!
!
policy-map DLCI_501
class HTTP_RESPONSES
bandwidth percent 80
!
interface Serial0/0/0
 bandwidth 384
 bandwidth inherit
 frame-relay traffic-shaping
!
interface Serial0/0/0.501 point-to-point
 frame-relay class DLCI_501
!
ip access-list extended HTTP_RESPONSES
permit tcp any eq www 443 139.1.11.0 0.0.0.255
!
map-class frame-relay DLCI_501
 frame-relay cir 384000
 frame-relay mincir 384000
 service-policy output DLCI_501
```

Task 8.2 Breakdown

This is a fairly straightforward configuration, using a MQC policy for frame traffic shaping. The “bandwidth inherit” command will pass configured bandwidth values to a subinterface to match what is configured on the primary interface. If you manually configure a bandwidth value on the subinterface, it will override the inherited value.

Task 8.2 Verification

Watch your ACL creation carefully, we are specifically told to watch for HTTP replies. Verify the policy configuration:

```
Rack1R5#show frame-relay pvc 501

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

DLCI = 501, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0/0.501

  input pkts 2353          output pkts 5770         in bytes 213730
  out bytes 1786756        dropped pkts 7           in pkts dropped 7
  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 5504      out bcast bytes 1727736
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 1000 bits/sec, 0 packets/sec
  pvc create time 03:40:46, last time pvc status changed 03:40:46
  cir 384000    bc 384000    be 0        byte limit 6000  interval 125
  mincir 384000    byte increment 6000  Adaptive Shaping none
  pkts 112       bytes 41576     pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  service policy DLCI 501
 Serial0/0/0.501: DLCI 501 -

  Service-policy output: DLCI_501

    Class-map: HTTP_RESPONSES (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name HTTP_RESPONSES
      Queueing
        Output Queue: Conversation 41
        Bandwidth 80 (%)
        Bandwidth 307 (kbps) Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0

    Class-map: class-default (match-any)
      109 packets, 40580 bytes
      5 minute offered rate 1000 bps, drop rate 0 bps
      Match: any
  Output queue size 0/max total 600/drops 0
```


Task 8.3 Solution

R1:

```
map-class frame-relay DLCI_105
 frame-relay cir 512000
 frame-relay bc 5120
 frame-relay fragment 640
!
interface Serial0/0
 frame-relay traffic-shaping
 frame-relay class DLCI_105
```

R5:

```
Interface Serial0/0/0
 Bandwidth 512
interface Serial0/0/0.502 point-to-point
 frame-relay class DLCI_502
!
map-class frame-relay DLCI_501
 frame-relay cir 512000
 frame-relay bc 5120
 frame-relay fragment 640
!
map-class frame-relay DLCI_502
 frame-relay cir 512000
 frame-relay mincir 128000
```

Task 8.3 Breakdown

Here we have some additional configuration between R5 and R1. In the earlier step, we were just given the CIR for the circuit, but not given the port speed. Here, we have the additional information for the port. By setting bc to 1% of the cir, we are configuring an interval of 10ms. By default, enabling traffic shaping will set circuits to a rate of 56k. In order to have DLCI 502 not be adversely affected, a basic class can be configured for that DLCI.

Task 8.3 Verification

Verify the Frame-Relay PVC shaping parameters:

```
Rack1R5#show frame-relay pvc 501 | begin fragment type
fragment type end-to-end fragment size 640
  cir 512000    bc   5120      be 0         limit 640    interval 10
  mincir 384000    byte increment 640   BECN response no  IF_CONG no
  frags 261       bytes 97278     frags delayed 0       bytes delayed 0
  shaping inactive
  traffic shaping drops 0

Rack1R5#show frame-relay pvc 502 | begin cir
cir 512000    bc 512000    be 0      byte limit 8000   interval 125
  mincir 128000    byte increment 8000  Adaptive Shaping none
  pkts 577       bytes 223590    pkts delayed 2       bytes delayed 166
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/40, 0 drop, 0 dequeued

Rack1R1#show frame-relay pvc 105 | begin fragment type
fragment type end-to-end fragment size 640
  cir 512000    bc   5120      be 0         limit 640    interval 10
  mincir 256000    byte increment 640   BECN response no  IF_CONG no
  frags 56        bytes 5070      frags delayed 0       bytes delayed 0
  shaping inactive
  traffic shaping drops 0
```


Task 8.4 Solution

R3:

```
interface FastEthernet0/0
 ip policy route-map POLICY_ROUTING
!
ip access-list extended FROM_VLAN_367_TO_VLAN_43
 permit ip 139.1.0.0 0.0.0.255 204.12.1.0 0.0.0.255
!
route-map POLICY_ROUTING permit 10
 match ip address FROM_VLAN_367_TO_VLAN_43
 match length 1251 1500
 set ip next-hop 139.1.23.2
```

R5:

```
interface FastEthernet0/1
 ip policy route-map POLICY_ROUTING
!
interface Serial0/1/0
 ip policy route-map POLICY_ROUTING
!
ip access-list extended FROM_VLAN_43_TO_VLAN_367
 permit ip 204.12.1.0 0.0.0.255 139.1.0.0 0.0.0.255
!
route-map POLICY_ROUTING permit 10
 match ip address FROM_VLAN_43_TO_VLAN_367
 match length 1251 1500
 set ip next-hop 139.1.25.2
```


Task 8.4 Verification

Generate packets of different sizes from R6 to BB3 and then enable policy route debugging at R3:

```
Rack1R3#debug ip policy
Policy routing debugging is on

Rack1R6#ping 204.12.1.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 204.12.1.254, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 88/91/92 ms


IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, len 100, FIB policy
match
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, len 100, FIB policy
rejected(deny) - normal forwarding
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, len 100, FIB policy
match
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, len 100, FIB policy
rejected(deny) - normal forwarding
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, len 100, FIB policy
match
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, len 100, FIB policy
rejected(deny) - normal forwarding
```

```
Rack1R6#ping 204.12.1.254 size 1300

Type escape sequence to abort.
Sending 5, 1300-byte ICMP Echos to 204.12.1.254, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max =
1008/1018/1060 ms

IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, len 1300, FIB policy
match
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, g=139.1.23.2, len
1300, FIB policy routed
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, len 1300, FIB policy
match
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, g=139.1.23.2, len
1300, FIB policy routed
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, len 1300, FIB policy
match
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, g=139.1.23.2, len
1300, FIB policy routed
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, len 1300, FIB policy
match
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, g=139.1.23.2, len
1300, FIB policy routed
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, len 1300, FIB policy
match
IP: s=139.1.0.6 (FastEthernet0/0), d=204.12.1.254, g=139.1.23.2, len
1300, FIB policy routed
```

You can also check the output of show route-map on R3 and R5 and verify matches:

```
Rack1R5#show route-map
route-map POLICY_ROUTING, permit, sequence 10
  Match clauses:
    ip address (access-lists): FROM_VLAN_43_TO_VLAN_367
    length 1251 1500
  Set clauses:
    ip next-hop 139.1.25.2
  Policy routing matches: 300 packets, 379500 bytes
```

Task 8.5 Solution

R5:

```
map-class frame-relay DLCI_502
 frame-relay cir 512000
 frame-relay bc 5120
 frame-relay fragment 640
 frame-relay ip rtp priority 16384 16383 512
```

R2:

```
interface Serial0/0
 frame-relay traffic-shaping
 frame-relay class DLCI_205
!
map-class frame-relay DLCI_205
 frame-relay cir 512000
 frame-relay bc 5120
 frame-relay fragment 640
 frame-relay ip rtp priority 16384 16383 512
```

Task 8.5 Verification

Verify the VoIP QoS configuration:

```
Rack1R5#show frame-relay pvc 502 | include Queueing|fragment|rtp
Queueing strategy: weighted fair
  fragment type end-to-end fragment size 640
  ip rtp priority parameters 16384 32767 512000

Rack1R2#show frame-relay pvc 205| include Queueing|fragment|rtp
Queueing strategy: weighted fair
 fragment type end-to-end fragment size 640
 ip rtp priority parameters 16384 32767 512000
```


Task 8.6 Solution

Find SW2's MAC address:

```
Rack1SW2#show arp
Protocol  Address     Age (min)  Hardware Addr   Type   Interface
Internet  139.1.48.8         -   0019.55cb.c341  ARPA   FastEthernet0/20
```

R4:

```
!
class-map SW2
 match destination mac 0019.55cb.c341

policy-map SWOUT
 class SW2
  set precedence 7
!
ace fastEthernet 0/1
 service-policy output SWOUT
```

Task 8.6 Verification

Verify by pinging through from BB3. By matching on the destination MAC, traffic to other hosts on VLAN 24 will not be affected.

```
Rack1R4#show policy-map int fa0/1
 FastEthernet0/1

  Service-policy output: SWOUT

    Class-map: SW2 (match-all)
      106 packets, 12030 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: destination-address mac 0019.55CB.C341
      QoS Set
        precedence 7
          Packets marked 105

    Class-map: class-default (match-any)
      523 packets, 120825 bytes
      5 minute offered rate 1000 bps, drop rate 0 bps
      Match: any
```


Lab 14 Solutions

Task 1.1 Solution

SW1:

```
monitor session 1 source vlan 1011 rx
monitor session 1 destination interface fastEthernet 0/12 ingress vlan
5
```

Task 1.1 Verification

```
Rack1SW1#show monitor session 1
Session 1
---------
Type                   : Local Session
Source VLANs           :
    RX Only            : 1011
Destination Ports      : Fa0/12
    Encapsulation      : Native
          Ingress      : Enabled, default VLAN = 5
    Ingress encap : Untagged
```

Task 1.2 Solution

SW1:

```
!
Rack1SW1#mkdir archive
Create directory filename [archive]?
Created dir flash:archive

Rack1SW1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.

Rack1SW1(config)#alias exec backup copy running-config
flash:/archive/backup.config

Rack1SW1(config)#boot config-file flash:/archive/backup.config
```


Task 1.2 Verification

```
Rack1SW1#dir flash:
Directory of flash:/

    2  -rwx     7963136   Jan 1 1970 02:44:50 +00:00  c3560-
advipservicesk9-mz.122-25.SEE2.bin
    3  -rwx        1197   Mar 1 1993 00:05:09 +00:00  config.old
    4  -rwx         856   Mar 1 1993 00:02:01 +00:00  vlan.dat
    5  -rwx        1914   Mar 1 1993 00:02:05 +00:00  config.text
    7  -rwx         831   Mar 1 1993 23:54:15 +00:00  log.txt
    8  drwx          64   Mar 1 1993 00:45:57 +00:00  archive
   10  -rwx          24   Mar 1 1993 00:45:57 +00:00  private-
config.text

32514048 bytes total (24540672 bytes free)

Rack1SW1#show aliases | include backup
  backup               copy running-config flash:/archive/backup.config

Rack1SW1#show boot
BOOT path-list      : flash:c3560-advipservicesk9-mz.122-25.SEE2.bin
Config file         : flash:/archive/backup.config
Private Config file : flash:/private-config.text
Enable Break        : no
Manual Boot         : no
HELPER path-list    :
Auto upgrade        : yes
```

Task 1.3 Solution

```
R5:
interface FastEthernet0/0
 mac-address 0000.0c12.3456

SW1:
interface FastEthernet0/5
 switchport mode access
 switchport port-security
 switchport port-security mac-address sticky
```


Task 1.3 Verification

```
Rack1SW1(config)#interface f0/5
Rack1SW1(config-if)#shutdown
%LINK-5-CHANGED: Interface FastEthernet0/5, changed state to
      administratively down
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/5,
      changed state to down
Rack1SW1(config-if)#switchport port-security
Rack1SW1(config-if)#switchport port-security mac-address sticky

Rack1R5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#interface e0/0
Rack1R5(config-if)#mac-address 0000.0c12.3456
Rack1R5(config-if)#

Rack1SW1(config-if)#no shutdown
%LINK-3-UPDOWN: Interface FastEthernet0/5, changed state to down
%LINK-3-UPDOWN: Interface FastEthernet0/5, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/5,
      changed state to up

Rack1SW1(config-if)#do show run interface fa0/5
Building configuration...

Current configuration : 231 bytes
!
interface FastEthernet0/5
 switchport access vlan 5
 switchport mode access
 switchport port-security
 switchport port-security mac-address sticky
 switchport port-security mac-address sticky 0000.0c12.3456
 no ip address
end

Rack1SW1#show port-security interface fastEthernet 0/5
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Shutdown
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 1
Configured MAC Addresses   : 0
Sticky MAC Addresses       : 1
Last Source Address:Vlan   : 0000.0c12.3456:5
Security Violation Count   : 0
```


Task 1.4 Solution

SW3 and SW4:

```
no spanning-tree vlan 1363
```

SW4:

```
interface FastEthernet0/20
 switchport backup interface Fa0/21
```

Task 1.4 Verification

```
Rack1SW4#show interface f0/20 switchport backup

Switch Backup Interface Pairs:

Active Interface        Backup Interface       State
------------------------------------------------------------------------
FastEthernet0/20        FastEthernet0/21       Active Up/Backup Standby

Rack1SW4#show spanning-tree vlan 1363

Spanning tree instance(s) for vlan 1363 does not exist.

Rack1SW4#
```

Further Reading

Configuring Flex Links

Task 1.5 Solution

SW3:

```
interface range fa0/14, 15
 no shutdown
 switchport mode access
 switchport access vlan 1363
 no cdep enable
```

SW1:

```
interface fastEthernet 0/1
 switchport access vlan 4
!
interface fastEthernet 0/17
 switchport access vlan 4
 no shut
 no cdp enable
!
interface fastEthernet 0/3
 switchport access vlan 42
!
interface fastEthernet 0/18
 switchport access vlan 42
 no shutdown
 no cdp enable
```

Task1.5 Verification

For testing, you can ping from R1 to R3 and look at the output of **show interface counters** on SW3 while the ping traffic is passing. If the traffic is passing through SW3, you should see activity that corresponds. The section states that VLANs can't be added, but does not prohibit you from using existing VLANs. In this example, CDP is disabled so that you don't get error messages about the native VLANs not matching on both sides.

```
Rack1R1#ping 204.12.1.3 repeat 100000
```

```
Rack1SW3#show int counters | i Port|0/1(4|5)
Port            InOctets   InUcastPkts   InMcastPkts   InBcastPkts
Fa0/14          12527415        105838           608             4
Fa0/15          12514046        105970           176             0
Port           OutOctets  OutUcastPkts  OutMcastPkts  OutBcastPkts
Fa0/14          12573479        105841          1075            79
Fa0/15          12619973        106018          1507            83
Rack1SW3#
```



```
Rack1SW3(config)#interface fastEthernet 0/14
Rack1SW3(config-if)#shutdown

Rack1R1#ping 204.12.1.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 204.12.1.3, timeout is 2 seconds:
 ...
Success rate is 0 percent (0/5)

Rack1SW3(config)#interface fastEthernet 0/14
Rack1SW3(config-if)#no shutdown

Rack1R1#ping 204.12.1.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 204.12.1.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms

Rack1SW3#show int counters | i Port|0/1(4|5)
Port            InOctets   InUcastPkts   InMcastPkts   InBcastPkts
Fa0/14          12527415        105838           608             4
Fa0/15          12514046        105970           176             0
Port           OutOctets  OutUcastPkts  OutMcastPkts  OutBcastPkts
Fa0/14          12573479        105841          1075            79
Fa0/15          12619973        106018          1507            83
```

Task 1.6 Solution

Note

The EIGRP requirement is to have separate passwords for two different neighbors on the same subnet. By using logical interfaces, this can be achieved. If you just configure using the physical interfaces, you may need to reconfigure later.

R1:

```
interface Virtual-Template13
ip address 167.1.135.1 255.255.255.0
!
interface Serial0/0
 encapsulation frame-relay
 frame-relay interface-dlci 103 ppp Virtual-Template13
no ip address 167.1.135.1 255.255.255.0
```

R3:

```
!
interface Virtual-Template13
ip address 167.1.135.3 255.255.255.0
!
interface Virtual-Template35
ip address 167.1.135.3 255.255.255.0
!
interface Serial1/0
 encapsulation frame-relay
 frame-relay interface-dlci 301 ppp Virtual-Template13
 frame-relay interface-dlci 305 ppp Virtual-Template35
 no ip address 167.1.135.3 255.255.255.0
```

R5:

```
!
interface Virtual-Template35
ip address 167.1.135.5 255.255.255.0
!
interface Serial0/0/0
 encapsulation frame-relay
 frame-relay interface-dlci 503 ppp Virtual-Template35
no ip address 167.1.135.5
```


Task 1.6 Verification

```
Rack1R3#show frame-relay pvc 301

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

DLCI = 301, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial1/0

  input pkts 30            output pkts 19           in bytes 6188
  out bytes 334            dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 0         out bcast bytes 0
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 01:44:02, last time pvc status changed 00:48:48
  Bound to Virtual-Access1 (up, cloned from Virtual-Template13)

Rack1R3#show frame-relay pvc 305

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

DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial1/0

  input pkts 33            output pkts 48           in bytes 8124
  out bytes 8370           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 24        out bcast bytes 7968
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 01:44:05, last time pvc status changed 01:28:31
  Bound to Virtual-Access2 (up, cloned from Virtual-Template35)

Rack1R3#ping 167.1.135.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 167.1.135.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/36 ms

Rack1R3#ping 167.1.135.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 167.1.135.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 60/60/64 ms
```


Verify that traffic from R1 to R5 traverses R3 and vice-versa:

```
Rack1R5#ping 167.1.135.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 167.1.135.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 116/116/116
ms

Rack1R5#traceroute 167.1.135.1

Type escape sequence to abort.
Tracing the route to 167.1.135.1

  1 167.1.135.3 32 msec 28 msec 32 msec
  2 167.1.135.1 56 msec *  56 msec

Rack1R5#show frame-relay map interface serial 0/0/0
Serial0/0/0 (up): ip 167.1.135.1 dlci 503(0x1F7,0x7C70), static,
              CISCO, status defined, active
```

Task 1.7 Solution

R4:

```
interface Loopback45
 ip address 167.1.45.4 255.255.255.255
!
Interface Multilink 1
 Ip unnumbered Loopback45

interface Serial0/1/0
 encap ppp
 ppp multilink
 ppp multilink group 1
 encapsulation ppp
```

R5:

```
interface Loopback45
 ip address 167.1.45.5 255.255.255.255
!
Interface Multilink 1
 Ip unnumbered Loopback45
interface Serial0/1/0
 encapsulation ppp
 ppp multilink
 ppp multilink group 1
 clockrate 64000
```


Task 1.7 Verification

Verify the PPP peer-neighbor route:

```
Rack1R4#show ip route connected | section 167.1.45
C       167.1.45.5/32 is directly connected, Multilink1
C       167.1.45.4/32 is directly connected, Loopback45

Rack1R5#show ip route connected | section 167.1.45
C       167.1.45.5/32 is directly connected, Loopback45
C       167.1.45.4/32 is directly connected, Multilink1
```

Verify connectivity:

```
Rack1R4#ping 167.1.45.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 167.1.45.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/29/32 ms
```


Task 2.1 Solution

R4:

```
key chain RIP
 key 1
  key-string CISCO
!
interface FastEthernet0/1
 ip rip authentication mode md5
 ip rip authentication key-chain RIP
 ip rip v2-broadcast
!
router rip
 version 2
 no auto-summary
 network 192.10.1.0
```

Task 2.1 Breakdown

Note

RIPv2 updates are typically sent to the multicast address 224.0.0.9. However, these packets can be sent to the all subnet broadcast address of 255.255.255.255 by issuing the **ip rip v2-broadcast** interface level command.

Task 2.1 Verification

```
Rack1R4#show ip protocols | section Routing Protocol is "rip"
Routing Protocol is "rip"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Sending updates every 30 seconds, next due in 19 seconds
  Invalid after 180 seconds, hold down 180, flushed after 240
  Redistributing: rip
  Default version control: send version 2, receive version 2
    Interface             Send  Recv  Triggered RIP  Key-chain
    FastEthernet0/1       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      00:00:07
  Distance: (default is 120)
```

Verify the RIP updates:

```
Rack1R4#debug ip rip
RIP protocol debugging is on
```



```
RIP: sending v2 update to 255.255.255.255 via FastEthernet0/1
(192.10.1.4)
RIP: build update entries - suppressing null update
RIP: received packet with MD5 authentication
RIP: received v2 update from 192.10.1.254 on FastEthernet0/1
      205.90.31.0/24 via 0.0.0.0 in 7 hops
      220.20.3.0/24 via 0.0.0.0 in 7 hops
      222.22.2.0/24 via 0.0.0.0 in 7 hops

Rack1R4#show key chain
Key-chain RIP:
    key 1 -- text "CISCO"
        accept lifetime (always valid) - (always valid) [valid now]
        send lifetime (always valid) - (always valid) [valid now]
```

Task 2.2 Solution

R2, R5, SW1, and SW2:

```
router ospf 1
 auto-cost reference-bandwidth 1000
```

Task 2.2 Verification

```
Rack1SW2#show ip ospf interface port-channel 1
Port-channel1 is up, line protocol is up (connected)
  Internet Address 167.1.78.8/24, Area 2578
  Process ID 1, Router ID 150.1.8.8, Network Type BROADCAST, Cost: 10
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 150.1.8.8, Interface address 167.1.78.8
  Backup Designated router (ID) 150.1.7.7, Interface address 167.1.78.7
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:04
  Supports Link-local Signaling (LLS)
  Index 2/2, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.7.7  (Backup Designated Router)
  Suppress hello for 0 neighbor(s)
```


Task 2.3 Solution

SW3:

```
!
! IGMP Snooping needs to be disabled in order to configure the static
! multicast MAC address
!
no ip igmp snooping vlan 1363
!
interface FastEthernet0/24
 ip access-group DENY_EIGRP in
!
ip access-list extended DENY_EIGRP
 deny   eigrp any any
 permit ip any any
!
mac-address-table static 0100.5e00.000a vlan 1363 int fa0/14 fa0/15
fa0/20 fa0/21
```

Task 2.3 Breakdown

Configuring the static MAC entry will limit the destinations for the multicast MAC to the ports where the other EIGRP neighbors are reachable, and will prevent the hello traffic being sent out Fa0/24 to BB3. Configuring the access list will block the traffic inbound from BB3. The OUI for multicast mapping is 01:00:5E, and the last 23bits of the address correspond to the last 23 bits of the Ethernet address. EIGRP's hello traffic uses the multicast address 224.0.0.10, the last three octets are 0.0.10, which converts to 00:00:0a for the last 23 bits in hex. Your configuration for this section may vary, depending on how you configured previously.

Task 2.3 Verification

Check EIGRP neighbors:

```
Rack1R6#show ip eigrp neighbors
IP-EIGRP neighbors for process 10
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
1   204.12.1.3              Fa0/0             10 05:10:29    5   200  0  118
0   204.12.1.1              Fa0/0             10 05:10:29  819  4914  0  90
```

Check EIGRP routes:

```
Rack1R3#show ip route eigrp
     167.1.0.0/16 is variably subnetted, 6 subnets, 2 masks
D       167.1.45.4/32 [90/20640000] via 167.1.34.4, 05:13:19,
Serial1/1.34
     150.1.0.0/24 is subnetted, 4 subnets
D       150.1.6.0 [90/156160] via 204.12.1.6, 05:11:16, FastEthernet0/0
D       150.1.4.0 [90/20640000] via 167.1.34.4, 05:13:19, Serial1/1.34
D       150.1.1.0 [90/156160] via 204.12.1.1, 05:13:19, FastEthernet0/0
```

The following verification could not be reproduced in the rental racks or in the real lab, as it configures the backbone router. However, it is presented here to illustrate how the task could potentially be verified.

```
BB3:
!
router eigrp 10
 network 204.12.1.0
!
access-list 100 permit eigrp 204.12.1.0 0.0.0.255 any
```

```
BB3#debug ip packet detail 100

IP: s=204.12.1.6 (Ethernet0), d=224.0.0.10, len 60, rcvd 2, proto=88
IP: s=204.12.1.3 (Ethernet0), d=224.0.0.10, len 60, rcvd 2, proto=88
IP: s=204.12.1.1 (Ethernet0), d=224.0.0.10, len 60, rcvd 2, proto=88
IP: s=204.12.1.254 (local), d=224.0.0.10 (Ethernet0), len 60, sending
broad/multicast, proto=88
IP: s=204.12.1.6 (Ethernet0), d=224.0.0.10, len 60, rcvd 2, proto=88
IP: s=204.12.1.3 (Ethernet0), d=224.0.0.10, len 60, rcvd 2, proto=88
```

```
BB3#show ip eigrp neighbors
IP-EIGRP neighbors for process 10
H   Address       Interface       Hold Uptime   SRTT   RTO  Q  Seq Type
                                  (sec)         (ms)       Cnt Num
2   204.12.1.1     Et0               3 00:03:12 1439  5000  0  15
1   204.12.1.3     Et0               1 00:03:12   24   200  0  28
0   204.12.1.6     Et0               1 00:03:12   19   200  0  10
```


Enable filtering and check debugging output again:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 10: Neighbor 204.12.1.6 (Ethernet0) is
down: holding time expired
destroy peer: 204.12.1.6
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 10: Neighbor 204.12.1.3 (Ethernet0) is
down: holding time expired
destroy peer: 204.12.1.3
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 10: Neighbor 204.12.1.1 (Ethernet0) is
down: holding time expired
destroy peer: 204.12.1.1

BB3#debug ip packet detail 100
IP packet debugging is on (detailed) for access list 100
IP: s=204.12.1.254 (local), d=224.0.0.10 (Ethernet0), len 60, sending
broad/multicast, proto=88
IP: s=204.12.1.254 (local), d=224.0.0.10 (Ethernet0), len 60, sending
broad/multicast, proto=88
IP: s=204.12.1.254 (local), d=224.0.0.10 (Ethernet0), len 60, sending
broad/multicast, proto=88
IP: s=204.12.1.254 (local), d=224.0.0.10 (Ethernet0), len 60, sending
broad/multicast, proto=88
```


Task 2.4 Solution

R4:

```
interface Serial0/1/0
 bandwidth 1536
 ip bandwidth-percent eigrp 10 25
```

R1:

```
key chain EIGRP
 key 13
  key-string CISCO13
!
interface Virtual-Template13
 ip authentication mode eigrp 10 md5
 ip authentication key-chain eigrp 10 EIGRP
!
router eigrp 10
 network 167.1.135.1 0.0.0.0
```

R3:

```
key chain EIGRP13
  key 13
   key-string CISCO13
!
key chain EIGRP35
  key 35
    key-string CISCO35
!
interface Virtual-Template13
  ip authentication mode eigrp 10 md5
  ip authentication key-chain eigrp 10 EIGRP13
!
interface Virtual-Template35
  ip authentication mode eigrp 10 md5
  ip authentication key-chain eigrp 10 EIGRP35
!
router eigrp 10
  network 167.1.135.3 0.0.0.0
```

R5:

```
key chain EIGRP
 key 35
  key-string CISCO35
!
interface Virtual-Template35
 ip authentication mode eigrp 10 md5
 ip authentication key-chain eigrp 10 EIGRP
!
interface Serial0/1/0
 bandwidth 1536
 ip bandwidth-percent eigrp 10 25

router eigrp 10
 network 167.1.135.5 0.0.0.0
```


Task 2.4 Verification

Verify EIGRP authentication:

```
Rack1R3#show ip eigrp interfaces detail virtual-access 2
IP-EIGRP interfaces for process 10

                   Xmit Queue   Mean   Pacing Time   Multicast    Pending
Interface  Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer   Routes
Vi2            1        0/0       121        0/1           637           0
  Hello interval is 5 sec
  Next xmit serial <none>
  Un/reliable mcasts: 0/0  Un/reliable ucasts: 4/8
  Mcast exceptions: 0  CR packets: 0  ACKs suppressed: 0
  Retransmissions sent: 0  Out-of-sequence rcvd: 0
  Authentication mode is md5,  key-chain is "EIGRP13"
  Use unicast

Rack1R3#show ip eigrp interfaces detail virtual-access 3
IP-EIGRP interfaces for process 10

                   Xmit Queue   Mean   Pacing Time   Multicast    Pending
Interface  Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer   Routes
Vi3            1        0/0       207        0/1           977           0
  Hello interval is 5 sec
  Next xmit serial <none>
  Un/reliable mcasts: 0/0  Un/reliable ucasts: 5/6
  Mcast exceptions: 0  CR packets: 0  ACKs suppressed: 0
  Retransmissions sent: 1  Out-of-sequence rcvd: 1
  Authentication mode is md5,  key-chain is "EIGRP35"
  Use unicast
```

Verify the EIGRP neighbors:

```
Rack1R3#show ip eigrp neighbors
IP-EIGRP neighbors for process 10
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
5   167.1.135.5             Vi3               11 00:03:51  207  1242  0  22
4   167.1.135.1             Vi2               13 00:04:05  121   726  0  107
3   204.12.1.6              Fa0/0             12 05:22:12    4   200  0  56
1   204.12.1.1              Fa0/0             12 05:24:15    4   200  0  105
0   167.1.13.1              Se1/2             14 12:54:50   60  1140  0  106
2   167.1.34.4              Se1/1.34         156 13:18:01   81  1140  0  48
```


Task 2.5 Solution

R1:

```
router eigrp 10
 eigrp stub connected
 timers active-time 1
```

Task 2.5 Verification

```
Rack1R3#show ip eigrp neighbors detail serial 1/2
IP-EIGRP neighbors for process 10
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q
Seq
                                            (sec)         (ms)
Cnt Num
4   167.1.13.1              Se1/2             11 00:00:49   45  1140  0
120
   Version 12.4/1.2, Retrans: 0, Retries: 0, Prefixes: 4
   Stub Peer Advertising ( CONNECTED ) Routes
   Suppressing queries
```


Task 2.6 Solution

R4:

```
interface FastEthernet0/1
 ip summary-address rip 167.1.0.0 255.255.0.0
 ip summary-address rip 150.1.0.0 255.255.240.0
!
router eigrp 10
 redistribute rip metric 10000 10 255 1 1500
!
router rip
 redistribute eigrp 10 metric 1
```

R5:

```
interface Virtual-Template35
 ip summary-address eigrp 10 0.0.0.0 0.0.0.0 5
!
interface Multilink1
 ip summary-address eigrp 10 0.0.0.0 0.0.0.0 5
!
router ospf 1
 default-information originate always
```


Task 2.6 Verification

Check for the default route:

```
Rack1R4#show ip route eigrp
D    204.12.1.0/24 [90/2172416] via 167.1.34.3, 00:45:43, Serial0/0/0
     167.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
D       167.1.135.1/32 [90/4729856] via 167.1.34.3, 00:45:43,
Serial0/0/0
D       167.1.135.0/24 [90/4729856] via 167.1.34.3, 00:45:43,
Serial0/0/0
D       167.1.135.5/32 [90/4729856] via 167.1.34.3, 05:55:37,
Serial0/0/0
D       167.1.13.0/24 [90/21024000] via 167.1.34.3, 00:45:43,
Serial0/0/0
     150.1.0.0/24 is subnetted, 4 subnets
D       150.1.6.0 [90/2300416] via 167.1.34.3, 00:45:43, Serial0/0/0
D       150.1.3.0 [90/2297856] via 167.1.34.3, 00:45:43, Serial0/0/0
D       150.1.1.0 [90/2300416] via 167.1.34.3, 00:45:43, Serial0/0/0
D*   0.0.0.0/0 [90/4354560] via 167.1.45.5, 00:43:17, Multilink1

Rack1SW1#show ip route ospf
     167.1.0.0/24 is subnetted, 3 subnets
O       167.1.58.0 [110/20] via 167.1.78.8, 00:46:17, Port-channel1
     150.1.0.0/24 is subnetted, 4 subnets
O       150.1.5.0 [110/21] via 167.1.78.8, 00:46:17, Port-channel1
O       150.1.2.0 [110/11] via 167.1.27.2, 00:46:17, FastEthernet0/14
O       150.1.8.0 [110/11] via 167.1.78.8, 00:46:17, Port-channel1
O*E2 0.0.0.0/0 [110/1] via 167.1.78.8, 00:46:17, Port-channel1
```

Verify that RIP routes are re-distributed into EIGRP:

```
Rack1R4#show ip route rip
R    222.22.2.0/24 [120/7] via 192.10.1.254, 00:00:07, FastEthernet0/1
R    220.20.3.0/24 [120/7] via 192.10.1.254, 00:00:07, FastEthernet0/1
R    205.90.31.0/24 [120/7] via 192.10.1.254, 00:00:07, FastEthernet0/1

Rack1R5#show ip route eigrp | i EX
D EX 192.10.1.0/24 [170/4229120] via 167.1.45.4, 00:13:51, Multilink1
D EX 222.22.2.0/24 [170/4229120] via 167.1.45.4, 00:13:51, Multilink1
D EX 220.20.3.0/24 [170/4229120] via 167.1.45.4, 00:13:51, Multilink1
D EX 205.90.31.0/24 [170/4229120] via 167.1.45.4, 00:13:51, Multilink1
```


Finally, test full connectivity with the following Tcl script:

```
foreach i {
167.1.135.1
150.1.1.1
167.1.13.1
204.12.1.1
150.1.2.2
167.1.27.2
167.1.135.3
167.1.34.3
150.1.3.3
167.1.13.3
204.12.1.3
167.1.34.4
167.1.45.4
150.1.4.4
192.10.1.4
167.1.135.5
167.1.45.5
150.1.5.5
167.1.58.5
150.1.6.6
204.12.1.6
150.1.7.7
167.1.27.7
167.1.78.7
167.1.58.8
150.1.8.8
167.1.78.8
222.22.2.1
} {ping $i}
```

Note that VLAN4, VLAN5, and Serial link from R6 to BB1 are excluded from connectivity test. Also, SW3 will not have reachability until later in the lab.

Task 2.7 Solution

R1:

```
router bgp 100
 neighbor 150.1.3.3 remote-as 100
 neighbor 150.1.3.3 update-source Loopback0
```

R3:

```
router bgp 100
 neighbor iBGP peer-group
 neighbor iBGP remote-as 100
 neighbor iBGP update-source Loopback0
 neighbor iBGP route-reflector-client
 neighbor iBGP send-community
 neighbor 150.1.1.1 peer-group iBGP
 neighbor 150.1.4.4 peer-group iBGP
 neighbor 167.1.135.5 peer-group iBGP
 neighbor 150.1.6.6 peer-group iBGP
 neighbor 150.1.9.9 peer-group iBGP
 neighbor 150.1.9.9 shutdown
 neighbor 150.1.10.10 peer-group iBGP
 neighbor 150.1.10.10 shutdown
```

R4:

```
router bgp 100
 neighbor 150.1.3.3 remote-as 100
 neighbor 150.1.3.3 update-source Loopback0
```

R5:

```
router bgp 100
 neighbor 150.1.3.3 remote-as 100
```

R6:

```
router bgp 100
 neighbor 150.1.3.3 remote-as 100
 neighbor 150.1.3.3 update-source Loopback0
 neighbor 150.1.3.3 next-hop-self
```


Task 2.7 Breakdown

BGP peer groups are a way to minimize redundant configuration between neighbors that share common attributes. For example, R3 is peering with R1, R4, R5, R6, and two additional devices. These devices are all in AS 100 and are route-reflector clients of R3. Instead of specifying two neighbor statements applying the **remote-as** and **route-reflector-client** options, a peer group has been defined that has these options applied. Then, instead of applying the options directly on the neighbor, the neighbor is simply specified as part of the predefined peer-group.

The **shutdown** option of the BGP neighbor command is typically used for the case that is described in this task. For example, a new circuit may be on order that involves a BGP peering session. Instead of waiting until the circuit is installed and up, the BGP configuration can be applied beforehand, and the neighbor disabled with the **neighbor [*address*] shutdown** option. Therefore, the only configuration that is required once the new circuit is up is to issue a **no** statement for the command with the shutdown applied.

Note that R6 has the `next-hop-self` command configured on its peering with R3. If it wasn't R3 would haven seen AS54 learned prefixes from R6 with a next-hop value of 54.X.1.254. As R5 is injecting the default route inside the EIGRP domain, R3 will be able to resolve the next hop, but packets will be actually forwarded towards R5.

Task 2.7 Verification

Verify the BGP neighbors:

```
Rack1R3#show ip bgp summary | begin Neighbor
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
150.1.1.1       4   100       7      11       19    0    0 00:02:30        2
150.1.4.4       4   100       4      11       19    0    0 00:02:42        0
150.1.6.6       4   100      12      11       19    0    0 00:02:19       10
150.1.9.9       4   100       0       0        0    0    0 never    Idle(Admin)
150.1.10.10     4   100       0       0        0    0    0 never    Idle(Admin)
167.1.135.5     4   100       4      11       19    0    0 00:02:26        0
204.12.1.254    4    54     696     699       19    0    0 11:31:59        2

Rack1R3#show  ip bgp peer-group
BGP peer-group is iBGP,  remote AS 100
  BGP version 4
  Default minimum time between advertisement runs is 0 seconds

 For address family: IPv4 Unicast
  BGP neighbor is iBGP, peer-group internal, members:
  150.1.1.1 150.1.4.4 150.1.6.6 150.1.9.9 150.1.10.10 167.1.135.5
  Index 0, Offset 0, Mask 0x0
  Route-Reflector Client
  Community attribute sent to this neighbor
  Update messages formatted 0, replicated 0
  Number of NLRIs in the update sent: max 0, min 0
```

Task 2.8 Solution

R4:

```
router bgp 100
 neighbor 192.10.1.254 remote-as 254
 neighbor 192.10.1.254 local-as 200
 neighbor 192.10.1.254 password CISCO
```

Task 2.8 Verification

```
Rack1R4#show ip bgp summary | begin Neighbor
Neighbor        V AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
150.1.3.3       4 100      15       9       15    0    0 00:04:23       11
192.10.1.254    4 254       5       8       12    0    0 00:00:27        3
```

Check local-AS configuration:

```
Rack1R4#show ip bgp  neighbors 192.10.1.254
BGP neighbor is 192.10.1.254,  remote AS 254,  local AS 200, external
link
  BGP version 4, remote router ID 222.22.2.1
  BGP state = Established, up for 00:01:03
  Last read 00:00:02, last write 00:00:02, hold time is 180, keepalive
interval is 60 seconds
<output omitted>
```


Check for any prepended AS:

```
Rack1R4#show ip bgp quote-regexp _254$
BGP table version is 15, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      192.10.1.254             0             0 200 254 ?
*> 220.20.3.0       192.10.1.254             0             0 200 254 ?
*> 222.22.2.0       192.10.1.254             0             0 200 254 ?
```

Task 2.9 Solution

R4:

```
router bgp 100
 neighbor 192.10.1.254 local-as 200 no-prepend
```

Task 2.9 Verification

Confirm that AS 200 is not prepended:

```
Rack1R4#show ip bgp quote-regexp _254$
BGP table version is 21, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      192.10.1.254             0             0 254 ?
*> 220.20.3.0       192.10.1.254             0             0 254 ?
*> 222.22.2.0       192.10.1.254             0             0 254 ?
```


Task 2.10 Solution

R1:

```
router bgp 100
 neighbor 204.12.1.254 route-map TO_BB3 out
!
ip prefix-list VLAN4_AND_VLAN5 seq 5 permit 167.1.4.0/23  ge 24 le 24
!
route-map TO_BB3 permit 10
 match ip address prefix-list VLAN4_AND_VLAN5
 set as-path prepend 100 100
!
route-map TO_BB3 permit 1000
```

R3:

```
router bgp 100
 neighbor 204.12.1.254 route-map TO_BB3 out
!
ip prefix-list VLAN4_AND_VLAN5 seq 5 permit 167.1.4.0/23 ge 24 le 24
!
route-map TO_BB3 permit 10
 match ip address prefix-list VLAN4_AND_VLAN5
set as-path prepend 100 100
!
route-map TO_BB3 permit 1000
```

R4:

```
router bgp 100
 network 167.1.4.0 mask 255.255.255.0
```

R5:

```
router bgp 100
 network 167.1.5.0 mask 255.255.255.0
```

R6:

```
router bgp 100
 neighbor 204.12.1.254 route-map TO_BB3 out
!
ip prefix-list VLAN4_AND_VLAN5 seq 5 permit 167.1.4.0/23 ge 24 le 24
!
route-map TO_BB3 permit 10
 match ip address prefix-list VLAN4_AND_VLAN5
 set as-path prepend 100 100
!
route-map TO_BB3 permit 1000
```


Task 2.10 Verification

Verify the BGP tables of BB1 and BB3:

```
BB1 >show ip bgp quote-regexp _100$
BGP table version is 987, local router ID is 212.18.3.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 167.1.4.0/24     54.1.1.6                               0 100 i
*> 167.1.5.0/24     54.1.1.6                               0 100 i
```

```
BB3>show ip bgp quote-regexp _100$
BGP table version is 35, local router ID is 31.3.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop        Metric LocPrf Weight Path
*>i167.1.4.0/24     172.16.4.1           0    100      0 100 i
*                   204.12.1.3                        0 100 100 100 i
*                   204.12.1.3                        0 100 100 100 i
*                   204.12.1.3                        0 100 100 100 i
*>i167.1.5.0/24     172.16.4.1           0    100      0 100 i
*                   204.12.1.3                        0 100 100 100 i
*                   204.12.1.3                        0 100 100 100 i
*                   204.12.1.3                        0 100 100 100 i
```

Task 2.11 Solution

SW1:

```
router bgp 65078
 network 150.1.7.0 mask 255.255.255.0
```

SW2:

```
router bgp 65078
 network 150.1.8.0 mask 255.255.255.0
 aggregate-address 150.1.0.0 255.255.240.0 summary-only
 distance bgp 20 200 255
```


Task 2.11 Breakdown

Once SW2 is configured for aggregation, it will install a local route to Null0 for 150.1.0.0/20. As SW2 does not have more specific routes for R1, R3, R4 and R6 Loopbacks, SW2 will drop traffic towards it. In this case R2, SW1 and SW2 will loose connectivity with these prefixes. This is why SW2 is configured with an administrative distance of 255 under BGP for local generated routes

Task 2.11 Verification

Check for the summary received from SW2:

```
Rack1R5#show ip bgp neighbors 167.1.58.8 routes
BGP table version is 31, 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
*> 150.1.0.0/20     167.1.58.8               0             0 65078 i

Total number of prefixes 1
```

Verify that SW2 has connectivity with the above Loopbacks:

```
Rack1SW2#show ip bgp rib-failure
Network          Next Hop                  RIB-failure   RIB-NH Matches
150.1.0.0/20     0.0.0.0          Admin distance >= 255             n/a
150.1.7.0/24     167.1.78.7       Higher admin distance             n/a

Rack1SW2#ping 150.1.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 58/62/67 ms

Rack1SW2#ping 150.1.3.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.3.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 58/60/67 ms
```


Task 2.12 Solution

R1, R3, and R6:

```
router bgp 100
 neighbor 204.12.1.254 remove-private-as
```

R4:

```
router bgp 100
 neighbor 192.10.1.254 remove-private-as
```

R6:

```
router bgp 100
neighbor 54.1.1.254 remove-private-as
```

Task 2.12 Verification

Check AS-path for aggregated prefix on BB1:

```
BB1>show ip bgp 150.1.0.0
BGP routing table entry for 150.1.0.0/20, version 990
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Flag: 0x840
  Advertised to non peer-group peers:
  172.16.4.3
  100, (aggregated by 65078 150.1.8.8)
    172.16.4.3 from 172.16.4.3 (31.3.0.1)
      Origin IGP, metric 0, localpref 100, valid, internal, atomic-
aggregate
  100, (aggregated by 65078 150.1.8.8)
    54.1.1.6 from 54.1.1.6 (150.1.6.6)
      Origin IGP, localpref 100, valid, external, atomic-aggregate,
best
```

```
BB2>show ip bgp 150.1.0.0
BGP routing table entry for 150.1.0.0/20, version 40
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Not advertised to any peer
  200 100, (aggregated by 65078 150.1.8.8)
    192.10.1.4 from 192.10.1.4 (150.1.4.4)
      Origin IGP, localpref 100, valid, external, atomic-aggregate,
best
```

Caution

As mentioned in previous labs, you will not have access to the BB routers to execute commands on during the real lab.

Task 2.13 Solution

R3:

```
router bgp 100
 bgp inject-map ORIGINATE exist-map LEARNED_PATH
neighbor iBGP route-map TO_IBGP_PEERS out
!
ip prefix-list ORIGINATED_ROUTES seq 10 permit 150.1.8.0/24
ip prefix-list ROUTE seq 5 permit 150.1.0.0/20
ip prefix-list ROUTE_SOURCE seq 5 permit 167.1.135.5/32
ip prefix-list SPECIFIC_ROUTES seq 10 permit 150.1.8.0/24
!
route-map LEARNED_PATH permit 10
 match ip address prefix-list ROUTE
 match ip route-source prefix-list ROUTE_SOURCE
!
route-map ORIGINATE permit 10
 set ip address prefix-list ORIGINATED_ROUTES
!
route-map TO_IBGP_PEERS deny 10
 match ip address prefix-list SPECIFIC_ROUTES
!
route-map TO_IBGP_PEERS permit 1000
```

R6:

```
router bgp 100
 bgp inject-map ORIGINATE exist-map LEARNED_PATH
 neighbor 150.1.3.3 route-map TO_R3 out
!
ip prefix-list ORIGINATED_ROUTES seq 10 permit 150.1.7.0/24
ip prefix-list ROUTE seq 5 permit 150.1.0.0/20
ip prefix-list ROUTE_SOURCE seq 5 permit 150.1.3.3/32
ip prefix-list SPECIFIC_ROUTES seq 5 permit 150.1.7.0/24
!
route-map LEARNED_PATH permit 10
 match ip address prefix-list ROUTE
 match ip route-source prefix-list ROUTE_SOURCE
!
route-map TO_R3 deny 10
 match ip address prefix-list SPECIFIC_ROUTES
!
route-map TO_R3 permit 1000
!
route-map ORIGINATE permit 10
 set ip address prefix-list ORIGINATED_ROUTES
!
route-map TO_BB3 deny 5
 match ip address prefix-list SPECIFIC_ROUTES
```


Task 2.13 Breakdown

The BGP conditional route injection feature allows a router to originate an arbitrary network block based on the existence of a prefix in the BGP table. This feature is designed to be used in the case that is described in this task.

In the above task, AS 100 is learning the aggregate block 150.1.0.0/20 from AS 65078. Since AS 100 has multiple exit points to AS 54, it may be desirable for AS 100 to create a traffic engineering policy based on longer matches. By re-injecting subnets that make up the aggregate, AS 100 can force it's upstream peers (AS 54 in this case) to follow a forwarding policy based on the longer match to the destination.

The BGP conditional route injection feature relies on two parts, the inject-map and the exist-map. When the prefix and route-source matched in the exist-map exist in the BGP table, the prefix or prefixes set in the inject-map are injected into the BGP table.

Task 2.13 Verification

Verify the BGP prefix injection:

```
Rack1R6#show ip bgp injected-paths
BGP table version is 18, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i150.1.7.0/24     167.1.58.8                             0 ?
```

```
Rack1R3#show ip bgp injected-paths
BGP table version is 32, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i150.1.8.0/24     167.1.58.8                             0 ?
```

Verify the specific prefix advertisements:

```
Rack1R3#show ip bgp neighbors 204.12.1.254 advertised-routes
BGP table version is 32, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i150.1.0.0/20     167.1.58.8               0    100      0 65078 i
*>i150.1.8.0/24     167.1.58.8                            0 ?
*>i167.1.4.0/24     150.1.4.4                0    100      0 i
*>i167.1.5.0/24     167.1.135.5              0    100      0 i
*>i205.90.31.0      192.10.1.254             0    100      0 254 ?
*>i220.20.3.0       192.10.1.254             0    100      0 254 ?
*>i222.22.2.0       192.10.1.254             0    100      0 254 ?
<output omitted>
```



```
Rack1R3#show ip bgp neighbors 150.1.1.1 advertised-routes
BGP table version is 61, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   204.12.1.254             0             0 54 i
*> 28.119.17.0/24   204.12.1.254             0             0 54 i
*>i112.0.0.0        150.1.6.6                0    100      0 54 50 60 i
*>i113.0.0.0        150.1.6.6                0    100      0 54 50 60 i
*>i114.0.0.0        150.1.6.6                0    100      0 54 i
*>i115.0.0.0        150.1.6.6                0    100      0 54 i
*>i116.0.0.0        150.1.6.6                0    100      0 54 i
*>i117.0.0.0        150.1.6.6                0    100      0 54 i
*>i118.0.0.0        150.1.6.6                0    100      0 54 i
*>i119.0.0.0        150.1.6.6                0    100      0 54 i
*>i150.1.0.0/20     167.1.58.8               0    100      0 65078 i
<output omitted>

Rack1R6#show ip bgp neighbors 204.12.1.254 advertised-routes
BGP table version is 18, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i150.1.0.0/20     167.1.58.8               0    100      0 65078 i
*>i167.1.4.0/24     150.1.4.4                0    100      0 i
*>i167.1.5.0/24     167.1.135.5              0    100      0 i
*>i205.90.31.0      192.10.1.254             0    100      0 254 ?
*>i220.20.3.0       192.10.1.254             0    100      0 254 ?
*>i222.22.2.0       192.10.1.254             0    100      0 254 ?
<output omitted>
```



```
Rack1R6#show ip bgp neigh 54.1.1.254 advertised-routes
BGP table version is 18, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   204.12.1.254             0             0 54 i
*> 28.119.17.0/24   204.12.1.254             0             0 54 i
*> 112.0.0.0        204.12.1.254                           0 54 50 60 i
*> 113.0.0.0        204.12.1.254                           0 54 50 60 i
*> 114.0.0.0        204.12.1.254                           0 54 i
*> 115.0.0.0        204.12.1.254                           0 54 i
*> 116.0.0.0        204.12.1.254                           0 54 i
*> 117.0.0.0        204.12.1.254                           0 54 i
*> 118.0.0.0        204.12.1.254                           0 54 i
*> 119.0.0.0        204.12.1.254                           0 54 i
*>i150.1.0.0/20     167.1.58.8               0    100      0 65078 i
*>i150.1.7.0/24     167.1.58.8                             0 ?
<snip>
```


Task 3.1 Solution

R6:

```
ipv6 unicast-routing
!
interface Loopback100
 ipv6 address 2001:150:1:26::6/64
 ipv6 eigrp 101
!
interface Loopback101
 ipv6 address 2001:150:1:2E::6/64
 ipv6 eigrp 101

ipv6 router eigrp 101
 no shut
```

Task 3.1 Verification

```
Rack1R6#show ipv6 interface brief | beg Loop
Loopback100                [up/up]
    FE80::213:C4FF:FEA6:9420
    2001:150:1:26::6
Loopback101                [up/up]
    FE80::213:C4FF:FEA6:9420
    2001:150:1:2E::6

Rack1R6#show ipv6 eigrp 101 interface detail
IPv6-EIGRP interfaces for process 101

                        Xmit Queue   Mean   Pacing Time   Multicast
Pending
Interface        Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer
Routes
Lo100              0        0/0         0       0/1            0
0
  Hello interval is 5 sec
  Next xmit serial <none>
  Un/reliable mcasts: 0/0  Un/reliable ucasts: 0/0
  Mcast exceptions: 0  CR packets: 0  ACKs suppressed: 0
  Retransmissions sent: 0  Out-of-sequence rcvd: 0
  Authentication mode is not set
  Use multicast
Lo101              0        0/0         0       0/1            0
0
  Hello interval is 5 sec
  Next xmit serial <none>
  Un/reliable mcasts: 0/0  Un/reliable ucasts: 0/0
  Mcast exceptions: 0  CR packets: 0  ACKs suppressed: 0
  Retransmissions sent: 0  Out-of-sequence rcvd: 0
  Authentication mode is not set
  Use multicast
Rack1R6#
```


Task 3.2 Solution

R4:

```
ipv6 unicast-routing
!
interface Tunnel46
 ipv6 address 2001:167:1:46::4/64
 tunnel source Loopback0
 tunnel destination 150.1.6.6
!
! IPv6 in IP has smallest overhead
!
 tunnel mode ipv6ip
```

R6:

```
interface Tunnel46
 ipv6 address 2001:167:1:46::6/64
 tunnel source Loopback0
 tunnel destination 150.1.4.4
 tunnel mode ipv6ip
```

Task 3.2 Verification

```
Rack1R6#show interfaces tunnel 46
Tunnel46 is up, line protocol is up
  Hardware is Tunnel
  MTU 1514 bytes, BW 100 Kbit, DLY 500000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 150.1.6.6 (Loopback0), destination 150.1.4.4
  Tunnel protocol/transport IPv6/IP
<output omitted>

Rack1R6#ping 2001:167:1:46::4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:167:1:46::4, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/68/72 ms
```

Task 3.3 Solution

R4:

```
interface FastEthernet0/0
 ipv6 address 2001:167:1:4::4/64
 ipv6 eigrp 101
!
interface Tunnel46
 ipv6 eigrp 101

ipv6 router eigrp 101
 no shut
```

R6:

```
interface Tunnel46
 ipv6 eigrp 101
ipv6 summary-address eigrp 101 0::0/0 7
ipv6 summary-address eigrp 101 2001:150:1:20::0/60 7
```

Task 3.3 Verification

```
Rack1R4#show ipv6 eigrp neighbors
IPv6-EIGRP neighbors for process 101
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   Link-local address:     Tu46              11 00:00:17   96   5000  0  5
    FE80::9601:606

Rack1R4#show ipv6 route eigrp
IPv6 Routing Table - Default - 7 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
D   ::/0 [90/27008000]
     via FE80::9601:606, Tunnel46
D   2001:150:1:20::/60 [90/27008000]
     via FE80::9601:606, Tunnel46

Rack1R4#ping 2001:150:1:26::6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:150:1:26::6, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/68/68 ms

Rack1R4#ping 2001:150:1:2E::6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:150:1:2E::6, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/70/72 ms
```


Task 5.1 Solution

R3, R4 and R5:

```
ip pim rp-address 150.1.4.4 override
```

Task 5.1 Verification

Verify the PIM RP to group mapping:

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

Group(s): 224.0.0.0/4, Static-Override
    RP: 150.1.4.4 (?)

Rack1R3#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s): 224.0.0.0/4, Static-Override
    RP: 150.1.4.4 (?)

Rack1R5#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s): 224.0.0.0/4, Static-Override
    RP: 150.1.4.4 (?)
```


Task 5.2 Solution

R3:

```
interface Serial1/1.34
 ip dvmrp unicast-routing
!
interface Virtual-Template35
 ip pim sparse-mode
 ip dvmrp unicast-routing
```

R4:

```
interface Tunnel0
 ip unnumbered FastEthernet0/0
 ip pim sparse-mode
 tunnel source FastEthernet0/1
 tunnel destination 220.20.3.192
 tunnel mode dvmrp
!
interface Multilink1
 ip pim sparse-mode
 ip dvmrp unicast-routing
!
interface Serial0/0/0
 ip dvmrp unicast-routing
```

R5:

```
interface Virtual-Template35
 ip pim sparse-mode
 ip dvmrp unicast-routing
!
interface Multilink1
 ip pim sparse-mode
 ip dvmrp unicast-routing
```


Task 5.2 Verification

Verify the DVMRP routes:

```
Rack1R4#show ip dvmrp route
DVMRP Routing Table - 7 entries
150.1.0.0/16 [0/2] uptime 00:09:21, expires 00:02:09
    via 167.1.34.3, Serial0/0
167.1.5.0/24 [0/2] uptime 00:09:21, expires 00:02:39
    via 167.1.45.5, Multilink1
167.1.45.4/32 [0/3] uptime 00:09:21, expires 00:02:09
    via 167.1.34.3, Serial0/0
167.1.135.0/24 [0/2] uptime 00:09:21, expires 00:02:09
    via 167.1.34.3, Serial0/0
167.1.135.3/32 [0/2] uptime 00:09:21, expires 00:02:39
    via 167.1.45.5, Multilink1
167.1.135.5/32 [0/2] uptime 00:08:51, expires 00:02:09
    via 167.1.34.3, Serial0/0
204.12.1.0/24 [0/2] uptime 00:08:51, expires 00:02:09
    via 167.1.34.3, Serial0/0

Rack1R5#show ip dvmrp route
DVMRP Routing Table - 8 entries
150.1.0.0/16 [0/2] uptime 00:09:28, expires 00:02:44
    via 167.1.45.4, Multilink1
167.1.4.0/24 [0/2] uptime 00:09:28, expires 00:02:44
    via 167.1.45.4, Multilink1
167.1.34.0/24 [0/2] uptime 00:09:28, expires 00:02:44
    via 167.1.45.4, Multilink1
167.1.45.5/32 [0/3] uptime 00:09:28, expires 00:02:32
    via 167.1.135.3, Virtual-Access2
167.1.135.0/24 [0/3] uptime 00:09:28, expires 00:02:44
    via 167.1.45.4, Multilink1
167.1.135.3/32 [0/4] uptime 00:08:28, expires 00:02:32
    via 167.1.135.3, Virtual-Access2
167.1.135.5/32 [0/3] uptime 00:09:28, expires 00:02:44
    via 167.1.45.4, Multilink1
204.12.1.0/24 [0/2] uptime 00:09:28, expires 00:02:32
    via 167.1.135.3, Virtual-Access2
```



```
Rack1R3#show ip dvmrp route
DVMRP Routing Table - 9 entries
150.1.0.0/16 [0/32] uptime 00:11:21, expires 00:02:57
    via 167.1.135.5, Virtual-Access3
167.1.4.0/24 [0/2] uptime 00:11:21, expires 00:02:39
    via 167.1.34.4, Serial1/1.34
167.1.5.0/24 [0/2] uptime 00:11:03, expires 00:02:57
    via 167.1.135.5, Virtual-Access3
167.1.34.0/24 [0/3] uptime 00:11:21, expires 00:02:57
    via 167.1.135.5, Virtual-Access3
167.1.45.4/32 [0/2] uptime 00:11:03, expires 00:02:57
    via 167.1.135.5, Virtual-Access3
167.1.45.5/32 [0/2] uptime 00:11:21, expires 00:02:39
    via 167.1.34.4, Serial1/1.34
167.1.135.0/24 [0/2] uptime 00:11:03, expires 00:02:57
    via 167.1.135.5, Virtual-Access3
167.1.135.3/32 [0/3] uptime 00:11:03, expires 00:02:39
    via 167.1.34.4, Serial1/1.34
167.1.135.5/32 [0/4] uptime 00:10:03, expires 00:02:57
    via 167.1.135.5, Virtual-Access3
```

Task 5.2 Breakdown

Configuring the tunnel and adding DMVRP routing to the interfaces is fairly straightforward. Make sure that you are careful about the interfaces themselves. When using logical interfaces, such as the Virtual Template or Multilink interface, then the multicast settings need to be applied on the logical interface. Section 5.1 stated that PIM was enabled on the Serial interfaces. Creating logical interfaces does not mean that the logical interfaces will inherit these settings, so you need to configure the logical interfaces for sparse mode. You can remove the sparse mode from the physical interfaces if you wish, and just have it configured on the logical interfaces. You should see PIM neighbors, and should see DVMRP routes from both interfaces that connect these devices.

Task 5.3 Solution

R4:

```
interface Tunnel0
 ip dvmrp metric 1 list VLAN4_AND_VLAN5
 ip dvmrp summary-address 167.1.4.0 255.255.254.0
 no ip dvmrp auto-summary
 ip dvmrp metric 0 list ALL
!
ip access-list standard VLAN4_AND_VLAN5
 permit 167.1.4.0 0.0.0.255
 permit 167.1.5.0 0.0.0.255
ip access-list standard ALL
 permit 0.0.0.0 255.255.255.255
```

Task 5.3 Verification

Verify the summary generation. Apply the configuration below to R4 Serial0/0 temporarily

R4:

```
!
interface Serial0/0
 ip dvmrp metric 1 list VLAN4_AND_VLAN5
 ip dvmrp summary-address 167.1.4.0 255.255.254.0
 no ip dvmrp auto-summary
```

Verify the DVMRP routes on R3:

```
Rack1R3#show ip dvmrp route interface s1/1.34
DVMRP Routing Table - 9 entries
167.1.4.0/23 [0/2] uptime 00:01:45, expires 00:02:14
    via 167.1.34.4, Serial1/1.34
167.1.135.0/24 [0/3] uptime 00:07:50, expires 00:02:10
    via 167.1.34.4, Serial1/1.34
167.1.135.3/32 [0/3] uptime 00:07:50, expires 00:02:10
    via 167.1.34.4, Serial1/1.34
```

R4:

```
interface Serial0/0
 no ip dvmrp metric 1 list VLAN4_AND_VLAN5
 no ip dvmrp summary-address 167.1.4.0 255.255.254.0
 ip dvmrp auto-summary
```


Task 6.1 Solution

R4:

```
ip tcp intercept list 100
ip tcp intercept connection-timeout 30

ip tcp intercept max-incomplete low 500 high 1000
!
access-list 100 permit tcp any host 167.1.4.119
```

Task 6.1 Verification

For testing, you can temporarily change the address of SW4 on the VLAN. Telnet to protected servers from R3, and R5:

```
Rack1R3#telnet 167.1.4.119 80
Trying 167.1.4.119, 80 ... Open

Rack1R5#telnet 167.1.4.119 80
Trying 167.1.4.119, 80 ... Open

Rack1R4#show tcp intercept connections
Incomplete:
Client            Server                State    Create   Timeout  Mode

Established:
Client            Server                State    Create   Timeout  Mode
167.1.45.5:26969  167.1.4.119:80        ESTAB    00:00:10 23:59:49 I
167.1.34.3:27455  167.1.4.119:80        ESTAB    00:00:26 23:59:35 I
```

Task 6.2 Solution

R6:

```
interface Serial0/0/0
 ip access-group FROM_BB1 in
!
ip access-list extended FROM_BB1
 deny   ip any any option any-options
 permit ip any any
```


Task 6.2 Verification

To verify, issue ping with ip options enabled from BB1:

```
BB1>ping
Protocol [ip]:
Target IP address: 54.1.1.6
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface:
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]: T
Number of timestamps [ 9 ]:
Loose, Strict, Record, Timestamp, Verbose[TV]:
Sweep range of sizes [n]:
Type escape sequence to abort.


Sending 5, 100-byte ICMP Echos to 54.1.1.6, timeout is 2 seconds:
Packet has IP options:  Total option bytes= 40, padded length=40
 Timestamp: Type 0.  Overflows: 0 length 40, ptr 5
  >>Current pointer<<
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)

Unreachable from 54.1.1.6.  Received packet has options
 Total option bytes= 40, padded length=40
 Timestamp: Type 0.  Overflows: 0 length 40, ptr 5
  >>Current pointer<<
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
```



```
Unreachable from 54.1.1.6.  Received packet has options
 Total option bytes= 40, padded length=40
 Timestamp: Type 0.  Overflows: 0 length 40, ptr 5
  >>Current pointer<<
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
  Time= 16:00:00.000 PST (00000000)
<output omitted>
```

Alternatively, there is also a global command "**ip options drop**" that could be used to drop packets with ip options. Using that command on R6 would have slightly different output on BB1, since the traffic is dropped and unreachables are not sent back to BB1. The problem with this second solution is that it will drop IP options packets regardless of the interface being received on and the task requires that only packets coming from BB1 to be subject to the policy.

```
Packet has IP options:  Total option bytes= 40, padded length=40
 Timestamp: Type 0.  Overflows: 0 length 40, ptr 5
  >>Current pointer<<
  Time= 17:00:00.000 PDT (00000000)
  Time= 17:00:00.000 PDT (00000000)
  Time= 17:00:00.000 PDT (00000000)
  Time= 17:00:00.000 PDT (00000000)
  Time= 17:00:00.000 PDT (00000000)
  Time= 17:00:00.000 PDT (00000000)
  Time= 17:00:00.000 PDT (00000000)
  Time= 17:00:00.000 PDT (00000000)
  Time= 17:00:00.000 PDT (00000000)

Request 0 timed out
Request 1 timed out
Request 2 timed out
Request 3 timed out
Request 4 timed out
Success rate is 0 percent (0/5)
```


Verify that packets without IP options are not dropped:

```
BB1>ping 54.1.1.6

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 = 32/33/36 ms
```

Task 6.3 Solution

Note

In order to configure views, you need to use the global command “enable view” and enter the enable password first.

```
Rack1R4#enable view
Password:

Rack1R4#
```

R4:

```
aaa new-model
username OPERATOR password CISCO
username ADMIN privilege 15 password CISCO
enable secret CISCO
!
! Local authentication is needed to configure the roles.
!
aaa authentication login default none
aaa authentication login VTY local
aaa authorization exec VTY local
!

parser view HTTP
 secret CISCO
 commands configure include-exclusive all ip http
 commands exec include show run
 commands exec include config t
!
username OPERATOR view HTTP

line vty 0 4
 login authentication VTY
 authorization exec VTY
```


Task 6.3 Breakdown

Role-based CLI allows you to assign commands to views. Views can then be grouped with other views, creating a superview, or can be assigned to users. When configuring Role based CLI, you need to enable AAA, and login using a username and password. The command "**enable view**" will allow you to configure the views with the **parser view** command. Similarly to the functionality seen when moving privilege levels, the command **show run** is a special case and will only allow the user to see items that they have the correct privilege to configure.

Task 6.3 Verification

```
Rack1R4#telnet 150.1.4.4
Trying 150.1.4.4 ... Open


User Access Verification

Username: OPERATOR
Password: CISCO

Rack1R4>show parser view
Current view is 'HTTP'
Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)>?
Configure commands:
  do    To run exec commands in config mode
  exit  Exit from configure mode
  ip    Global IP configuration subcommands

Rack1R4(config)#ip ?
Global IP configuration subcommands:
  http  HTTP server configuration

Rack1R4(config)#ip http ?
  access-class                  Restrict http server access by access-
class
  active-session-modules        Set up active http server session
modules
  authentication                Set http server authentication method
  client                        Set http client parameters
  max-connections               Set maximum number of concurrent http
server
                                connections
  path                          Set base path for HTML
  port                          Set http port
  secure-active-session-modules Set up active http secure server
session
                                modules

<snip>
```

```
Rack1R4>show run
Building configuration...

Current configuration : 60 bytes
!
!
!
!
no ip http server
no ip http secure-server
!
!
end

Rack1R4>
```

Task 6.4 Solution

R6:

```
ip traffic-export profile BB1
 interface FastEthernet0/0
 bidirectional
 mac-address 0060.0060.0060
 incoming sample one-in-every 50
 outgoing sample one-in-every 20

interface Serial 0/0/0
 ip traffic-export apply BB1
```

SW3:

```
mac-address 60.60.60 vlan 1363 drop
```


Task 6.4 Verification

For testing, enable debug ip traffic-export events and generate some traffic on the interface. (PING traffic from R6 to BB1 is sufficient) You can also look at the output of **show ip traffic-export**.

```
Rack1R6#ping 54.1.1.254 re 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 54.1.1.254, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max =
28/32/44 ms

RITE: exported output packet # 60
RITE: exported input packet # 30

Rack1R6#show ip traffic-export
Router IP Traffic Export Parameters
Monitored Interface             Serial0/0/0
        Export Interface                FastEthernet0/0
        Destination MAC address 0060.0060.0060
        bi-directional traffic export is on
Output IP Traffic Export Information    Packets/Bytes Exported    5/500
        Packets Dropped           101
        Sampling Rate             one-in-every 20 packets
        No Access List configured
Input IP Traffic Export Information     Packets/Bytes Exported    4/359
        Packets Dropped           216
        Sampling Rate             one-in-every 50 packets
        No Access List configured
        Profile BB1 is Active
```


Task 7.1 Solution

R6:

```
username NOC privilege 15 password 0 CISCO
username NOC autocommand menu NOC
!
menu NOC title #
Menu for Level 1 NOC users
#
menu NOC prompt #
Choose your selection:
#
menu NOC text 1. View Current Configuration
menu NOC command 1. show running-config
menu NOC text 2. Backup Current Configuration
menu NOC command 2. copy running-config
https://NOC:CISCO@167.1.5.115:8080/CONFIGS/R6_CONFIG.txt
menu NOC text 3. Exit
menu NOC command 3. exit
!
line vty 0 807
 login local
```


Task 7.1 Verification

Verify the menu:

```
Rack1R6#telnet 150.1.6.6
Trying 150.1.6.6 ... Open


User Access Verification

Username: NOC
Password: <CISCO>
Menu for Level 1 NOC users


    1.          View Current Configuration

    2.          Backup Current Configuration

    3.          Exit
<2>

Address or name of remote host [167.1.5.115]?
Destination filename [CONFIGS/R6_CONFIG.txt]?
%Error writing https://NOC:CISCO@167.1.5.115:8080/CONFIGS/R6_CONFIG.txt
(I/O error)
```

Task 4.1 Breakdown

In order to view the entire configuration, the user will need to be at privilege level 15. Make sure that you test your menu after configuring. In general, it is recommended to test from a telnet session, since forgetting the 'exit' option can prevent you from being able to exit.

Task 7.2 Solution

R2:

```
interface Loopback0
 ip nat inside
!
interface FastEthernet0/0
 ip address 172.16.0.2 255.255.255.0 secondary
 ip nat outside
 ip policy route-map POLICY
!
ip nat pool INSIDE_GLOBAL 167.1.27.100 167.1.27.199 netmask
255.255.255.0
ip nat inside source list INSIDE_LOCAL pool INSIDE_GLOBAL
!
ip access-list standard INSIDE_LOCAL
 permit 172.16.0.0 0.0.0.255
!
route-map POLICY permit 10
 match ip address INSIDE_LOCAL
 set interface Loopback0
```

Task 7.2 Verification

```
Rack1R2#debug ip nat detailed
IP NAT detailed debugging is on
Rack1R2#debug ip policy
Policy routing debugging is on
```

Configure SW1 to simulate packets from the virtual host:

```
SW1:
!
ip local policy route-map LOCAL
!
ip access-list standard LOCAL
 permit 172.16.0.0 0.0.0.255
!
route-map LOCAL permit 10
 match ip address LOCAL
 set ip default next-hop 167.1.27.2
!
interface FastEthernet0/14
 ip address 172.16.0.8 255.255.255.0 secondary

Rack1SW1#ping 167.1.13.3 source 172.16.0.8

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 167.1.13.3, timeout is 2 seconds:
Packet sent with a source address of 172.16.0.8
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 80/83/84 ms
```

View R2's debugging output:

```
Rack1R2#
IP: s=172.16.0.8 (FastEthernet0/0), d=167.1.13.3, len 100, policy match
IP: route map POLICY, item 10, permit
IP: s=172.16.0.8 (FastEthernet0/0), d=167.1.13.3 (Loopback0), len 100,
policy routed
IP: FastEthernet0/0 to Loopback0 167.1.13.3
NAT: installing alias for address 167.1.27.100
NAT: i: icmp (172.16.0.8, 4) -> (167.1.13.3, 4) [20]
NAT: s=172.16.0.8->167.1.27.100, d=167.1.13.3 [20]
```


Note the return packets:

```
NAT*: o: icmp (167.1.13.3, 4) -> (167.1.27.100, 4) [21]
NAT*: s=167.1.13.3, d=167.1.27.100->172.16.0.8 [21]
IP: s=167.1.13.3 (FastEthernet0/0), d=172.16.0.8 (FastEthernet0/0), len
100, policy rejected -- normal forwarding

Rack1R2#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside
global
--- 167.1.27.100       172.16.0.8         ---                ---

Rack1R3#
ICMP: echo reply sent, src 167.1.13.3, dst 167.1.27.100
ICMP: echo reply sent, src 167.1.13.3, dst 167.1.27.100
ICMP: echo reply sent, src 167.1.13.3, dst 167.1.27.100
ICMP: echo reply sent, src 167.1.13.3, dst 167.1.27.100
ICMP: echo reply sent, src 167.1.13.3, dst 167.1.27.100

SW1:
no ip local policy route-map LOCAL
no ip access-list standard LOCAL
!
no route-map LOCAL permit 10
!
interface FastEthernet1/8
 no ip address 172.16.0.8 255.255.255.0 secondary

R2:
undebug all
```

Task 7.3 Solution

```
R5:
ip icmp rate-limit unreachable 5000
```

Task 7.3 Verification

Ping the unreachable destination from R4:

```
Rack1R4#ping 167.1.8.8 repeat 10

Type escape sequence to abort.
Sending 10, 100-byte ICMP Echos to 167.1.8.8, timeout is 2 seconds:
U...U...U.
Success rate is 0 percent (0/10)
```

Without rate-limit configuration you would get:

```
Rack1R4#ping 167.1.8.8 repeat 10

Type escape sequence to abort.
Sending 10, 100-byte ICMP Echos to 167.1.8.8, timeout is 2 seconds:
UUUUUUUUUU
Success rate is 0 percent (0/10)
```

Task 7.3 Breakdown

The default value for the 'ICMP rate limit unreachable' command is 500, which means that an unreachable will only be sent once every 500 milliseconds.

Task 7.4 Solution

R1:

```
track 1 interface Serial0/0 line-protocol
!
interface FastEthernet0/0
 standby 1 ip 204.12.1.100
 standby 1 priority 101
 standby 1 track 1
 standby 1 preempt
```

R3:

```
interface FastEthernet0/0
 standby 1 ip 204.12.1.100
 standby 1 preempt
```

R6:

```
interface FastEthernet0/0
 standby 1 ip 204.12.1.100
 standby 1 preempt
 standby 1 track Serial0/0/0
```

Task 7.4 Breakdown

R6 will be the active router over R3 if their priorities are the same, since R6's IP address is numerically higher. If this weren't the case, R6 would require a higher HSRP priority than R3.

Task 7.4 Verification

```
Rack1R1#show standby
FastEthernet0/0 - Group 1
  State is Active
    2 state changes, last state change 00:02:31
  Virtual IP address is 204.12.1.100
  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 1.890 secs
  Preemption enabled
  Active router is local
  Standby router is 204.12.1.6, priority 100 (expires in 9.699 sec)
  Priority 101 (configured 101)
    Track object 1 state Up decrement 10
  IP redundancy name is "hsrp-Fa0/0-1" (default)
```



```
Rack1R6#show standby
FastEthernet0/0 - Group 1
  State is Standby
    1 state change, last state change 00:01:23
  Virtual IP address is 204.12.1.100
  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 1.328 secs
  Preemption enabled
  Active router is 204.12.1.1, priority 101 (expires in 10.304 sec)
  Standby router is local
  Priority 100 (default 100)
    Track interface Serial0/0/0 state Up decrement 10
  Group name is "hsrp-Fa0/0-1" (default)

Rack1R3#show standby
FastEthernet0/0 - Group 1
  State is Listen
    2 state changes, last state change 00:02:15
  Virtual IP address is 204.12.1.100
  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
  Preemption enabled
  Active router is 204.12.1.1, priority 101 (expires in 7.191 sec)
  Standby router is 204.12.1.6, priority 100 (expires in 9.924 sec)
  Priority 100 (default 100)
  IP redundancy name is "hsrp-Fa0/0-1" (default)

Rack1R3#show standby brief
                     P indicates configured to preempt.
                     |
Interface   Grp Prio P State    Active          Standby         Virtual IP
Fa0/0       1   100  P Listen   204.12.1.1      204.12.1.6      204.12.1.100

Rack1R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.

Rack1R1(config)#interface s0/0
Rack1R1(config-if)#shutdown
%HSRP-5-STATECHANGE: FastEthernet0/0 Grp 1 state Active -> Speak
%LINK-5-CHANGED: Interface Serial0/0, changed state to administratively
down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed
state to down

Rack1R1#show track
Track 1
  Interface Serial0/0 line-protocol
  Line protocol is Down (hw admin-down)
    2 changes, last change 00:00:08
  Tracked by:
    HSRP FastEthernet0/0 1
```

```
Rack1R6#show standby
FastEthernet0/0 - Group 1
  State is Active
    2 state changes, last state change 00:01:30
  Virtual IP address is 204.12.1.100
  Active virtual MAC address is 0000.0c07.ac01
    Local virtual MAC address is 0000.0c07.ac01 (v1 default)
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 0.000 secs
  Preemption enabled
  Active router is local
  Standby router is 204.12.1.3, priority 100 (expires in 7.992 sec)
  Priority 100 (default 100)
    Track interface Serial0/0 state Up decrement 10
  IP redundancy name is "hsrp-Gi0/0-1" (default)

Rack1R6#conf t
Enter configuration commands, one per line.  End with CNTL/Z.

Rack1R6(config)#interface s0/0/0
Rack1R6(config-if)#shutdown
%LINK-5-CHANGED: Interface Serial0/0, changed state to administratively
down
%HSRP-5-STATECHANGE: FastEthernet0/0 Grp 1 state Active -> Speak
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed
state to down

Rack1R3#show standby
FastEthernet0/0 - Group 1
  State is Active
    4 state changes, last state change 00:00:09
  Virtual IP address is 204.12.1.100
  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 2.848 secs
  Preemption enabled
  Active router is local
  Standby router is unknown
  Priority 100 (default 100)
  IP redundancy name is "hsrp-Fa0/0-1" (default)
```


Task 8.1 Solution

R4:

```
class-map VIP
 match access-group name VIP
!
policy-map LLQ
 class VIP
   priority percent 99
!
interface FastEthernet0/1
service-policy output LLQ
!
ip access-list extended VIP
 permit ip host 167.1.4.204 any
```

Task 8.1 Breakdown

In some earlier IOS versions, it was necessary to enter the command "max-reserved-bandwidth" when applying a policy with items that added up to more than 75%. In some IOS versions, you can specify a priority percentage of 100. In other versions, you may receive the error "Sum total of class bandwidths exceeds 99 percent." Here, we have specified a percentage of 99.

Task 8.1 Verification

Verify the LLQ configuration:

```
Rack1R4#show policy-map interface fastEthernet 0/1
 FastEthernet0/1

  Service-policy output: LLQ

    queue stats for all priority classes:

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

    Class-map: VIP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name VIP
      Priority: 99% (99000 kbps), burst bytes 2475000, b/w exceed
drops: 0


    Class-map: class-default (match-any)
      3 packets, 531 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 3/213
```


Task 8.2 Solution

R6:

```
interface Serial0/0/0
 custom-queue-list 1
!
access-list 182 permit tcp host 167.1.4.119 eq www any
!
queue-list 1 protocol ip 1 list 182
queue-list 1 default 2
```

Task 8.2 Verification

Verify that there are only two queues in the custom queue configuration:

```
Rack1R6#show queueing custom
Current custom queue configuration:

List    Queue  Args
1       2      default
1       1      protocol ip          list 182
```


Lab 15 Solutions

Task 1.1 Solution

SW2:
```
interface FastEthernet0/4
 switchport access vlan 254
 switchport mode dot1q-tunnel
```

SW3:
```
interface FastEthernet0/5
 switchport access vlan 254
 switchport mode dot1q-tunnel
```

SW4:
```
interface FastEthernet0/4
 switchport access vlan 254
 switchport mode dot1q-tunnel
```

R4:
```
interface FastEthernet0/0.45
 encapsulation dot1Q 45
 ip address 130.1.45.4 255.255.255.0
!
interface FastEthernet0/1.54
 encapsulation dot1Q 54
 ip address 130.1.54.4 255.255.255.0
```

R5:
```
interface FastEthernet0/1.45
 encapsulation dot1q 45
 ip address 130.1.45.5 255.255.255.0
!
interface FastEthernet0/1.54
 encapsulation dot1Q 54
 ip address 130.1.54.5 255.255.255.0
```

SW1, SW2:
```
system mtu 1504
System mtu routing 1500
```

SW3, SW4:
```
system mtu 1504
```

SW4:
```
access-list hardware program nonblocking
```

SW3:
```
interface FastEthernet0/24
  duplex half
  speed 10
  storm-control broadcast level 7.50
```


Task 1.1 Verification

```
Rack1R5#ping 130.1.54.4 size 1600

Type escape sequence to abort.
Sending 5, 1600-byte ICMP Echos to 130.1.54.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/4 ms

Rack1R5#ping 130.1.45.4 size 1600

Type escape sequence to abort.
Sending 5, 1600-byte ICMP Echos to 130.1.45.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms
Rack1R5#
```

Note: In order to properly pass traffic that is larger than 1500 bytes, you will need to adjust the MTU on the switches. Depending on the trunks that you have configured, and the spanning tree bridge election, SW1 may be in the transit path.

```
Rack1SW3#show interfaces f0/24
FastEthernet0/24 is up, line protocol is up (connected)
  Hardware is Fast FastEthernet, address is 0016.9d31.839a (bia
0016.9d31.839a)
  MTU 1504 bytes, BW 10000 Kbit, DLY 1000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Half-duplex, 10Mb/s, media type is 10/100BaseTX

Rack1SW2#show dot1q-tunnel

dot1q-tunnel mode LAN Port(s)
-----------------------------
Fa0/4

Rack1SW3#show dot1q-tunnel

dot1q-tunnel mode LAN Port(s)
-----------------------------
Fa0/5

Rack1SW3#show storm-control broadcast
Interface  Filter State    Upper       Lower       Current
---------  -------------   ----------- ----------- ----------
Fa0/24     Forwarding         7.50%       7.50%       0.00%
```


Task 1.2 Solution

R4:

```
interface Dialer1
 ip address dhcp
 encapsulation ppp
 dialer pool 1
 ip mtu 1492
!
interface fastEthernet 0/1.54
 no ip address
 pppoe enable
 pppoe-client dial-pool-number 1
```

R5:

```
interface fastEthernet 0/1.54
 no ip address
!
interface Virtual-Template1
 ip address 130.1.54.5 255.255.255.0
 peer default ip address dhcp-pool VLAN54
!
ip dhcp pool VLAN54
   host 130.1.54.4 255.255.255.0
   client-identifier
0063.6973.636f.2d30.3031.612e.3663.3662.2e64.3339.612d.4469.31
!
bba-group pppoe MYPPP
 virtual-template 1
!
interface fastEthernet 0/1.54
 pppoe enable group MYPPP
```


Task 1.2 Verification

Here, we have a fairly basic PPPoE configuration. Make sure to adjust the MTU setting, or you may run into a mismatch, preventing the OSPF adjacency.

```
Rack1R5#show pppoe session
     1 session  in LOCALLY_TERMINATED (PTA) State
     1 session  total

Uniq ID  PPPoE  RemMAC          Port                  Source   VA
State
           SID  LocMAC                                       VA-st
      4      3  001b.d422.0b69  Fa0/1.54              Vt1      Vi1.1
PTA
                 001c.581a.dec5  VLAN :54                    UP
Rack1R5#

Rack1R4#show pppoe session
     1 client session

Uniq ID  PPPoE  RemMAC          Port                  Source   VA
State
           SID  LocMAC                                       VA-st
    N/A      3  001c.581a.dec5  Fa0/1.54              Di1      Vi2
UP
                 001b.d422.0b69  VLAN :54                    UP

Rack1R4#show int virtual-access2
Virtual-Access2 is up, line protocol is up
  Hardware is Virtual Access interface
  MTU 1500 bytes, BW 56 Kbit/sec, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation PPP, LCP Open
  Listen: CDPCP
  Open: IPCP
  PPPoE vaccess, cloned from Dialer1
```



```
Rack1R4#show int dialer1
Dialer1 is up, line protocol is up (spoofing)
  Hardware is Unknown
  Internet address is 130.1.54.1/24
  MTU 1500 bytes, BW 56 Kbit/sec, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation PPP, loopback not set
  Keepalive set (10 sec)
  Interface is bound to Vi2

Rack1R4#show  ip ospf neighbor dialer 1

Neighbor ID     Pri   State      Dead Time   Address         Interface
150.1.5.5         0   FULL/  -   00:00:33    130.1.54.5      Dialer1
Rack1R4#ping 130.1.54.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 130.1.54.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
```

Task 2.1 Solution

R1:
```
router eigrp 100
 distance 91 130.1.124.4 0.0.0.0 1
!
access-list 1 permit 130.1.3.0
```

R3:
```
interface FastEthernet0/1
 ip ospf database-filter all out
```

R1:
```
interface FastEthernet0/0
 ip ospf message-digest-key 1 md5 CISCO
!
router ospf 1
 area 51 authentication message-digest
```

SW1:
```
interface Vlan17
 ip ospf message-digest-key 1 md5 CISCO
!
router ospf 1
 area 51 authentication message-digest
```

SW2:
```
!
! VLAN 17 was missing from the initial configs on purpose
!
vlan 17
```


Task 2.1 Verification

Initially, R1 will load balance between R2 and R4. After making the change, R1 will prefer the path via R2. By setting to a value that is just slightly higher, the alternate path can still be used if the primary path is not available.

```
Rack1R1#show ip route 130.1.3.0
Routing entry for 130.1.3.0/24
  Known via "eigrp 100", distance 90, metric 2684416, type internal
  Redistributing via eigrp 100
  Last update from 130.1.124.2 on Serial0/0, 00:00:02 ago
  Routing Descriptor Blocks:
  * 130.1.124.2, from 130.1.124.2, 00:00:02 ago, via Serial0/0
      Route metric is 2684416, traffic share count is 1
      Total delay is 40100 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2

Rack1R1#show ip eigrp topology 130.1.3.0 255.255.255.0
IP-EIGRP (AS 100): Topology entry for 130.1.3.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is
2684416
  Routing Descriptor Blocks:
  130.1.124.2 (Serial0/0), from 130.1.124.2, Send flag is 0x0
      Composite metric is (2684416/2172416), Route is Internal
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 40100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
  130.1.124.4 (Serial0/0), from 130.1.124.4, Send flag is 0x0
      Composite metric is (2684416/2172416), Route is Internal
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 40100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
```



```
Rack1R1#show ip protocols | section eigrp
Routing Protocol is "eigrp 100"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Default networks flagged in outgoing updates
  Default networks accepted from incoming updates
  EIGRP metric weight K1=1, K2=0, K3=1, K4=0, K5=0
  EIGRP maximum hopcount 100
  EIGRP maximum metric variance 1
  Redistributing: eigrp 100
  EIGRP NSF-aware route hold timer is 240s
  Automatic network summarization is not in effect
  Maximum path: 4
  Routing for Networks:
    130.1.124.1/32
    150.1.1.1/32
  Routing Information Sources:
    Gateway         Distance      Last Update
    130.1.124.4           91      00:07:14
    130.1.124.2           90      00:07:15
  Distance: internal 90 external 170
    Address         Wild mask       Distance  List
    130.1.124.4           0.0.0.0         91  1
```

```
Rack1R3#show ip ospf interface fastEthernet 0/1
FastEthernet0/1 is up, line protocol is up
  Internet Address 130.1.33.3/24, Area 0
  Process ID 1, Router ID 150.1.3.3, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State WAITING, Priority 1
  No designated router on this network
  No backup designated router on this network
  Database-filter all out
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:05
    Wait time before Designated router selection 00:00:35
  Supports Link-local Signaling (LLS)
  Index 1/1, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 0, maximum is 0
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 0, Adjacent neighbor count is 0
  Suppress hello for 0 neighbor(s)
```


Authentication can be configured under the process or under the interface. In both cases, the key is configured on the interface.

```
Rack1SW1#show ip ospf interface vlan 17
Vlan17 is up, line protocol is up
<snip>
    Adjacent with neighbor 150.1.1.1  (Backup Designated Router)
  Suppress hello for 0 neighbor(s)
  Message digest authentication enabled
    Youngest key id is 1
```


Task 2.2 Solution

R1:

```
!
router eigrp 100
 redistribute ospf 1 metric 10000 10 255 1 1500
!
router ospf 1
 redistribute eigrp 100 subnets
```

R3:

```
!
route-map OSPF_TO_EIGRP deny 10
 match tag 90
!
route-map OSPF_TO_EIGRP permit 20
 set tag 110
!
route-map EIGRP_TO_OSPF deny 10
 match tag 110
!
route-map EIGRP_TO_OSPF permit 20
 set tag 90
!
router eigrp 100
 redistribute ospf 1 metric 1500 10 255 1 1500 route-map OSPF_TO_EIGRP

!
router ospf 1
 redistribute eigrp 100 subnets route-map EIGRP_TO_OSPF
 distance ospf external 171
```

R4:

```
!
route-map OSPF_TO_EIGRP deny 10
 match tag 90
!
route-map OSPF_TO_EIGRP permit 20
 set tag 110
!
route-map EIGRP_TO_OSPF deny 10
 match tag 110
!
route-map EIGRP_TO_OSPF permit 20
 set tag 90
!
router eigrp 100
 redistribute ospf 1 metric 1500 10 255 1 1500 route-map OSPF_TO_EIGRP

!
router ospf 1
 redistribute eigrp 100 subnets route-map EIGRP_TO_OSPF
 distance ospf external 171
```


R6:

```
router eigrp 100
 redistribute eigrp 10
```

Task 2.2 Verification

Use the following tcl script to verify full reachability on all tcl-enabled routers.

```
foreach i {
130.1.17.1
150.1.1.1
130.1.124.1
130.1.234.2
150.1.2.2
130.1.26.2
130.1.124.2
130.1.234.3
130.1.3.3
150.1.3.3
130.1.33.3
130.1.35.3
130.1.234.4
150.1.4.4
130.1.45.4
130.1.54.4
130.1.124.4
130.1.5.5
150.1.5.5
130.1.35.5
130.1.45.5
130.1.54.5
192.10.1.5
54.1.1.6
130.1.6.6
150.1.6.6
130.1.26.6
130.1.17.7
150.1.7.7
204.12.1.7
130.1.78.7
130.1.8.8
150.1.8.8
130.1.78.8
} { ping $i  }
```

Note that you might not be able to ping the local unmapped IP addresses on the Frame-Relay interfaces, but this is not considered to be an issue. Also, the link between SW2 and SW3 is not advertised into any IGP and thus is not subject to connectivity test.

Task 2.3 Solution

R1:

```
router bgp 65178
 address-family ipv4
 neighbor 130.1.124.4 capability orf prefix-list receive
```

R4:

```
router bgp 200
 address-family ipv4
 neighbor 130.1.124.1 capability orf prefix-list send
 neighbor 130.1.124.1 prefix-list ORFFILTER in
!
ip prefix-list ORFFILTER seq 5 permit 28.119.16.0/23 le 24
```

R2:

```
router bgp 65026
address-family ipv4
 neighbor 130.1.234.3 capability orf prefix-list receive
```

R3:

```
router bgp 200
 address-family ipv4
  neighbor 130.1.234.2 capability orf prefix-list send
  neighbor 130.1.234.2 prefix-list ORFFILTER in
!
ip prefix-list ORFFILTER seq 5 permit 112.0.0.0/5 ge 8 le 8
```


Task 2.3 Verification

```
Rack1R1#show ip bgp neighbors 130.1.124.4 | beg ORF
    Outbound Route Filter (ORF) type (128) Prefix-list:
      Send-mode: received
      Receive-mode: advertised
  Outbound Route Filter (ORF): received (1 entries)
                                 Sent       Rcvd
  Prefix activity:               ----       ----
    Prefixes Current:               2          4 (Consumes 208 bytes)
    Prefixes Total:                 2          4
    Implicit Withdraw:              0          0
    Explicit Withdraw:              0          0
    Used as bestpath:             n/a          4
    Used as multipath:            n/a          0

                                  Outbound    Inbound
  Local Policy Denied Prefixes:   --------    -------
    AS_PATH loop:                      n/a         10
    Bestpath from this peer:             4        n/a
    ORF prefix-list:                     8        n/a
    Total:                              12         10
  Number of NLRIs in the update sent: max 8, min 0
<snip>

Rack1R1#show ip bgp neighbors 130.1.124.4 received prefix-filter
Address family: IPv4 Unicast
ip prefix-list 130.1.124.4: 1 entries
   seq 5 permit 28.119.16.0/23 le 24

Rack1R2#show ip bgp neighbors 130.1.234.3 received prefix-filter
Address family: IPv4 Unicast
ip prefix-list 130.1.234.3: 1 entries
   seq 5 permit 112.0.0.0/5 le 8
```

Task 3.1 Solution

R6:

```
interface FastEthernet0/1
 ipv6 nd suppress-ra
```

R2:

```
interface Tunnel25
 ipv6 address 2001:130:1:25::2/64
 tunnel source Loopback0
 tunnel destination 150.1.5.5
 tunnel mode ipv6ip
```

R5:

```
interface Tunnel25
 ipv6 address 2001:130:1:25::5/64
 tunnel source Loopback0
 tunnel destination 150.1.2.2
 tunnel mode ipv6ip
```


Task 3.1 Verification

```
Rack1R5#ping 2001:130:1:25::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:130:1:25::2, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 132/133/136
ms
Rack1R5#
```

Task 3.2 Solution

R2:

```
interface Loopback0
 ipv6 ospf 1 area 26
!
interface FastEthernet0/0
 ipv6 ospf 1 area 26
!
ipv6 route 2001:130:1:5::/64 Tunnel25
!
ipv6 router ospf 1
 redistribute static
```

R6:

```
interface Loopback0
 ipv6 ospf 1 area 26
!
interface FastEthernet0/0
 ipv6 ospf 1 area 26
!
interface FastEthernet0/1
 ipv6 ospf 1 area 26
```


Task 3.3 Solution

R5:

```
ipv6 route 2001:100::/25 Tunel 25
```

Tasks 3.2-3.3 Verification

```
Rack1R2#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
150.1.6.6         1   FULL/DR         00:00:38    4               FastEthernet0/0
```

```
Rack1R6#show ipv6 route ospf
IPv6 Routing Table - Default - 8 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   2001:130:1:2::2/128 [110/1]
     via FE80::212:1FF:FE0E:B8A0, FastEthernet0/1
OE2 2001:130:1:5::/64 [110/20]
     via FE80::212:1FF:FE0E:B8A0, FastEthernet0/1
```

```
Rack1R6#ping 2001:130:1:5::5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:130:1:5::5, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 136/139/141
ms
```

Verify IPv6 connectivity between R2, R5 and R6; note that there is no connectivity between Tunnel25 and R6 IPv6 attached networks as it's not requested.

```
Rack1R2#show ipv6 interface brief loopback 0
Loopback0                  [up/up]
    FE80::211:92FF:FE0E:5300
    2001:130:1:2::2
```

```
Rack1R6#show ipv6 interface brief loopback 0
Loopback0                  [up/up]
    FE80::219:56FF:FED4:F922
    2001:150:1:6::6
```



```
Rack1R5#ping 2001:130:1:6::6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:130:1:6::6, timeout is 2
seconds:
 ...
Success rate is 0 percent (0/5)

Rack1R5#ping 2001:130:1:6::6 source fastEthernet 0/0.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:130:1:6::6, timeout is 2
seconds:
Packet sent with a source address of 2001:130:1:5::5
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 136/136/140
ms

Rack1R5#ping 2001:130:1:2::2 source fastEthernet 0/0.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:130:1:2::2, timeout is 2
seconds:
Packet sent with a source address of 2001:130:1:5::5
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 132/132/136
ms
```

Task 4.1 Solution

R1:

```
ip vrf TESTVRF
 rd 1:1

interface tunnel 12
 tunnel source 150.1.1.1
 tunnel dest 150.1.2.2
 ip vrf forw TESTVRF
 ip address 130.1.124.1 255.255.255.252
```

R2:

```
ip vrf TESTVRF
 rd 2:2
!
interface tunnel 12
 tunnel source 150.1.2.2
 tunnel dest 150.1.1.1
 ip vrf forw TESTVRF
 ip address 130.1.124.2 255.255.255.252
```


Task 4.1 Verification

```
Rack1R1#show ip route vrf TESTVRF

Routing Table: TESTVRF
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     130.1.0.0/30 is subnetted, 1 subnets
C       130.1.124.0 is directly connected, Tunnel12
Rack1R1#

Rack1R1#ping vrf TESTVRF 130.1.124.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 130.1.124.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 72/72/72 ms
Rack1R1#
```

Task 5.1 Solution

R1:

```
ip pim rp-candidate Serial0/0 group-list R1_GROUP
!
ip access-list standard R1_GROUP
 permit 224.0.0.0 7.255.255.255
```

R2:

```
ip pim rp-candidate Serial0/0.124 group-list R2_GROUP
!
ip access-list standard R2_GROUP
 permit 232.0.0.0 7.255.255.255
```

R3:

```
ip pim bsr-candidate Serial1/0 0
```


Task 5.1 Verification

```
Rack1R3#show ip pim rp mapping
PIM Group-to-RP Mappings
This system is the Bootstrap Router (v2)

Group(s) 224.0.0.0/5
  RP 130.1.124.1 (?), v2
    Info source: 130.1.124.1 (?), via bootstrap, priority 0, holdtime
150
         Uptime: 00:00:20, expires: 00:02:08
Group(s) 232.0.0.0/5
  RP 130.1.124.2 (?), v2
    Info source: 130.1.234.2 (?), via bootstrap, priority 0, holdtime
150
         Uptime: 00:00:06, expires: 00:02:22

Rack1R6#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/5
  RP 130.1.124.1 (?), v2
    Info source: 130.1.234.3 (?), via bootstrap, priority 0, holdtime
150
         Uptime: 00:00:05, expires: 00:02:21
Group(s) 232.0.0.0/5
  RP 130.1.124.2 (?), v2
    Info source: 130.1.234.3 (?), via bootstrap, priority 0, holdtime
150
         Uptime: 00:00:05, expires: 00:02:23
```

Task 5.2 Solution

SW2:

```
interface Vlan8
 ip igmp access-group DENY_R2_GROUP
 ip igmp limit 3
!
ip access-list standard DENY_R2_GROUP
 deny 232.0.0.0 7.255.255.255
 permit any
```


Task 5.2 Verification

```
Rack1SW2#show ip igmp interface vlan 8
Vlan8 is up, line protocol is up
  Internet address is 130.1.8.8/24
  IGMP is enabled on interface
  Current IGMP host version is 2
  Current IGMP router version is 2
  IGMP query interval is 60 seconds
  IGMP querier timeout is 120 seconds
  IGMP max query response time is 10 seconds
  Last member query count is 2
  Last member query response interval is 1000 ms
  Inbound IGMP access group is DENY_R2_GROUP
  IGMP activity: 0 joins, 0 leaves
  Interface IGMP State Limit : 0 active out of 3 max
  Multicast routing is enabled on interface
  Multicast TTL threshold is 0
  Multicast designated router (DR) is 130.1.8.8 (this system)
  IGMP querying router is 130.1.8.8 (this system)
  No multicast groups joined by this system
```

Task 5.3 Solution

R5:

```
ipv6 multicast-routing

interface Fa0/1.45
 ipv6 address 2001:130:1:45::5/64
```

R4:

```
ipv6 multicast-routing

interface fasEthernet0/0.45
 ipv6 address 2001:130:45::4/64
```

R6:

```
ipv6 multicast-routing
```


Task 5.3 Verification

```
Rack1R4#show ipv6 pim int fa0/0.45
Interface          PIM  Nbr   Hello  DR
                        Count Intvl  Prior

FastEthernet0/0.45 on   1     30     1
    Address: FE80::21B:D4FF:FE22:B68
    DR     : FE80::21C:58FF:FE1A:DEC5

Rack1R5#show ipv6 pim int fa0/1.45
Interface          PIM  Nbr   Hello  DR
                        Count Intvl  Prior

FastEthernet0/1.45 on   1     30     1
    Address: FE80::21C:58FF:FE1A:DEC5
    DR     : this system

Rack1R6#show ipv6 pim int fa0/0
Interface          PIM  Nbr   Hello  DR
                        Count Intvl  Prior

FastEthernet0/0    on   0     30     1
    Address: FE80::21A:6CFF:FE30:8A28
    DR     : this system
```

Task 5.4 Solution

R4:

```
!
interface tunnel46
tunnel source loopback0
tunnel destination 150.1.6.6
ipv6 address 2001:130:1:46::4/64
!
ipv6 route 2001:130:1:6::/64 tunnel46 2001:130:1:46::6
ipv6 route 2001:130:1:6::/64 Fa0/0.45 2001:130:1:45::5 2
```


R5:

```
!
interface tunnel56
 tunnel source loopback0
 tunnel destination 150.1.6.6
 ipv6 add 2001:130:1:56::5/64
!
! R5 already has a route to R6 loopback. A more specific
! one is used here so that tunnel between R5 and R6 is preferred over
! the FastEthernet between R5 and R4
!
ipv6 route 2001:130:1:6::/64 tunnel56 2001:130:1:56::6
ipv6 route 2001:130:1:6::/64 Fa0/1.45 2001:130:1:45::4 2
!
interface FastEthernet 0/0.5
 no ipv6 pim
!
interface FastEthernet 0/0.52
 no ipv6 pim
!
interface Tunnel25
 no ipv6 pim
```

R6:

```
!
ipv6 pim bsr candidate bsr 2001:130:1:6::6 priority 192
ipv6 pim bsr candidate rp 2001:130:1:6::6
!
interface tunnel46
 tunnel source loopback0
 tunnel destination 150.1.4.4
 ipv6 address 2001:130:1:46::6/64
!
interface tunnel56
 tunnel source loopback0
 tunnel destination 150.1.5.5
 ipv6 address 2001:130:1:56::6/64
!
interface FastEthernet 0/1
 no ipv6 pim
!
interface Loopback0
 no ipv6 pim
```


Task 5.4 Verification

```
Rack1R6#show ipv6 pim bsr rp-cach
PIMv2 BSR C-RP Cache

BSR Candidate RP Cache

Group(s) FF00::/8, RP count 1
  RP 2001:130:1:6::6 SM
    Priority 192, Holdtime 150
    Uptime: 00:00:23, expires: 00:02:06

Rack1R6#show ipv6 pim bsr candidate-rp
PIMv2 C-RP information
    Candidate RP: 2001:130:1:6::6 SM
      All Learnt Scoped Zones, Priority 192, Holdtime 150
      Advertisement interval 60 seconds
      Next advertisement in 00:00:26

Rack1R6#show ipv6 pim bsr election
PIMv2 BSR information

BSR Election Information
  Scope Range List: ff00::/8
  This system is the Bootstrap Router (BSR)
     BSR Address: 2001:130:1:6::6
     Uptime: 00:20:25, BSR Priority: 192, Hash mask length: 126
     RPF: FE80::219:56FF:FED4:F922,FastEthernet0/0
     BS Timer: 00:00:31
  This system is candidate BSR
      Candidate BSR address: 2001:130:1:6::6, priority: 192, hash mask
length: 126

Rack1R4#show ipv6 pim bsr election
PIMv2 BSR information

BSR Election Information
  Scope Range List: ff00::/8
     BSR Address: 2001:130:1:6::6
     Uptime: 00:18:51, BSR Priority: 192, Hash mask length: 126
     RPF: FE80::9601:606,Tunnel46
     BS Timer: 00:01:21

Rack1R4#show ipv6 rpf 2001:130:1:6::6
RPF information for 2001:130:1:6::6
  RPF interface: Tunnel46
  RPF neighbor: 2001:130:1:46::6
  RPF route/mask: 2001:130:1:6::/64
  RPF type: Unicast
  RPF recursion count: 0
  Metric preference: 1
  Metric: 0
```



```
Rack1R5#show ipv6 pim bsr election
PIMv2 BSR information

BSR Election Information
  Scope Range List: ff00::/8
     BSR Address: 2001:130:1:6::6
     Uptime: 00:19:31, BSR Priority: 192, Hash mask length: 126
     RPF: FE80::9601:606,Tunnel56
     BS Timer: 00:01:41

Rack1R5#show ipv6 rpf 2001:130:1:6::6
RPF information for 2001:130:1:6::6
  RPF interface: Tunnel56
  RPF neighbor: 2001:130:1:56::6
  RPF route/mask: 2001:130:1:6::/64
  RPF type: Mroute
  RPF recursion count: 0
  Metric preference: 1
  Metric: 0

Rack1R5#show ipv6 pim int | i on
Tunnel56           on   1     30     1
FastEthernet0/1.45 on   1     30     1

Rack1R4#show ipv6 pim int | i on
FastEthernet0/0.45 on   1     30     1
Tunnel46           on   1     30     1

Rack1R6#show ipv6 pim int | i on
Tunnel56           on   1     30     1
FastEthernet0/0    on   0     30     1
Tunnel46           on   1     30     1
```


Task 6.1 Solution

R5:

```
class-map match-all SQL
 match protocol sqlserver
 match packet length min 404 max 404
!
policy-map STOP
 class SQL
  drop
!
interface FastEthernet0/0.5
 service-policy input STOP
```

Task 6.1 Verification

```
Rack1R5#show policy-map interface Fa0/0.5
 Ethernet0/0.5

  Service-policy input: STOP

   Class-map: SQL (match-all)
   0 packets, 0 bytes
   5 minute offered rate 0 bps, drop rate 0 bps
   Match: protocol sqlserver
   Match: packet length min 404 max 404
   drop

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

Rack1R5#show ip nbar port-map | section sqlserver
port-map sqlserver                tcp 1433
```


Task 6.2 Solution

R5:

```
ip inspect name CBAC tcp router-traffic
ip inspect name CBAC udp
ip inspect name CBAC icmp router-traffic
!
! Numbered ACL is selected to work with ACL violation accounting
!
ip access-list extended 100
 deny ip any any
!
interface FastEthernet0/0.52
 ip access-group 100 in
 ip inspect CBAC out
```

Task 6.2 Breakdown

By adding the “router-traffic” keywords on the inspections for tcp and ICMP, it is not necessary to have explicit entries for traffic initiated by the router itself. This will allow the BGP adjacency to form, and will allow R5 to ping BB2.

Task 6.2 Verification

```
Rack1R5#show ip inspect config
Session audit trail is disabled
Session alert is enabled
one-minute (sampling period) thresholds are [unlimited : unlimited]
connections
max-incomplete sessions thresholds are [unlimited : unlimited]
max-incomplete tcp connections per host is unlimited. Block-time 0
minute.
tcp synwait-time is 30 sec -- tcp finwait-time is 5 sec
tcp idle-time is 3600 sec -- udp idle-time is 30 sec
tcp reassembly queue length 16; timeout 5 sec; memory-limit 1024 kilo
bytes
dns-timeout is 5 sec
Inspection Rule Configuration
 Inspection name CBAC
    tcp alert is on audit-trail is off timeout 3600
 inspection of router local traffic is enabled
    udp alert is on audit-trail is off timeout 30
    icmp alert is on audit-trail is off timeout 10
 inspection of router local traffic is enabled

Rack1R5#ping 192.10.1.254

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



```
Rack1R5#show ip bgp summary | begin Nei
Neighbor        V AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
130.1.35.3      4 200    2252    2228       51    0    0   1d13h           9
130.1.45.4      4 200    2211    2209       51    0    0   1d12h           0
192.10.1.254    4 254    2220    2208       51    0    0   1d12h           3

Rack1R4#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 = 4/6/8 ms

Rack1R4#ping 192.10.1.254 repeat 10000
Type escape sequence to abort.
Sending 10000, 100-byte ICMP Echos to 192.10.1.254, timeout is 2
seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

<output omitted>

Rack1R5# show ip inspect sessions detail
Established Sessions
 Session 66B922C8 (130.1.45.4:8)=>(192.10.1.254:0) icmp SIS_OPEN
  Created 00:00:02, Last heard 00:00:02
   ECHO request
  Bytes sent (initiator:responder) [7200:7200]
  In  SID 192.10.1.254[0:0]=>130.1.45.4[0:0] on ACL 100  (100 matches)
  In  SID 0.0.0.0[0:0]=>130.1.45.4[3:3] on ACL 100
  In  SID 0.0.0.0[0:0]=>130.1.45.4[11:11] on ACL 100
 Session 66B92590 (192.10.1.5:29978)=>(192.10.1.254:179) tcp SIS_OPEN
  Created 00:01:56, Last heard 00:00:56
  Bytes sent (initiator:responder) [276:181]
  In  SID 192.10.1.254[179:179]=>192.10.1.5[29978:29978] on ACL 100
(14 matches)
```


Task 6.3 Solution

R5:

```
ip inspect hashtable-size 2048
```

Task 6.4 Solution

R6:

```
!
! The task requires filtering traffic destined to the router only
!
object-group network IP_OPTIONS
 host 130.1.6.6
 host 130.1.26.6
 host 54.1.1.6
 host 150.1.6.6
!
ip access-list extended DROP_IP_SOURCE_ROUTE
 deny   ip any object-group IP_OPTIONS option lsr
 deny   ip any object-group IP_OPTIONS option ssr
 permit ip any any
!
interface FastEthernet 0/0
 ip access-group DROP_IP_SOURCE_ROUTE in
!
interface FastEthernet 0/1
 ip access-group DROP_IP_SOURCE_ROUTE in
!
interface Serial 0/0/0
 ip access-group DROP_IP_SOURCE_ROUTE in
```


Task 6.4 Verification

Ping from R2 to both R6 and BB1 with LSR before the configuration is applied; it should be successful:

```
Rack1R2#ping
Protocol [ip]:
Target IP address: 130.1.26.6
Repeat count [5]: 1
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: FastEthernet0/0
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]: L
Source route: 130.1.26.6
Loose, Strict, Record, Timestamp, Verbose[LV]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 130.1.26.6, timeout is 2 seconds:
Packet sent with a source address of 130.1.26.2
Packet has IP options:  Total option bytes= 7, padded length=8
 Loose source route: <*>
   (130.1.26.6)

Reply to request 0 (1 ms).  Received packet has options
 Total option bytes= 8, padded length=8
 Loose source route:
   (130.1.26.6)
   <*>
 End of list

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



```
Rack1R2#ping
Protocol [ip]:
Target IP address: 54.1.1.254
Repeat count [5]: 1
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: FastEthernet0/0
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]: L
Source route: 54.1.1.254
Loose, Strict, Record, Timestamp, Verbose[LV]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 54.1.1.254, timeout is 2 seconds:
Packet sent with a source address of 130.1.26.2
Packet has IP options:  Total option bytes= 7, padded length=8
 Loose source route: <*>
   (54.1.1.254)

Reply to request 0 (81 ms).  Received packet has options
 Total option bytes= 8, padded length=8
 Loose source route:
   (54.1.1.254)
   <*>
 End of list

Success rate is 100 percent (1/1), round-trip min/avg/max = 81/81/81 ms
```


Ping after configuration is applied; now only pings to BB1 should be successful:

```
Rack1R2#debug ip icmp
Rack1R2#ping
Protocol [ip]:
Target IP address: 130.1.26.6
Repeat count [5]: 1
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: FastEthernet0/0
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]: L
Source route: 130.1.26.6
Loose, Strict, Record, Timestamp, Verbose[LV]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 130.1.26.6, timeout is 2 seconds:
Packet sent with a source address of 130.1.26.2
Packet has IP options:  Total option bytes= 7, padded length=8
 Loose source route: <*>
   (130.1.26.6)

Unreachable from 130.1.26.6.  Received packet has options
 Total option bytes= 7, padded length=8
 Loose source route: <*>
   (130.1.26.6)

Success rate is 0 percent (0/1)

Rack1R2#
ICMP: dst (130.1.26.2) administratively prohibited unreachable rcv from
130.1.26.6
```



```
Rack1R2#ping
Protocol [ip]:
Target IP address: 54.1.1.254
Repeat count [5]: 1
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: FastEthernet0/0
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]: L
Source route: 54.1.1.254
Loose, Strict, Record, Timestamp, Verbose[LV]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 54.1.1.254, timeout is 2 seconds:
Packet sent with a source address of 130.1.26.2
Packet has IP options:  Total option bytes= 7, padded length=8
 Loose source route: <*>
   (54.1.1.254)

Reply to request 0 (76 ms).  Received packet has options
 Total option bytes= 8, padded length=8
 Loose source route:
   (54.1.1.254)
   <*>
 End of list

Success rate is 100 percent (1/1), round-trip min/avg/max = 76/76/76 ms

Rack1R2#
ICMP: echo reply rcvd, src 54.1.1.254, dst 130.1.26.2
Rack1R2#undebug ip icmp

Rack1R6#show ip access-lists
Extended IP access list DROP IP SOURCE ROUTE
    10 deny ip any object-group IP_OPTIONS option lsr (1 match)
    20 deny ip any object-group IP_OPTIONS option ssr
    30 permit ip any any (1333 matches)
```


Task 7.1 Solution

First, determine the interface-index for the Serial 0/0 interface. Note that snmp-server needs to be enabled before this.

```
Rack1R1#show snmp mib ifmib ifindex serial 0/0
Interface = Serial0/0, Ifindex = 2
```

You can use the index to construct the SNMP OID “ifEntry.21.2”. Now apply your configuration.

```
R1:
snmp-server host 130.1.17.100 IETRAP
snmp-server ifindex persist
!
rmon event 1 trap IETRAP description "WARNING: Frame Relay Circuit
Congested"
!
rmon event 2 trap IETRAP description "NOTICE: Frame Relay Circuit
Within Normal Utilization"
!
rmon alarm 1 ifEntry.21.2 60 absolute rising-threshold 750 1 falling-
threshold 100 2 owner config
```


Task 7.1 Verification

```
Rack1R1#show rmon alarms
Alarm 1 is active, owned by config
 Monitors ifOutQLen.2 every 60 second(s)
 Taking absolute samples, last value was 0
 Rising threshold is 750, assigned to event 0
 Falling threshold is 100, assigned to event 2
 On startup enable rising or falling alarm

Rack1R1#show rmon events
Event 1 is active, owned by config
 Description is WARNING: Frame Relay Circuit Congested
 Event firing causes trap to community IETRAP,
 last event fired at  0y0w0d,00:00:00,
 Current uptime       0y0w0d,03:39:46
Event 2 is active, owned by config
 Description is WARNING: Frame Relay Circuit Within Normal Utilization
 Event firing causes trap to community IETRAP,
 last event fired at  0y0w0d,03:38:45,
 Current uptime       0y0w0d,03:39:46
```

Task 7.2 Solution

R6:

```
aaa new-model
aaa authentication login NO_AUTH none
!
banner exec %
################################################
########   AS 100 Route View Server   ##########
# Use this device to view the Internet routing #
#     table from the perspective of AS 100     #
################################################
%
!
line vty 0 807
 login authentication NO_AUTH
 privilege level 1
```

Task 7.2 Verification

```
Rack1R6#telnet 150.1.6.6
Trying 150.1.6.6 ... Open

################################################
########   AS 100 Route View Server   ##########
# Use this device to view the Internet routing #
#     table from the perspective of AS 100     #
################################################

Rack1R6>
```


Task 7.3 Solution

R6:

```
line vty 0 807
 transport output none
```

Task 7.3 Verification

```
Rack1R6#telnet 150.1.6.6
Trying 150.1.6.6 ... Open

################################################
########   AS 100 Route View Server   ##########
# Use this device to view the Internet routing #
#     table from the perspective of AS 100     #
################################################

Rack1R6>telnet 150.1.1.1
% telnet connections not permitted from this terminal
```

Task 7.4 Solution

R2:

```
interface FastEthernet0/0
 vrrp 1 ip 130.1.26.1
```

R6:

```
interface FastEthernet0/1
 vrrp 1 ip 130.1.26.1
 vrrp 1 preempt delay minimum 300
```

Task 7.4 Verification

```
Rack1R6#show vrrp all
FastEthernet0/1 - Group 1
  State is Master
  Virtual IP address is 130.1.26.1
  Virtual MAC address is 0000.5e00.0101
  Advertisement interval is 1.000 sec
  Preemption enabled, delay min 300 secs
  Priority is 105
  Master Router is 130.1.26.6 (local), priority is 105
  Master Advertisement interval is 1.000 sec
  Master Down interval is 3.589 sec
```


Task 7.5 Solution

R6:

```
track 1 ip route 200.0.0.0 255.255.255.0 reachability
!
interface FastEthernet0/1
 vrrp 1 track 1 decrement 20
```

Task 7.5 Verification

```
Rack1R6#show track
Track 1
  IP route 200.0.0.0 255.255.255.0 reachability
  Reachability is Up (EIGRP)
    1 change, last change 00:12:31
  First-hop interface is Serial0/0
  Tracked by:
    VRRP FastEthernet0/1 1

Rack1R6#show vrrp
FastEthernet0/1 - Group 1
  State is Master
  Virtual IP address is 130.1.26.1
  Virtual MAC address is 0000.5e00.0101
  Advertisement interval is 1.000 sec
  Preemption enabled, delay min 300 secs
  Priority is 100
    Track object 1 state Up decrement 20
  Master Router is 130.1.26.6 (local), priority is 100
  Master Advertisement interval is 1.000 sec
  Master Down interval is 3.609 sec
```

Task 7.6 Solution

R5:

```
ip accounting-threshold 100
!
interface FastEthernet0/0.52
 ip accounting access-violations
```


Task 7.6 Verification

Note that this task should be verified after the CBAC-based firewall has been configured on R5.

```
Rack1R5#show ip accounting access-violations
   Source           Destination      Packets                 Bytes   ACL
 192.10.1.254     224.0.0.5                2                   152   100

Accounting data age is 0
```

Task 7.7 Solution

R2:

```
interface FastEthernet 0/0
 ip helper-address 130.1.3.100
!
 ip dhcp relay information option
```

R6:

```
interface FastEthernet 0/1
 ip helper-address 130.1.3.100
!
ip dhcp relay information option
```

Task 7.7 Verification

```
Rack1R6#show ip helper-address fastEthernet 0/0
Interface                  Helper-Address  VPN VRG Name        VRG State
FastEthernet0/1            130.1.3.100     0   None              Unknown
```

Task 7.8 Solution

R2:

```
interface FastEthernet 0/0
 ip address 10.1.26.2 255.255.255.0 secondary
!
ip dhcp smart-relay
```

R6:

```
interface FastEthernet 0/1
 ip address 10.1.26.6 255.255.255.0 secondary
!
ip dhcp smart-relay
```


Task 7.9 Solution

R4:

```
!
! Notice that every command below should be on a single line
!
event manager applet EVENT_INTERFACE
 event interface name FastEthernet0/0 parameter receive_throttle entry-
op ge entry-val 3 entry-val-is-increment true poll-interval 60
!
! Execute the command - the result is stored in $_cli_result
!
 action 1.0 cli command "show interface FastEthernet 0/0"
 action 1.1 mail server 130.1.3.100 to noc@ine from router@ine.com cc
admin@ine.com subject " R4 Fa0/0 warning - receive_throttle incremented
over 3" body "show interface command output: $_cli_result"
```

Task 7.9 Verification

```
Rack1R4#show event manager detector interface
No.  Name                 Version   Node        Type
1    interface            01.00     node0/0     RP

Rack1R4#show event manager policy registered
No.  Class     Type    Event Type          Trap  Time Registered
Name
1    applet    user    interface           Off   Wed Dec 9 16:22:08
2009   EVENT_INTERFACE
 name {FastEthernet0/0} parameter {receive_throttle} entry_op ge
entry_val 3 entry_type increment poll_interval 60.000
 maxrun 20.000
 action 1.0 cli command "show interface FastEthernet 0/0"
 action 1.1 mail server "130.1.3.100" to "noc@ine" from
"router@ine.com" cc "admin@ine.com" subject " R4 Fa0/0 warning
receive_throttle incremented over 3" body "show interface command
output $_cli_result"
```


Task 8.1 Solution

R5:

```
class-map match-all TELNET
 match protocol telnet
!
class-map match-all FTP
 match protocol ftp
!
class-map match-all WWW
 match protocol http
!
policy-map MIGRATION
 class WWW
  bandwidth percent 50
  queue-limit 30
 class FTP
  bandwidth percent 30
 class TELNET
  bandwidth percent 5
 class class-default
  bandwidth percent 15
!
policy-map PARENT
 class class-default
  shape average percent 100
  service-policy MIGRATION
!
interface FastEthernet0/0.52
 service-policy output PARENT
```


Task 8.1 Verification

```
Rack1R5#show policy-map interface fastEthernet 0/0.52
 FastEthernet0/0.52

  Service-policy output: PARENT

    Class-map: class-default (match-any)
      25552 packets, 2950379 bytes
      5 minute offered rate 73000 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 25505/2934493
      shape (average) cir 100000000, bc 400000, be 400000
      target shape rate 100000000

      Service-policy : MIGRATION

        Class-map: WWW (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: protocol http
          Queueing
          queue limit 30 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
          bandwidth 50% (50000 kbps)

        Class-map: FTP (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: protocol ftp
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
          bandwidth 30% (30000 kbps)

        Class-map: TELNET (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: protocol telnet
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
          bandwidth 5% (5000 kbps)
```



```
 Class-map: class-default (match-any)
          25552 packets, 2950379 bytes
          5 minute offered rate 73000 bps, drop rate 0 bps
          Match: any

          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 25506/2934611
          bandwidth 25% (25000 kbps)
Rack1R5#
```

Task 8.2 Solution

R1:

```
interface Serial0/0
 priority-group 1
!
access-list 182 permit udp host 130.1.17.139 any eq 8940
!
priority-list 1 protocol ip high list 182
```

Task 8.2 Verification

```
Rack1R1#show queueing interface serial 0/0
Interface Serial0/0 queueing strategy: priority

Output queue utilization (queue/count)
        high/0 medium/0 normal/0 low/0

Rack1R1#show queueing priority
Current DLCI priority queue configuration:
Current priority queue configuration:

List   Queue  Args
1      high   protocol ip          list 182
```


Lab 16 Solutions

Task 1.1 Solution

SW1, SW2:

```
vtp mode transparent
```

SW1:

```
!
! VLAN 47 was missing from the initial configs
!
vlan 47
!
interface FastEthernet 0/14
 switchport trunk allowed vlan 16,47
!
interface FastEthernet0/19
 switchport trunk allowed vlan none
```

SW2:

```
!
! VLANs 16 and 63 were missing
!
vlan 16,63
!
interface FastEthernet0/14
 switchport trunk allowed vlan 16,47
!
interface FastEthernet0/16
 switchport trunk allowed vlan 63
!
interface FastEthernet0/6
 switchport trunk allowed vlan 16,63
```

SW3:

```
interface FastEthernet0/16
 switchport trunk allowed vlan 63
```

SW4:

```
interface FastEthernet0/13
 switchport trunk allowed vlan none
```


Strategy Tip

Drawing a layer two diagram of the network for this task is recommended

Task 1.1 Verifications

Check VTP operational mode on SW1 and SW2:

```
Rack1SW1#show vtp status
VTP Version                     : running VTP1 (VTP2 capable)
Configuration Revision          : 0
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 11
VTP Operating Mode              : Transparent
VTP Domain Name                 : CCIE
VTP Pruning Mode                : Disabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x7F 0xDC 0x37 0x25 0x67 0x4D 0x62
0x52
Configuration last modified by 150.1.7.7 at 3-1-93 00:08:15

Rack1SW2#show vtp status
VTP Version                     : running VTP1 (VTP2 capable)
Configuration Revision          : 0
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 11
VTP Operating Mode              : Transparent
VTP Domain Name                 : CCIE
VTP Pruning Mode                : Disabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x57 0x3C 0x11 0xC3 0x67 0xD6 0xB2
0xCC
Configuration last modified by 154.1.38.8 at 3-1-93 00:08:35

Rack1SW3#show interfaces trunk | begin forwarding
Port        Vlans in spanning tree forwarding state and not pruned
Fa0/16      63


Rack1SW1#show interfaces trunk | begin forwarding
Port        Vlans in spanning tree forwarding state and not pruned
Fa0/14      16,47
Fa0/19      none

Rack1SW2#show interfaces trunk | begin forwarding
Port        Vlans in spanning tree forwarding state and not pruned
Fa0/6       16,63
Fa0/14      16,47
Fa0/16      63
```



```
Rack1SW4#show interfaces trunk | begin forwarding
Port        Vlans in spanning tree forwarding state and not pruned
Fa0/13      none
```

Task 1.2 Solution

SW1:

```
!
interface FastEthernet0/14
 switchport trunk allowed vlan add 100,200
```

SW2:

```
!
interface FastEthernet0/14
 switchport trunk allowed vlan add 100,200
```

SW3:

```
interface FastEthernet0/13
 switchport access vlan 45
 switchport mode access
 no shutdown
!
interface FastEthernet0/14
 switchport access vlan 45
 switchport mode access
 no shutdown
```

SW4:

```
interface FastEthernet0/16
 switchport access vlan 45
 switchport mode access
 no shutdown
!
interface FastEthernet0/17
 switchport access vlan 45
 switchport mode access
 no shutdown
```

Task 1.2 Verification

```
Rack1R4#ping 154.1.45.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 154.1.45.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
Rack1R4#
```


Task 1.3 Solution

R1:

```
bridge irb
!
interface FastEthernet0/0
 no ip address
 bridge-group 1
!
interface Serial0/0
 encapsulation frame-relay
 frame-relay map bridge 102 broadcast
 bridge-group 1
 no shutdown
!
interface BVI1
 ip address 192.10.1.1 255.255.255.0
!
bridge 1 protocol ieee
bridge 1 route ip
```

R2:

```
bridge irb
!
interface FastEthernet0/0
 no ip address
 bridge-group 1
!
interface Serial0/0
 encapsulation frame-relay
 frame-relay map bridge 201 broadcast
 bridge-group 1
 no shutdown
!
interface BVI1
 ip address 192.10.1.2 255.255.255.0
!
bridge 1 protocol ieee
bridge 1 route ip
```


Task 1.3 Verification

Verify bridging configuration. Check that all bridges agree on common root. There should be no blocking ports, and unstable topology:

```
Rack1R1#show spanning-tree 1

 Bridge group 1 is executing the ieee compatible Spanning Tree protocol
  Bridge Identifier has priority 32768, address 000d.bc1f.8c00
  Configured hello time 2, max age 20, forward delay 15
  We are the root of the spanning tree
  Topology change flag set, detected flag set
  Number of topology changes 1 last change occurred 00:00:43 ago
          from FastEthernet0/0
  Times:  hold 1, topology change 35, notification 2
          hello 2, max age 20, forward delay 15
  Timers: hello 1, topology change 29, notification 0, aging 15

 Port 4 (FastEthernet0/0) of Bridge group 1 is forwarding
    Port path cost 19, Port priority 128, Port Identifier 128.4.
    Designated root has priority 32768, address 000d.bc1f.8c00
    Designated bridge has priority 32768, address 000d.bc1f.8c00
    Designated port id is 128.4, designated path cost 0
    Timers: message age 0, forward delay 0, hold 0
    Number of transitions to forwarding state: 1
    BPDU: sent 28, received 1

 Port 5 (Serial0/0) of Bridge group 1 is forwarding
    Port path cost 647, Port priority 128, Port Identifier 128.5.
    Designated root has priority 32768, address 000d.bc1f.8c00
    Designated bridge has priority 32768, address 000d.bc1f.8c00
    Designated port id is 128.5, designated path cost 0
    Timers: message age 0, forward delay 0, hold 0
    Number of transitions to forwarding state: 1
    BPDU: sent 28, received 2
```



```
Rack1R2#show spanning-tree 1

 Bridge group 1 is executing the ieee compatible Spanning Tree protocol
  Bridge Identifier has priority 32768, address 0011.920e.5300
  Configured hello time 2, max age 20, forward delay 15
  Current root has priority 32768, address 000d.bc1f.8c00
  Root port is 5 (Serial0/0), cost of root path is 647
  Topology change flag not set, detected flag not set
  Number of topology changes 2 last change occurred 00:01:00 ago
          from Serial0/0
  Times:  hold 1, topology change 35, notification 2
          hello 2, max age 20, forward delay 15
  Timers: hello 0, topology change 0, notification 0, aging 300

 Port 4 (FastEthernet0/0) of Bridge group 1 is forwarding
    Port path cost 19, Port priority 128, Port Identifier 128.4.
    Designated root has priority 32768, address 000d.bc1f.8c00
    Designated bridge has priority 32768, address 0011.920e.5300
    Designated port id is 128.4, designated path cost 647
    Timers: message age 0, forward delay 0, hold 0
    Number of transitions to forwarding state: 1
    BPDU: sent 45, received 1

 Port 5 (Serial0/0) of Bridge group 1 is forwarding
    Port path cost 647, Port priority 128, Port Identifier 128.5.
    Designated root has priority 32768, address 000d.bc1f.8c00
    Designated bridge has priority 32768, address 000d.bc1f.8c00
    Designated port id is 128.5, designated path cost 0
    Timers: message age 1, forward delay 0, hold 0
    Number of transitions to forwarding state: 1
    BPDU: sent 2, received 46

Rack1SW1#show spanning-tree vlan 16

VLAN0016
  Spanning tree enabled protocol ieee
  Root ID     Priority    32768
              Address     000d.bc1f.8c00
              Cost        19
              Port        3 (FastEthernet0/1)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32784  (priority 32768 sys-id-ext 16)
              Address     001b.d490.7a80
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------
Fa0/1               Root FWD 19        128.3    P2p
Fa0/14              Desg FWD 19        128.16   P2p
```



```
Rack1SW2#show spanning-tree vlan 22

VLAN0022
  Spanning tree enabled protocol ieee
  Root ID     Priority    32768
              Address     000d.bc1f.8c00
              Cost        666
              Port        4 (FastEthernet0/2)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32790  (priority 32768 sys-id-ext 22)
              Address     001b.d4df.ec80
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- ----------------------
Fa0/2               Root FWD 19        128.4    P2p
Fa0/24              Desg FWD 100       128.26   Shr
```

Check briding talbes:

```
Rack1R1#show bridge 1 verbose

Total of 300 station blocks, 295 free
Codes: P - permanent, S - self

BG Hash      Address     Action  Interface   VC  Age RX count  TX count
 1 53/0   0011.920e.5300 forward Serial0/0  102    0      136        32
 1 68/0   001b.d4df.ec84 forward Serial0/0  102    0       35         0
 1 6E/0   0000.0c34.1a74 forward Serial0/0  102    0      347        28
 1 DB/0   0019.56d4.f922 forward FastEthernet0/0 - 0      211        42
 1 F9/0   001b.d490.7a83 forward FastEthernet0/0 - 0       36         0

Flood ports (BG 1)           RX count    TX count
FastEthernet0/0                    52         134
Serial0/0                         134          51

Rack1R2#show bridge 1 verbose

Total of 300 station blocks, 295 free
Codes: P - permanent, S - self

BG Hash      Address      Action  Interface   VC  Age RX count TX count
 1 68/0   001b.d4df.ec84 forward  FastEthernet0/0 - 0      41         0
 1 6E/0   0000.0c34.1a74 forward  FastEthernet0/0 - 0      414       48
 1 8C/0   000d.bc1f.8c00 forward  Serial0/0  201    0      179       30
 1 DB/0   0019.56d4.f922 forward  Serial0/0  201    0      194       15
 1 F9/0   001b.d490.7a83 forward  Serial0/0  201    0       42        0

Flood ports (BG 1)           RX count    TX count
FastEthernet0/0                   137          78
Serial0/0                          78         135
```


Finally, test IP connectivity:

```
Rack1R6#ping 192.10.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.10.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Rack1R6#ping 192.10.1.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.10.1.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/8 ms

Rack1R6#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 80 percent (4/5), round-trip min/avg/max = 4/7/8 ms
```


Task 1.4 Solution

SW1:

```
interface FastEthernet0/16
 l2protocol-tunnel drop-threshold stp 100
!
interface FastEthernet0/17
 l2protocol-tunnel drop-threshold stp 100
```

SW2:

```
interface FastEthernet0/19
 l2protocol-tunnel drop-threshold stp 100
!
interface FastEthernet0/20
 l2protocol-tunnel drop-threshold stp 100
```

Task 1.4 Verification

```
Rack1SW1#show l2protocol-tunnel interface fastEthernet 0/17
COS for Encapsulated Packets: 5

Port         Protocol  Shutdown   Drop     Encapsulation  Decapsulation  Drop
                       Threshold Threshold  Counter        Counter        Counter
------------------- -------- --------- --------- ------------- ------------- --
Fa0/17          cdp        ----      ----          23           18             0
                stp         ----       100        651           1             0
                ---         ----      ----          ----       ----          ----
                ---         ----      ----          ----       ----          ----
                ---         ----      ----          ----       ----          ----
                ---         ----      ----          ----       ----          ----
.
```

Task 1.5 Solution

R4:

```
interface lo0
 ip ospf network point-to-point
!
mpls ldp explicit-null
!
interface fastEthernet 0/1
 mpls ip
```

R5:

```
interface Loopback0
 ip ospf network point-to-point
!
mpls ldp explicit-null
!
int fastEthernet 0/1
 mpls ip
```

Task 1.4 Verification

```
Rack1R4#show mpls ldp neighbor
    Peer LDP Ident: 150.1.5.5:0; Local LDP Ident 150.1.4.4:0
        TCP connection: 150.1.5.5.58833 - 150.1.4.4.646
        State: Oper; Msgs sent/rcvd: 44/44; Downstream
        Up time: 00:13:06
          FastEthernet0/1, Src IP addr: 154.1.45.5
        LDP discovery sources:
        Addresses bound to peer LDP Ident:
          154.1.5.5       154.1.45.5      154.1.0.5       150.1.5.5

Rack1R4#show mpls forwarding-table
Local  Outgoing      Prefix          Bytes Label  Outgoing   Next Hop
Label  Label or VC   or Tunnel Id    Switched     interface
17     No Label      150.1.7.7/32      0            Fa0/0   154.1.47.7
18     explicit-n    154.1.5.0/24      0            Fa0/1   154.1.45.5
19     explicit-n    150.1.5.0/24      0            Fa0/1   154.1.45.5
```



```
Rack1R5#show mpls ldp neighbor
    Peer LDP Ident: 150.1.4.4:0; Local LDP Ident 150.1.5.5:0
        TCP connection: 150.1.4.4.646 - 150.1.5.5.58833
        State: Oper; Msgs sent/rcvd: 45/45; Downstream
        Up time: 00:14:27
        LDP discovery sources:
          FastEthernet0/1, Src IP addr: 154.1.45.4
        Addresses bound to peer LDP Ident:
          154.1.47.4      154.1.45.4      154.1.0.4       150.1.4.4

Rack1R5#show mpls forwarding-table
Local  Outgoing      Prefix          Bytes Label   Outgoing    Next Hop
Label  Label or VC   or Tunnel Id    Switched      interface
16     explicit-n    150.1.4.0/24      0             Fa0/1    154.1.45.4
17     explicit-n    154.1.47.0/24     0             Fa0/1    154.1.45.4
18     17            150.1.7.7/32      0             Fa0/1    154.1.45.4
```


Task 2.1 Solution

R3:

```
!
router ospf 1
 network 150.1.3.3 0.0.0.0 area 3457
 network 154.1.0.3 0.0.0.0 area 3457
 neighbor 154.1.0.5 cost 195
 neighbor 154.1.0.4 cost 97
!
interface Serial1/0
 ip ospf network point-to-multipoint non-broadcast
```

R4:

```
!
interface Serial0/0/0
 ip ospf network point-to-multipoint non-broadcast
!
router ospf 1
 router-id 150.1.4.4
 network 150.1.4.4 0.0.0.0 area 3457
 network 154.1.0.4 0.0.0.0 area 3457
 network 154.1.45.4 0.0.0.0 area 3457
 network 154.1.47.4 0.0.0.0 area 3457
```

R5:

```
!
interface Serial0/0/0
 ip ospf network point-to-multipoint non-broadcast
!
router ospf 1
 router-id 150.1.5.5
 network 150.1.5.5 0.0.0.0 area 3457
 network 154.1.0.5 0.0.0.0 area 3457
 network 154.1.45.5 0.0.0.0 area 3457
```

SW1:

```
!
router ospf 1
 router-id 150.1.7.7
 network 150.1.7.7 0.0.0.0 area 3457
 network 154.1.47.7 0.0.0.0 area 3457
```


Task 2.1 Verification

Verify OSPF neighbors at R3 and R4:

```
Rack1R3#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.2.2         0   FULL/  -        00:00:31    154.1.23.2       Serial1/3
150.1.1.1         0   FULL/  -        00:00:31    154.1.13.1       Serial1/2
150.1.8.8         1   FULL/DR         00:00:33    154.1.38.8 FastEthernet0/0
150.1.5.5         0   FULL/  -        00:01:37    154.1.0.5        Serial1/0
150.1.4.4         0   FULL/  -        00:01:42    154.1.0.4        Serial1/0

Rack1R4#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.7.7         1   FULL/BDR        00:00:34    154.1.47.7 FastEthernet0/0
150.1.5.5         1   FULL/DR         00:00:33    154.1.45.5 FastEthernet0/1
150.1.3.3         0   FULL/  -        00:01:59    154.1.0.3      Serial0/0/0
```

Confirm that R3 takes in account different PVCs CIR. To do that, temporarily shutdown interface FastEthernet0/1 at R4.

```
Rack1R3#show ip route ospf
     154.1.0.0/16 is variably subnetted, 12 subnets, 2 masks
O       154.1.0.5/32 [110/195] via 154.1.0.5, 00:00:01, Serial1/0
O       154.1.0.4/32 [110/97] via 154.1.0.4, 00:00:01, Serial1/0
O       154.1.47.0/24 [110/98] via 154.1.0.4, 00:00:01, Serial1/0
O       154.1.45.0/24 [110/196] via 154.1.0.5, 00:00:01, Serial1/0
O    192.10.1.0/24 [110/845] via 154.1.23.2, 00:01:24, Serial1/3
                   [110/845] via 154.1.13.1, 00:01:24, Serial1/2
     150.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
O       150.1.5.0/24 [110/196] via 154.1.0.5, 00:00:01, Serial1/0
O       150.1.4.0/24 [110/98] via 154.1.0.4, 00:00:01, Serial1/0
O       150.1.8.8/32 [110/2] via 154.1.38.8, 00:01:24, FastEthernet0/0
O       150.1.7.7/32 [110/99] via 154.1.0.4, 00:00:01, Serial1/0
O       150.1.6.6/32 [110/846] via 154.1.23.2, 00:01:24, Serial1/3
                     [110/846] via 154.1.13.1, 00:01:24, Serial1/2
O       150.1.2.2/32 [110/782] via 154.1.23.2, 00:01:24, Serial1/3
O       150.1.1.1/32 [110/782] via 154.1.13.1, 00:01:25, Serial1/2
```



```
Rack1R3#show ip ospf neighbor detail serial 1/0
 Neighbor 150.1.5.5, interface address 154.1.0.5
    In the area 3457 via interface Serial1/0
    Neighbor priority is 0 (configured 0), State is FULL, 13 state
changes, Cost is 195
    DR is 0.0.0.0 BDR is 0.0.0.0
    Options is 0x52
    LLS Options is 0x1 (LR)
    Dead timer due in 00:01:39
    Neighbor is up for 00:07:56
   Index 1/4, retransmission queue length 0, number of retransmission 1
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 0, maximum is 1
    Last retransmission scan time is 0 msec, maximum is 0 msec
 Neighbor 150.1.4.4, interface address 154.1.0.4
    In the area 3457 via interface Serial1/0
    Neighbor priority is 0 (configured 0), State is FULL, 13 state
changes, Cost is 97
    DR is 0.0.0.0 BDR is 0.0.0.0
    Options is 0x52
    LLS Options is 0x1 (LR)
    Dead timer due in 00:01:39
    Neighbor is up for 00:07:56
   Index 2/5, retransmission queue length 0, number of retransmission 2
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 1, maximum is 1
    Last retransmission scan time is 0 msec, maximum is 0 msec
```


Task 2.2 Solution

R1:

```
router ospf 1
 network 192.10.1.1 0.0.0.0 area 51
 network 150.1.1.1 0.0.0.0 area 51
 area 51 range 150.1.0.0 255.255.252.0
```

R2:

```
router ospf 1
 network 192.10.1.2 0.0.0.0 area 51
 network 150.1.2.2 0.0.0.0 area 51
 area 51 range 150.1.0.0 255.255.252.0
```

R6:

```
router ospf 1
 router-id 150.1.6.6
 network 192.10.1.6 0.0.0.0 area 51
 network 150.1.6.6 0.0.0.0 area 51
```

Task 2.2 Verification

Verify OSPF neighbors:

```
Rack1R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.3.3         0   FULL/  -        00:00:35    154.1.13.3      Serial0/1
150.1.2.2         1   FULL/DROTHER    00:00:33    192.10.1.2          BVI1
150.1.6.6         1   FULL/DROTHER    00:00:30    192.10.1.6          BVI1
192.10.1.254      1   FULL/DR         00:00:34    192.10.1.254        BVI1
```

Check OSPF routes:

```
Rack1R1#show ip route ospf
     51.0.0.0/32 is subnetted, 1 subnets
O E2    51.51.51.51 [110/20] via 192.10.1.254, 00:02:56, BVI1
     154.1.0.0/16 is variably subnetted, 9 subnets, 2 masks
O       154.1.23.0/24 [110/845] via 154.1.13.3, 00:03:36, Serial0/1
O       154.1.3.0/24 [110/74] via 154.1.13.3, 00:03:36, Serial0/1
O IA    154.1.0.3/32 [110/64] via 154.1.13.3, 00:02:56, Serial0/1
O IA    154.1.0.5/32 [110/171] via 154.1.13.3, 00:02:56, Serial0/1
O IA    154.1.0.4/32 [110/161] via 154.1.13.3, 00:02:56, Serial0/1
O IA    154.1.47.0/24 [110/171] via 154.1.13.3, 00:02:56, Serial0/1
O IA    154.1.45.0/24 [110/171] via 154.1.13.3, 00:02:56, Serial0/1
     150.1.0.0/16 is variably subnetted, 8 subnets, 3 masks
O IA    150.1.7.7/32 [110/172] via 154.1.13.3, 00:02:56, Serial0/1
O       150.1.6.6/32 [110/65] via 192.10.1.6, 00:02:56, BVI1
O IA    150.1.5.5/32 [110/172] via 154.1.13.3, 00:02:56, Serial0/1
O IA    150.1.4.4/32 [110/162] via 154.1.13.3, 00:02:56, Serial0/1
O IA    150.1.3.3/32 [110/65] via 154.1.13.3, 00:02:56, Serial0/1
O       150.1.2.2/32 [110/65] via 192.10.1.2, 00:02:57, BVI1
O       150.1.0.0/22 is a summary, 00:02:57, Null0
```


Verify summary OSPF prefix for Area 51:

```
Rack1R5#show ip route ospf
     51.0.0.0/32 is subnetted, 1 subnets
O E2    51.51.51.51 [110/20] via 154.1.0.3, 00:02:20, Serial0/0/0
     154.1.0.0/16 is variably subnetted, 10 subnets, 2 masks
O IA    154.1.23.0/24 [110/845] via 154.1.0.3, 00:12:43, Serial0/0/0
O IA    154.1.13.0/24 [110/845] via 154.1.0.3, 00:12:43, Serial0/0/0
O IA    154.1.3.0/24 [110/65] via 154.1.0.3, 00:12:43, Serial0/0/0
O       154.1.0.3/32 [110/64] via 154.1.0.3, 00:12:43, Serial0/0/0
O       154.1.0.4/32 [110/1] via 154.1.45.4, 00:10:07, FastEthernet0/1
O       154.1.47.0/24 [110/2] via 154.1.45.4, 00:10:07, FastEthernet0/1
O IA    154.1.38.0/24 [110/65] via 154.1.0.3, 00:12:43, Serial0/0/0
O IA 192.10.1.0/24 [110/909] via 154.1.0.3, 00:02:20, Serial0/0/0
     150.1.0.0/16 is variably subnetted, 7 subnets, 3 masks
O       150.1.4.0/24 [110/2] via 154.1.45.4, 00:10:07, FastEthernet0/1
O IA    150.1.8.8/32 [110/66] via 154.1.0.3, 00:12:43, Serial0/0/0
O       150.1.7.7/32 [110/3] via 154.1.45.4, 00:10:07, FastEthernet0/1
O IA    150.1.6.6/32 [110/910] via 154.1.0.3, 00:02:01, Serial0/0/0
O       150.1.3.3/32 [110/65] via 154.1.0.3, 00:12:43, Serial0/0/0
O IA    150.1.0.0/22 [110/846] via 154.1.0.3, 00:02:35, Serial0/0/0
```

Task 2.3 Solution

R3:

```
router ospf 1
 network 154.1.38.3 0.0.0.0 area 38
 area 3457 filter-list prefix AREA_38 in
!
ip prefix-list AREA_38 seq 5 deny 150.1.8.8/32
ip prefix-list AREA_38 seq 10 deny 154.1.38.0/24
ip prefix-list AREA_38 seq 15 permit 0.0.0.0/0 le 32
```

SW2:

```
ip routing
!
router ospf 1
 router-id 150.1.8.8
 network 154.1.38.8 0.0.0.0 area 38
 network 150.1.8.8 0.0.0.0 area 38
```


Task 2.3 Verification

Verify OSPF neighbors. Note that OSPF was pre-configured between R3 and SW2 in the initial configurations using OSPF area 0:

```
Rack1SW2#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.3.3         1   FULL/BDR        00:00:34    154.1.38.3      FastEthernet0/15
```

Confirm prefix filtering:

```
Rack1R4#sho ip route 150.1.8.8
% Subnet not in table

Rack1R4#show ip route 154.1.38.0
% Subnet not in table
```

Task 2.4 Solution

R3:

```
router ospf 1
 timers throttle spf 4000 10000 90000
```

Task 2.4 Verification

Verify SPF throttle timers:

```
Rack1R3#show ip ospf
 Routing Process "ospf 1" with ID 150.1.3.3
 Supports only single TOS(TOS0) routes
 Supports opaque LSA
 Supports Link-local Signaling (LLS)
 Supports area transit capability
 It is an area border router
 Initial SPF schedule delay 4000 msecs
 Minimum hold time between two consecutive SPFs 10000 msecs
 Maximum wait time between two consecutive SPFs 90000 msecs
 Incremental-SPF disabled
<output omitted>
```


Task 2.5 Solution

R6:

```
router bgp 100
 neighbor 54.1.8.254 route-map BB1_IN in
 neighbor 204.12.1.254 route-map BB3_IN in
!
ip bgp-community new-format
!
ip as-path access-list 1 permit ^54$
ip as-path access-list 2 permit _60$
!
route-map BB1_IN permit 10
 match as-path 1
 set community 54:1
!
route-map BB1_IN permit 20
 match as-path 2
 set community 60:1
!
route-map BB3_IN permit 10
 match as-path 1
 set community 54:3
!
route-map BB3_IN permit 20
 match as-path 2
 set community 60:3
```

☞ Quick Note
Only for clarity of display, does not affect community values or processing.

Task 2.5 Verification

Verify community tagging:

```
Rack1R6#show ip bgp 119.0.0.0
BGP routing table entry for 119.0.0.0/8, version 30
Paths: (2 available, best #1, table Default-IP-Routing-Table)
Flag: 0x8C0
  Advertised to update-groups:
     1
  54
    54.1.8.254 from 54.1.8.254 (212.18.3.1)
      Origin IGP, metric 0, localpref 100, valid, external, best
      Community: 54:1
  54
    204.12.1.254 from 204.12.1.254 (31.3.0.1)
      Origin IGP, localpref 100, valid, external
      Community: 54:3
```



```
Rack1R6#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 31
Paths: (2 available, best #1, table Default-IP-Routing-Table)
Flag: 0x8C0
  Advertised to update-groups:
     1
  54 50 60
    54.1.8.254 from 54.1.8.254 (212.18.3.1)
      Origin IGP, metric 0, localpref 100, valid, external, best
      Community: 60:1
  54 50 60
    204.12.1.254 from 204.12.1.254 (31.3.0.1)
      Origin IGP, localpref 100, valid, external
   Community: 60:3
```

Task 2.6 Solution

R6:

```
route-map BB1_IN permit 10
 set weight 1
!
route-map BB3_IN permit 20
 set weight 1
```


Task 2.6 Verification

Verify best-paths:

```
Rack1R6#show ip bgp
BGP table version is 46, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   54.1.8.254                             1 54 i
*                   204.12.1.254             0             0 54 i
*> 28.119.17.0/24   54.1.8.254                             1 54 i
*                   204.12.1.254             0             0 54 i
*> 54.1.8.0/24      0.0.0.0                  0         32768 i
*  112.0.0.0        54.1.8.254               0             0 54 50 60 i
*>                  204.12.1.254                           1 54 50 60 i
*  113.0.0.0        54.1.8.254               0             0 54 50 60 i
*>                  204.12.1.254                           1 54 50 60 i
*> 114.0.0.0        54.1.8.254               0             1 54 i
*                   204.12.1.254                           0 54 i
*> 115.0.0.0        54.1.8.254               0             1 54 i
*                   204.12.1.254                           0 54 i
*> 116.0.0.0        54.1.8.254               0             1 54 i
*                   204.12.1.254                           0 54 i
*> 117.0.0.0        54.1.8.254               0             1 54 i
*                   204.12.1.254                           0 54 i
*> 118.0.0.0        54.1.8.254               0             1 54 i
*                   204.12.1.254                           0 54 i
*> 119.0.0.0        54.1.8.254               0             1 54 i
*                   204.12.1.254                           0 54 i
*> 204.12.1.0       0.0.0.0                  0         32768 i
*> 205.90.31.0      192.10.1.254                           0 200 254 ?
*> 220.20.3.0       192.10.1.254                           0 200 254 ?
*> 222.22.2.0       192.10.1.254                           0 200 254 ?
```

```
Rack1R6#show ip bgp 28.119.16.0
BGP routing table entry for 28.119.16.0/24, version 43
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  54
    54.1.8.254 from 54.1.8.254 (212.18.3.1)
      Origin IGP, localpref 100, weight 1, valid, external, best
      Community: 54:1
  54
    204.12.1.254 from 204.12.1.254 (31.3.0.1)
      Origin IGP, metric 0, localpref 100, valid, external
      Community: 54:3
```



```
Rack1R6#traceroute 28.119.16.1

Type escape sequence to abort.
Tracing the route to 28.119.16.1

  1 54.1.8.254 20 msec 16 msec 20 msec
  2 172.16.4.3 20 msec *  108 msec


Rack1R6#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 41
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  54 50 60
    54.1.8.254 from 54.1.8.254 (212.18.3.1)
      Origin IGP, metric 0, localpref 100, valid, external
      Community: 60:1
  54 50 60
    204.12.1.254 from 204.12.1.254 (31.3.0.1)
      Origin IGP, localpref 100, weight 1, valid, external, best
      Community: 60:3

Rack1R6#traceroute 112.0.0.1

Type escape sequence to abort.
Tracing the route to 112.0.0.1

  1 204.12.1.254 4 msec 4 msec 4 msec
  2 172.16.4.1 24 msec *  16 msec
```

Task 2.7 Solution

R1:

```
router bgp 200
 neighbor 154.1.13.3 route-map R3_OUT out
!
ip bgp-community new-format
ip community-list 1 permit 60:3
!
route-map R3_OUT permit 10
 match community 1
 set as-path prepend 200
!
route-map R3_OUT permit 1000
```


R2:

```
router bgp 200
 neighbor 154.1.23.3 route-map R3_OUT out
!
ip bgp-community new-format
ip community-list 1 permit 54:1
!
route-map R3_OUT permit 10
 match community 1
 set as-path prepend 200
!
route-map R3_OUT permit 1000
```

R6 should be configured as well to allows AS54 devices to route back to the pod networks. This is not explicitly required but follows from the reachibility requirements.

R6:

```
!
router bgp 100
 redistribute ospf 1 route-map IGP_TO_BGP
 neighbor 192.10.1.1 route-map TO_R1 out
!
ip prefix-list INTERNAL_NETWORK seq 5 permit 154.1.0.0/16 le 32
ip prefix-list INTERNAL_NETWORK seq 10 permit 150.1.0.0/16 le 32

!
route-map IGP_TO_BGP permit 10
 match ip address prefix-list INTERNAL_NETWORK
!
route-map TO_R1 deny 10
 match ip address prefix-list INTERNAL_NETWORK
!
route-map TO_R1 permit 1000
```

R3:

```
!
ip bgp-community new-format
```


Task 2.7 Verification

Verify BGP best-paths at R3:

```
Rack1R3#show ip bgp community
BGP table version is 53, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop    Metric LocPrf Weight Path
*  28.119.16.0/24   154.1.23.2                    0  200 200 100 54 i
*>                  154.1.13.1                    0  200 100 54 i
*  28.119.17.0/24   154.1.23.2                    0  200 200 100 54 i
*>                  154.1.13.1                    0  200 100 54 i
*> 112.0.0.0        154.1.23.2                    0  200 100 54 50 60 i
*                   154.1.13.1                    0  200 200 100 54 50
60 i
*> 113.0.0.0        154.1.23.2                    0  200 100 54 50 60 i
*                   154.1.13.1                    0  200 200 100 54 50
60 i
*  114.0.0.0        154.1.23.2                    0  200 200 100 54 i
*>                  154.1.13.1                    0  200 100 54 i
*  115.0.0.0        154.1.23.2                    0  200 200 100 54 i
*>                  154.1.13.1                    0  200 100 54 i
*  116.0.0.0        154.1.23.2                    0  200 200 100 54 i
*>                  154.1.13.1                    0  200 100 54 i
*  117.0.0.0        154.1.23.2                    0  200 200 100 54 i
*>                  154.1.13.1                    0  200 100 54 i
*  118.0.0.0        154.1.23.2                    0  200 200 100 54 i
*>                  154.1.13.1                    0  200 100 54 i
*  119.0.0.0        154.1.23.2                    0  200 200 100 54 i
*>                  154.1.13.1                    0  200 100 54 i
```



```
Rack1R3#show ip bgp 28.119.16.0
BGP routing table entry for 28.119.16.0/24, version 39
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2
  200 200 100 54
    154.1.23.2 from 154.1.23.2 (150.1.2.2)
      Origin IGP, localpref 100, valid, external
      Community: 54:1
  200 100 54
    154.1.13.1 from 154.1.13.1 (150.1.1.1)
      Origin IGP, localpref 100, valid, external, best
      Community: 54:1

Rack1R3#show ip bgp 112.0.0.0
BGP routing table entry for 112.0.0.0/8, version 37
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1           2
  200 100 54 50 60
    154.1.23.2 from 154.1.23.2 (150.1.2.2)
      Origin IGP, localpref 100, valid, external, best
      Community: 60:3
  200 200 100 54 50 60
    154.1.13.1 from 154.1.13.1 (150.1.1.1)
      Origin IGP, localpref 100, valid, external
      Community: 60:3

Rack1R3#ping 112.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 112.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 124/124/125
ms

Rack1R3#ping 28.119.16.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 28.119.16.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 60/67/84 ms
```


Verify that R6 does not advertise to its R1 peering OSPF learned prefixes:

```
Rack1R6#show ip bgp neighbors 192.10.1.1 policy detail
 Neighbor: 192.10.1.1, Address-Family: IPv4 Unicast
 Locally configured policies:
  route-map TO R1 out
  send-community

 Neighbor: 192.10.1.1, Address-Family: IPv4 Unicast <detail>
 Locally configured policies:
  route-map TO_R1 out
route-map TO R1, deny, sequence 10
  Match clauses:
    ip address prefix-lists: INTERNAL_NETWORK
ip prefix-list INTERNAL_NETWORK: 2 entries
   seq 5 permit 154.1.0.0/16 le 32
   seq 10 permit 150.1.0.0/16 le 32

  Set clauses:
  Policy routing matches: 0 packets, 0 bytes
route-map TO_R1, permit, sequence 1000
  Match clauses:
  Set clauses:
  Policy routing matches: 0 packets, 0 bytes

Rack1R6#show ip bgp nei 192.10.1.1 advertised-routes
BGP table version is 68, local router ID is 150.1.6.6
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   54.1.8.254                             1 54 i
*> 28.119.17.0/24   54.1.8.254                             1 54 i
*> 54.1.8.0/24      0.0.0.0                  0         32768 i
*> 112.0.0.0        204.12.1.254                           1 54 50 60 i
*> 113.0.0.0        204.12.1.254                           1 54 50 60 i
*> 114.0.0.0        54.1.8.254               0             1 54 i
*> 115.0.0.0        54.1.8.254               0             1 54 i
*> 116.0.0.0        54.1.8.254               0             1 54 i
*> 117.0.0.0        54.1.8.254               0             1 54 i
*> 118.0.0.0        54.1.8.254               0             1 54 i
*> 119.0.0.0        54.1.8.254               0             1 54 i
*> 204.12.1.0       0.0.0.0                  0         32768 i
```


Task 2.8 Solution

R1:

```
router bgp 200
 neighbor 192.10.1.2 next-hop-self
```

R3:

```
router bgp 300
 neighbor 154.1.23.2 route-map TO_R2 out
!
ip prefix-list VLAN5 seq 5 permit 154.1.5.0/24
!
route-map TO_R2 permit 10
 match ip address prefix-list VLAN5
 set metric 100
!
route-map TO_R2 permit 1000
```

R5:

```
router bgp 400
 network 154.1.5.0 mask 255.255.255.0
```


Task 2.8 Verification

Verify best-path for VLAN5:

```
Rack1R2#show ip bgp 154.1.5.0
BGP routing table entry for 154.1.5.0/24, version 41
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to non peer-group peers:
  154.1.23.3 192.10.1.254
  300 400
    154.1.23.3 from 154.1.23.3 (150.1.3.3)
      Origin IGP, metric 100, localpref 100, valid, external
  300 400
    192.10.1.1 from 192.10.1.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal, best

Rack1R2#traceroute 154.1.5.5

Type escape sequence to abort.
Tracing the route to 154.1.5.5

  1 192.10.1.1 4 msec 8 msec 4 msec
  2 154.1.13.3 24 msec 16 msec 16 msec
  3 154.1.0.4 48 msec 48 msec 44 msec
  4 154.1.45.5 52 msec *  44 msec

Rack1R6#traceroute 154.1.5.5

Type escape sequence to abort.
Tracing the route to 154.1.5.5

  1 192.10.1.1 0 msec 4 msec 0 msec
  2 154.1.13.3 16 msec 16 msec 16 msec
  3 154.1.0.4 44 msec 44 msec 44 msec
  4 154.1.45.5 44 msec *  44 msec
```

Task 3.1 Solution

R6:

```
interface Tunnel56
 ipv6 address 2001:CC1E:1:56::6/64
 tunnel source Loopback0
 tunnel destination 150.1.5.5
 tunnel mode ipv6ip
```

R5:

```
interface Tunnel56
 ipv6 address 2001:CC1E:1:56::5/64
 tunnel source Loopback0
 tunnel destination 150.1.6.6
 tunnel mode ipv6ip
```


Task 3.1 Verification

Verify tunnel configuration:

```
Rack1R5#show interfaces tunnel 56
Tunnel56 is up, line protocol is up
  Hardware is Tunnel
  MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 150.1.5.5 (Loopback0), destination 150.1.6.6
  Tunnel protocol/transport IPv6/IP
<output omitted>

Rack1R5#ping 2001:CC1E:1:56::6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:56::6, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5),round-trip min/avg/max = 92/99/104 ms
```

Task 3.2 Solution

R3:

```
interface Serial1/3
 ip policy route-map POLICY_ROUTE_IPv6IP
!
interface Serial1/2
 ip policy route-map POLICY_ROUTE_IPv6IP
!
ip access-list extended IPv6IP
 permit 41 any any
!
route-map POLICY_ROUTE_IPv6IP permit 10
 match ip address IPv6IP
 set ip next-hop 154.1.0.5
```


Task 3.2 Verification

Verify the path traffic takes before configuration is applied;notice that traffic is load-balanced between R1 and R2:

```
Rack1R6#show ip route 150.1.5.5
Routing entry for 150.1.5.0/24
  Known via "ospf 1", distance 110, metric 164, type inter area
  Redistributing via bgp 100
  Advertised by bgp 100 route-map IGP_TO_BGP
  Last update from 192.10.1.1 on FastEthernet0/0.16, 01:34:51 ago
  Routing Descriptor Blocks:
  * 192.10.1.2, from 150.1.2.2, 01:34:51 ago, via FastEthernet0/0.16
      Route metric is 164, traffic share count is 1
    192.10.1.1, from 150.1.1.1, 01:34:51 ago, via FastEthernet0/0.16
      Route metric is 164, traffic share count is 1

Rack1R6#traceroute 150.1.5.5 source loopback 0

Type escape sequence to abort.
Tracing the route to 150.1.5.5

  1 192.10.1.1 4 msec
    192.10.1.2 40 msec
    192.10.1.1 4 msec
  2 154.1.23.3 52 msec
    154.1.13.3 16 msec
    154.1.23.3 56 msec
  3 154.1.0.4 44 msec 60 msec 44 msec
  4 154.1.45.5 80 msec *  80 msec
```

Note that IPv4 traffic is still routed normally, only protocol 41 is policy routed:

```
Rack1R3#debug ip policy

Rack1R6#ping 2001:CC1E:1:56::5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:56::5, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 100/102/104
ms

Rack1R3#show ip access-lists
Extended IP access list IPv6IP
    10 permit 41 any any (15 matches)
```



```
Rack1R3#
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5, len 120, FIB policy match
CEF-IP-POLICY: fib for address 154.1.0.5 is with flag 0
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5, len 120, policy match
IP: route map POLICY_ROUTE_IPv6IP, item 10, permit
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5 (Serial1/0), len 120, policy
routed
IP: Serial1/2 to Serial1/0 154.1.0.5
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5, len 120, FIB policy match
CEF-IP-POLICY: fib for address 154.1.0.5 is with flag 0
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5, len 120, policy match
IP: route map POLICY_ROUTE_IPv6IP, item 10, permit
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5 (Serial1/0), len 120, policy
routed
IP: Serial1/2 to Serial1/0 154.1.0.5
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5, len 120, FIB policy match
CEF-IP-POLICY: fib for address 154.1.0.5 is with flag 0
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5, len 120, policy match
IP: route map POLICY_ROUTE_IPv6IP, item 10, permit
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5 (Serial1/0), len 120, policy
routed
IP: Serial1/2 to Serial1/0 154.1.0.5
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5, len 120, FIB policy match
CEF-IP-POLICY: fib for address 154.1.0.5 is with flag 0
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5, len 120, policy match
IP: route map POLICY_ROUTE_IPv6IP, item 10, permit
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5 (Serial1/0), len 120, policy
routed
IP: Serial1/2 to Serial1/0 154.1.0.5
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5, len 120, FIB policy match
CEF-IP-POLICY: fib for address 154.1.0.5 is with flag 0
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5, len 120, policy match
IP: route map POLICY_ROUTE_IPv6IP, item 10, permit
IP: s=150.1.6.6 (Serial1/2), d=150.1.5.5 (Serial1/0), len 120, policy
routed

Rack1R3#undebug all
```

Task 3.3 Solution

R5:

```
!
ipv6 router eigrp 56
 router-id 5.5.5.5
 no shutdown
!
interface loopback0
 ipv6 eigrp 56
!
interface fastEthernet 0/0
 ipv6 eigrp 56
!
interface Tunnel 56
 ipv6 eigrp 56
```

R6:

```
!
ipv6 router eigrp 56
 router-id 6.6.6.6
 no shutdown
!
interface Loopback0
 ipv6 eigrp 56
!
int fastEthernet 0/0
 ipv6 eigrp 56
!
interface tunnel56
 ipv6 eigrp 56
 ipv6 summary-address eigrp 56 2001:200::/22
!
interface loopback 100
 ipv6 eigrp 56
!
interface loopback 101
 ipv6 eigrp 56
!
interface loopback 103
 ipv6 eigrp 56
```


Task 3.3 Verification

```
Rack1R5#show ipv6 eigrp neighbors
IPv6-EIGRP neighbors for process 56
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   Link-local address:   Tu56              12 00:18:09  156  5000  0  11
    FE80::9601:606

Rack1R5#show ipv6 route eigrp
IPv6 Routing Table - Default - 8 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
D   2001::/22 [90/26882560]
     via FE80::9601:606, Tunnel56
D   2001:CC1E:1::6/128 [90/27008000]
     via FE80::9601:606, Tunnel56

Rack1R6#show ipv6 route eigrp
IPv6 Routing Table - Default - 15 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
D   2001::/22 [5/28160]
     via Null0, directly connected
D   2001:CC1E:1::5/128 [90/27008000]
     via FE80::9601:505, Tunnel56
D   2001:CC1E:1:5::/64 [90/26882560]
     via FE80::9601:505, Tunnel56

Rack1R5#ping 2001:220:20:3::1 source lo0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:220:20:3::1, timeout is 2
seconds:
Packet sent with a source address of 2001:CC1E:1::5
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 176/176/180
ms
```



```
Rack1R5#ping 2001:222:22:2::1 source lo0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:222:22:2::1, timeout is 2
seconds:
Packet sent with a source address of 2001:CC1E:1::5
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 176/176/176
ms

Rack1R5#ping 2001:205:90:31::1 source lo0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:205:90:31::1, timeout is 2
seconds:
Packet sent with a source address of 2001:CC1E:1::5
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 176/177/180
ms

Rack1R5#ping 2001:192:1:0:21D:A1FF:FE5F:FA02 source loopback 0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:192:1:0:21D:A1FF:FE5F:FA02,
timeout is 2 seconds:
Packet sent with a source address of 2001:CC1E:1::5
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 104/104/104
ms
```


Task 5.1 Solution

R1:

```
ip pim autorp listener
!
interface BVI1
 ip pim sparse-mode
```

R3:

```
!
interface Loopback0
 ip pim sparse-mode
!
ip pim send-rp-discovery Loopback0 scope 16
ip pim rp-announce-filter rp-list R4 group-list GROUP_224
ip pim rp-announce-filter rp-list R5 group-list GROUP_224
ip pim rp-announce-filter rp-list R6 group-list GROUP_232
ip pim rp-announce-filter rp-list ANY_RP group-list ADMIN_SCOPE
ip mroute 154.1.0.4 255.255.255.255 Tunnel34
ip mroute 150.1.5.5 255.255.255.255 Tunnel35
!
ip access-list standard GROUP_224
 permit 224.0.0.0 7.255.255.255
!
ip access-list standard GROUP_232
 permit 232.0.0.0 3.255.255.255
 permit 236.0.0.0 1.255.255.255
 permit 238.0.0.0 0.255.255.255
!
ip access-list standard ADMIN_SCOPE
 deny 239.0.0.0 0.255.255.255
!
ip access-list standard R4
 permit 154.1.0.4
!
ip access-list standard R5
 permit 150.1.5.5
!
ip access-list standard R6
 permit 150.1.6.6
!
ip access-list standard ANY_RP
 deny 154.1.0.4
 deny 150.1.5.5
 deny 150.1.6.6
 permit any
```


R4:

```
ip pim send-rp-announce Serial0/0/0 scope 16 group-list GROUP_224
!
ip access-list standard GROUP_224
 permit 224.0.0.0 7.255.255.255
```

R5:

```
!
interface Loopback0
 ip pim sparse-mode
!
ip pim send-rp-announce Loopback0 scope 16 group-list GROUP_224
!
ip access-list standard GROUP_224
 permit 224.0.0.0 7.255.255.255
```

R6:

```
!
interface Loopback0
 ip pim sparse-mode
!
ip pim send-rp-announce Loopback0 scope 16 group-list GROUP_232
!
ip access-list standard GROUP_232
 permit 232.0.0.0 3.255.255.255
 permit 236.0.0.0 1.255.255.255
 permit 238.0.0.0 0.255.255.255
```


Task 5.1 Verification

Verify RP-mapping at Mapping Agent:

```
Rack1R3#show ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP-mapping agent (Loopback0)

Group(s) 224.0.0.0/5
  RP 154.1.0.4 (?), v2v1
    Info source: 154.1.0.4 (?), elected via Auto-RP
         Uptime: 00:04:53, expires: 00:02:04
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.5.5 (?), via Auto-RP
         Uptime: 00:03:36, expires: 00:02:19
Group(s) 232.0.0.0/6
  RP 150.1.6.6 (?), v2v1
    Info source: 150.1.6.6 (?), elected via Auto-RP
         Uptime: 00:03:23, expires: 00:02:35
Group(s) 236.0.0.0/7
  RP 150.1.6.6 (?), v2v1
    Info source: 150.1.6.6 (?), elected via Auto-RP
         Uptime: 00:03:23, expires: 00:02:36
Group(s) 238.0.0.0/8
  RP 150.1.6.6 (?), v2v1
    Info source: 150.1.6.6 (?), elected via Auto-RP
         Uptime: 00:03:23, expires: 00:02:34
```

Confirm that RP-mapping information is disseminated:

```
Rack1R1#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/5
  RP 154.1.0.4 (?), v2v1
    Info source: 150.1.3.3 (?), elected via Auto-RP
         Uptime: 00:05:16, expires: 00:02:36
Group(s) 232.0.0.0/6
  RP 150.1.6.6 (?), v2v1
    Info source: 150.1.3.3 (?), elected via Auto-RP
         Uptime: 00:04:02, expires: 00:02:38
Group(s) 236.0.0.0/7
  RP 150.1.6.6 (?), v2v1
    Info source: 150.1.3.3 (?), elected via Auto-RP
         Uptime: 00:04:02, expires: 00:02:38
Group(s) 238.0.0.0/8
  RP 150.1.6.6 (?), v2v1
    Info source: 150.1.3.3 (?), elected via Auto-RP
         Uptime: 00:04:02, expires: 00:02:38
```


Temporarily disable RP-announces at R4:

R4:
```
no ip pim send-rp-announce Serial0/0 scope 16 group-list GROUP_224
```

Wait some time for announces to expire, and check Mapping Agent. R5 will be the RP for 224.0.0.0/5 group:

```
Rack1R3#show ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP-mapping agent (Loopback0)

Group(s) 224.0.0.0/5
  RP 150.1.5.5 (?), v2v1
    Info source: 150.1.5.5 (?), elected via Auto-RP
         Uptime: 00:15:46, expires: 00:02:12
Group(s) 232.0.0.0/6
  RP 150.1.6.6 (?), v2v1
    Info source: 150.1.6.6 (?), elected via Auto-RP
         Uptime: 00:15:32, expires: 00:02:27
Group(s) 236.0.0.0/7
  RP 150.1.6.6 (?), v2v1
    Info source: 150.1.6.6 (?), elected via Auto-RP
         Uptime: 00:15:32, expires: 00:02:26
Group(s) 238.0.0.0/8
  RP 150.1.6.6 (?), v2v1
    Info source: 150.1.6.6 (?), elected via Auto-RP
         Uptime: 00:15:32, expires: 00:02:26
```

R4:
```
ip pim send-rp-announce Serial0/0 scope 16 group-list GROUP_224
```

To test that the MA filter any candidate RPs announcements for the administratively scoped multicast group, temporarely configure R5 as following:

R5:
```
ip access-list standard GROUP_224
 permit 239.0.0.0 0.255.255.255
```

```
Rack1R3#debug ip pim auto-rp
Auto-RP(0): Filtered 239.0.0.0/8 for RP 150.1.5.5
Auto-RP(0): Filtered 239.0.0.0/8 for RP 150.1.5.5

Rack1R3#undebug ip pim auto-rp
```


R5:
```
ip access-list standard GROUP_224
 no 20
```

Task 5.2 Solution

R6:
```
interface FastEthernet0/0.63
 ip multicast boundary AUTORP
!
ip access-list standard AUTORP
 deny   224.0.1.39
 deny   224.0.1.40
 permit 224.0.0.0 15.255.255.255
```

Task 5.2 Verification

```
Rack1R6#show ip multicast interface fastEthernet 0/0.63
FastEthernet0/0.63 is up, line protocol is up
  Internet address is 204.12.1.6/24
  Multicast routing: enabled
  Multicast switching: fast
  Multicast packets in/out: 0/0
  Multicast boundary: AUTORP     (in/out)
  Multicast TTL threshold: 0
  Multicast Tagswitching: disabled
```

Simulate multicast client at BB3:

BB3:
```
ip multicast-routing
!
interface FastEthernet0
 ip pim sparse-mode
 ip igmp join-group 224.0.1.39
 ip igmp join-group 224.0.1.40
```

Before applying multicast-boundary:

```
Rack1R6#show ip igmp groups
IGMP Connected Group Membership
Group Address  Interface              Uptime    Expires   Last Reporter
224.0.1.39     FastEthernet0/0.63   00:05:19  00:02:09  204.12.1.254
224.0.1.40     FastEthernet0/0.63   00:05:15  00:02:07  204.12.1.254
224.0.1.40     FastEthernet0/0.16   00:43:49  00:02:49  192.10.1.6
```



```
Rack1R6#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
<output omitted>

(*, 224.0.1.39), 00:28:54/stopped, RP 0.0.0.0, flags: DC
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0.63, Forward/Sparse, 00:02:08/00:00:00
    FastEthernet0/0.16, Forward/Sparse, 00:28:54/00:00:00
<output omitted>
```

After that:

```
Rack1R6#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
<output omitted>

(*, 224.0.1.39), 00:29:36/stopped, RP 0.0.0.0, flags: DC
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0.16, Forward/Sparse, 00:29:36/00:00:00
```

BB3:

```
no ip multicast-routing
!
interface FastEthernet0
 no ip pim sparse-mode
 no ip igmp join-group 224.0.1.39
 no ip igmp join-group 224.0.1.40
```


Task 6.1 Solution

R2:

```
ip cef
!
interface BVI1
 ip verify unicast source reachable-via any
```

Task 6.1 Verification

Verify uRPF configuration:

```
Rack1R2#show ip interface bvi 1 | section IP verify
  IP verify source reachable-via ANY
```

Task 6.2 Solution

R6:

```
interface Virtual-Template1
 ip access-group RFC_1918 in
!
ip access-list standard RFC_1918
 deny   10.0.0.0 0.255.255.255
 deny   172.16.0.0 0.15.255.255
 deny   192.168.0.0 0.0.255.255
 permit any
```

Task 6.2 Verification

```
Rack1R6#show ip interface virtual-template 1 | section Inbound
  Inbound  access list is RFC_1918

Rack1R6#show ip access-lists RFC_1918
Standard IP access list RFC_1918
    10 deny   10.0.0.0, wildcard bits 0.255.255.255
    20 deny   172.16.0.0, wildcard bits 0.15.255.255
    30 deny   192.168.0.0, wildcard bits 0.0.255.255
    40 permit any (114 matches)
```


Task 7.1 Solution

R2 & R6:

```
!
aaa new-model
aaa authentication login default line
aaa authorization exec default if-authenticated
aaa authentication login NO_AUTH none
aaa authentication attempts login 1
aaa authentication fail-message ^
Authentication Failed. Username or Password was Incorrect
^
line con 0
 login authentication NO_AUTH
```

Task 7.1 Verification

Telnet to R2:

```
Rack1R6#telnet 150.1.2.2
Trying 150.1.2.2 ... Open


User Access Verification

Password: <cccc>

Authentication Failed. Username or Password was Incorrect

[Connection to 150.1.2.2 closed by foreign host]
```


Task 7.2 Solution

R2 & R6:

```
aaa authentication password-prompt "Passcode: "
aaa authentication username-prompt "Login Name: "
aaa authentication login default local
username cisco password 0 cisco
```

Task 7.2 Verification

Telnet to R6:

```
Rack1R6#telnet 150.1.6.6
Trying 150.1.6.6 ... Open


User Access Verification

Login Name: cisco
Passcode: <cisco>

Rack1R6>
```

Task 7.3 Solution

R6:

```
!
interface FastEthernet0/0.16
 ip nat inside
!
interface FastEthernet0/0.63
 ip nat outside
!
interface Virtual-Template1
 ip nat outside
!
ip nat inside source static tcp 192.10.1.112 22 54.1.8.6 22
ip nat inside source static tcp 192.10.1.112 23 54.1.8.6 23
ip nat inside source static tcp 192.10.1.112 22 204.12.1.6 22
ip nat inside source static tcp 192.10.1.112 23 204.12.1.6 23
```

Task 7.3 Verification

Verify NAT table before any sessions have been established:

```
Rack1R6#sho ip nat translations
Pro Inside global      Inside local       Outside local      Outside
global
tcp 54.1.8.6:22        192.10.1.112:22    ---                ---
tcp 54.1.8.6:23        192.10.1.112:23    ---                ---
tcp 204.12.1.6:22      192.10.1.112:22    ---                ---
tcp 204.12.1.6:23      192.10.1.112:23    ---                ---
```

Initiate session from BB1/BB3 and verify translations again:

```
BB3#telnet 204.12.1.6 22
Trying 204.12.1.6, 22 ...

BB1>telnet 54.1.8.6 23
Trying 54.1.8.6 ...

Rack1R6#show ip nat translations
Pro Inside global   Inside local       Outside local      Outside
global
tcp 204.12.1.6:22   192.10.1.112:22    204.12.1.254:11195
204.12.1.254:11195
tcp 54.1.8.6:22     192.10.1.112:22    ---                ---
tcp 54.1.8.6:23     192.10.1.112:23    54.1.8.254:21187
54.1.8.254:21187
tcp 54.1.8.6:23     192.10.1.112:23    ---                ---
tcp 204.12.1.6:22   192.10.1.112:22    ---                ---
tcp 204.12.1.6:23   192.10.1.112:23    ---                ---
```


Task 7.4 Solution

```
R4:
interface Loopback44
ip address 154.1.44.4 255.255.255.0
ip nat inside
!
interface FastEthernet0/0
ip nat outside
!
interface Serial0/0
ip nat outside
!
interface FastEthernet0/1
ip nat outside
!
ip nat inside source list NAT interface Loopback0 overload
ip ip telnet source-interface Loopback44
!
ip access-list extended NAT
permit tcp host 154.1.44.4 any eq telnet
```

Task 7.5 Verification

Telnet from R4 to R1:

```
Rack1R4#debug ip tcp transactions
TCP special event debugging is on

Rack1R4#telnet 150.1.1.1
Trying 150.1.1.1 ... Open


User Access Verification

Password:

TCP: Random local port generated 15392
TCB65C28EA8 created
TCB65C28EA8 setting property TCP_TOS (11) 65A34198
TCB65C28EA8 bound to 154.1.44.4.15392
TCP: sending SYN, seq 4184047953, ack 0
TCP0: Connection to 150.1.1.1:23, advertising MSS 536
TCP0: state was CLOSED -> SYNSENT [15392 -> 150.1.1.1(23)]
TCP0: state was SYNSENT -> ESTAB [15392 -> 150.1.1.1(23)]
TCP: tcb 65C28EA8 connection to 150.1.1.1:23, peer MSS 536, MSS is 536
TCB65C28EA8 connected to 150.1.1.1.23

Password: <cisco>
Rack1R1>en
Password: <cisco>
```


Verify the source address of the connection:

```
Rack1R1#show tcp brief
TCB        Local Address               Foreign Address         (state)
8340F280   150.1.1.1.23                150.1.4.4.38441         ESTAB
83758978   154.1.13.1.179              154.1.13.3.44104        ESTAB
8375D4AC   192.10.1.1.11000            192.10.1.2.179          ESTAB
8375A2D8   192.10.1.1.179              192.10.1.6.44781        ESTAB

Rack1R1#exit

[Connection to 150.1.1.1 closed by foreign host]
```

Check for the NAT translation on R4:

```
Rack1R4#show ip nat translations
Pro Inside global     Inside local       Outside local      Outside
global
tcp 150.1.4.4:38441   154.1.44.4:38441   150.1.1.1:23       150.1.1.1:23
```


Task 8.1 Solution

For the Be calculation on spokes, we assume that T1 speed is 1536Kbps

R3:

```
interface Serial1/0
 frame-relay traffic-shaping
 frame-relay interface-dlci 304
  class DLCI_304
 frame-relay interface-dlci 305
  class DLCI_305
!
map-class frame-relay DLCI_304
 frame-relay cir 1024000
 frame-relay bc 10240
!
map-class frame-relay DLCI_305
 frame-relay cir 512000
 frame-relay bc 5120
```

R4:

```
interface Serial0/0/0
 frame-relay traffic-shaping
 frame-relay interface-dlci 403
  class DLCI_403
!
map-class frame-relay DLCI_403
 frame-relay cir 1024000
 frame-relay bc 10240
 frame-relay be 5120
```

R5:

```
interface Serial0/0/0
 frame-relay traffic-shaping
 frame-relay interface-dlci 503
  class DLCI_503
!
map-class frame-relay DLCI_503
 frame-relay cir 512000
 frame-relay bc 5120
 frame-relay be 10240
```


Task 8.1 Verification

Verify FRTS configuration:

```
Rack1R3#show frame-relay pvc 304

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

DLCI = 304, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial1/0

  input pkts 719           output pkts 1656         in bytes 59701
  out bytes 138706         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 21        out bcast bytes 6972
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 04:03:28, last time pvc status changed 03:57:41
  cir 1024000   bc 10240     be 0         byte limit 1280   interval 10
  mincir 512000    byte increment 1280  Adaptive Shaping none
  pkts 14        bytes 1210      pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/40, 0 drop, 0 dequeued

Rack1R3#show frame-relay pvc 305

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

DLCI = 305, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial1/0

  input pkts 708           output pkts 453          in bytes 63554
  out bytes 47292          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 21        out bcast bytes 6972
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 04:03:32, last time pvc status changed 03:57:34
  cir 512000    bc 5120      be 0         byte limit 640    interval 10
  mincir 256000    byte increment 640   Adaptive Shaping none
  pkts 2         bytes 176       pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/40, 0 drop, 0 dequeued
```



```
Rack1R5#show frame-relay pvc 503

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

DLCI = 503, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0

  input pkts 458           output pkts 940          in bytes 47928
  out bytes 82590          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 21        out bcast bytes 6972
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 04:05:12, last time pvc status changed 03:59:07
  cir 512000    bc 5120      be 10240     byte limit 1920   interval 10
  mincir 256000    byte increment 640   Adaptive Shaping none
  pkts 20        bytes 1921      pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/40, 0 drop, 0 dequeued

Rack1R4#show frame-relay pvc 403

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

DLCI = 403, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0

  input pkts 1463          output pkts 972          in bytes 122170
  out bytes 80251          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 21        out bcast bytes 6972
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 04:05:53, last time pvc status changed 04:00:03
  cir 1024000   bc 10240     be 5120      byte limit 1920   interval 10
  mincir 512000    byte increment 1280  Adaptive Shaping none
  pkts 24        bytes 1917      pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/40, 0 drop, 0 dequeued
```


Task 8.2 Solution

R5:

```
class-map match-all QUAKE_DURING_WORK_HOURS
match access-group name QUAKE_DURING_WORK_HOURS
!
policy-map DROP_QUAKE_DURING_WORK_HOURS
class QUAKE_DURING_WORK_HOURS
drop
!
interface FastEthernet0/0
service-policy input DROP_QUAKE_DURING_WORK_HOURS
!
ip access-list extended QUAKE_DURING_WORK_HOURS
permit udp host 154.1.5.100 154.1.47.0 0.0.0.255 eq 27960 time-range
WORK_HOURS
permit udp host 154.1.5.100 154.1.3.0 0.0.0.255 eq 27960 time-range
WORK_HOURS
!
time-range WORK_HOURS
periodic weekdays 9:00 to 11:59
periodic weekdays 13:00 to 16:59
```


Task 8.2 Verification

Verify policy-map configuration:

```
Rack1R5#show policy-map interface FastEthernet 0/0
 FastEthernet0/0

  Service-policy input: DROP_QUAKE_DURING_WORK_HOURS

    Class-map: QUAKE_DURING_WORK_HOURS (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name QUAKE_DURING_WORK_HOURS
      drop

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

Rack1R5#show ip access-list QUAKE_DURING_WORK_HOURS
Extended IP access list QUAKE_DURING_WORK_HOURS
    10 permit udp host 154.1.5.100 154.1.47.0 0.0.0.255 eq 27960 time-
range WORK_HOURS (inactive)
    20 permit udp host 154.1.5.100 154.1.3.0 0.0.0.255 eq 27960 time-
range WORK_HOURS (inactive)

Rack1R5#show time-range
time-range entry: WORK HOURS (inactive)
   periodic weekdays 9:00 to 11:59
   periodic weekdays 13:00 to 16:59
   used in: IP ACL entry
   used in: IP ACL entry
```


Task 8.3 Solution

R5:

```
class-map match-all QUAKE_TO_VLAN3003
 match access-group name QUAKE_TO_VLAN3003
!
policy-map PRIORITY_FOR_QUAKE
 class QUAKE_TO_VLAN3003
  priority percent 99
!
ip access-list extended QUAKE_TO_VLAN3003
 permit udp host 154.1.5.100 eq 27960 154.1.3.0 0.0.0.255
!
map-class frame-relay DLCI_503
 frame-relay mincir 1536000
 service-policy output PRIORITY_FOR_QUAKE
```

Quick Note
Maximum possible output rate is (Bc + Be) * 1000/Tc.

Task 8.3 Verification

Verify policy-map configuration:

```
Rack1R5#show policy-map interface serial 0/0/0
 Serial0/0/0: DLCI 503 -

  Service-policy output: PRIORITY_FOR_QUAKE

    queue stats for all priority classes:

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

    Class-map: QUAKE_TO_VLAN3003 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name QUAKE_TO_VLAN3003
      Priority: 99% (506 kbps), burst bytes 12650, b/w exceed drops: 0


    Class-map: class-default (match-any)
      341 packets, 32826 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

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


Task 8.4 Solution

R1:

```
interface Serial 0/1
 compress stac
```

R2:

```
interface Serial 0/1
 compress stac
```

R3:

```
interface Serial 1/2
 compress stac
!
interface Serial 1/3
 compress stac
```

Task 8.4 Verification

```
Rack1R1#ping 154.1.13.3 repeat 100 size 500

Type escape sequence to abort.
Sending 100, 500-byte ICMP Echos to 154.1.13.3, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max =
16/18/36 ms

Rack1R1#show compress
 Serial0/1
         Software compression enabled
         uncompressed bytes xmt/rcv 52180/51456
         compressed bytes   xmt/rcv 4481/4547
         Compressed bytes sent:      4481 bytes   0 Kbits/sec  ratio:
11.644
         Compressed bytes recv:      4547 bytes   0 Kbits/sec  ratio:
11.316
         1  min avg ratio xmt/rcv 8.767/8.903
         5  min avg ratio xmt/rcv 7.806/8.113
         10 min avg ratio xmt/rcv 7.806/8.113
         no bufs xmt 0 no bufs rcv 0
         resyncs 0
         Additional Stac Stats:
         Transmit bytes:  Uncompressed =        0 Compressed =
4481
         Received bytes:  Compressed =       4659 Uncompressed =
```

Lab 17 Solutions

Task 1.1 Solution

R4:

```
interface Serial0/0/0
 frame-relay interface-dlci 405
  class DLCI_405
!
map-class frame-relay DLCI_405
 frame-relay end-to-end keepalive mode bidirectional
```

R5:

```
interface Serial0/0/0.54 point-to-point
backup interface Serial0/1/0
frame-relay interface-dlci 504
class DLCI_504
!
map-class frame-relay DLCI_504
frame-relay end-to-end keepalive mode bidirectional
!
interface Serial0/1/0
  clock rate 64000
```

Task 1.1 Verification

```
Rack1R5#show frame-relay end-to-end keepalive

End-to-end Keepalive Statistics for Interface Serial0/0 (Frame Relay
DTE)

DLCI = 504, DLCI USAGE = LOCAL, VC STATUS = ACTIVE (EEK UP)

SEND SIDE STATISTICS

Send Sequence Number: 1,        Receive Sequence Number: 2
Configured Event Window: 3,     Configured Error Threshold: 2
Total Observed Events: 4,       Total Observed Errors: 0
Monitored Events: 3,            Monitored Errors: 0
Successive Successes: 3,        End-to-end VC Status: UP

RECEIVE SIDE STATISTICS

Send Sequence Number: 2,        Receive Sequence Number: 1
Configured Event Window: 3,     Configured Error Threshold: 2
Total Observed Events: 4,       Total Observed Errors: 0
Monitored Events: 3,            Monitored Errors: 0
Successive Successes: 3,        End-to-end VC Status: UP
```



```
Rack1R5#show backup
Primary Interface   Secondary Interface   Status
-----------------   -------------------   ------
Serial0/0.54        Serial0/1             normal operation
```

To verify this task simulate a link fault:

```
Rack1R4(config)#interface s0/0
Rack1R4(config-if)#shutdown

Rack1R5#debug backup
Backup events debugging is on
BACKUP(Serial0/0.54): event = primary interface went down
BACKUP(Serial0/0.54): changed state to "waiting to backup"
BACKUP(Serial0/0.54): event = timer expired on primary
BACKUP(Serial0/0.54): secondary interface (Serial0/1) made active
BACKUP(Serial0/0.54): changed state to "backup mode"
%LINK-3-UPDOWN: Interface Serial0/1, changed state to up
BACKUP(Serial0/1): event = secondary interface came up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1, changed
state to up
BACKUP(Serial0/1): event = secondary interface came up

Rack1R5#show backup
Primary Interface   Secondary Interface   Status
-----------------   -------------------   ------
Serial0/0.54        Serial0/1             backup mode
```


Task 2.1 Solution

R1:
```
interface FastEthernet0/0
 ip ospf network point-to-multipoint
!
router ospf 1
 network 173.1.137.1 0.0.0.0 area 137
```

R3:
```
interface FastEthernet0/0
 ip ospf network point-to-multipoint
!
router ospf 1
 network 173.1.137.3 0.0.0.0 area 137
```

SW1:
```
ip routing
!
interface Vlan137
 ip ospf network point-to-multipoint
!
router ospf 1
 router-id 150.1.7.7
 network 173.1.137.7 0.0.0.0 area 137
 neighbor 173.1.137.3 cost 10
 neighbor 173.1.137.1 cost 1
```

R2:
```
router ospf 1
 network 173.1.23.2 0.0.0.0 area 23
```

R3:
```
router ospf 1
 network 173.1.23.3 0.0.0.0 area 23
```

SW1:
```
router ospf 1
 redistribute connected subnets route-map CONNECTED2OSPF
!
route-map CONNECTED2OSPF permit 10
 match interface Loopback0
```

SW2:
```
ip routing
!
router ospf 1
 router-id 150.1.8.8
 network 173.1.8.8 0.0.0.0 area 23
 redistribute connected subnets route-map CONNECTED2OSPF
 network 173.1.23.8 0.0.0.0 area 23
 no network 0.0.0.0 255.255.255.255 area 0
!
route-map CONNECTED2OSPF permit 10
 match interface Loopback0
```


R3:

```
router ospf 1
 area 23 nssa
 area 23 nssa default-information-originate
```

R2 & SW2:

```
router ospf 1
area 23 nssa
```

Task 2.1 Verification

```
Rack1SW1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.1.1         0   FULL/  -        00:01:39    173.1.137.1     Vlan137
150.1.3.3         0   FULL/  -        00:01:50    173.1.137.3     Vlan137

Rack1SW1#show ip ospf interface vlan 137
Vlan137 is up, line protocol is up
  Internet Address 173.1.137.7/24, Area 137
  Process ID 1, Router ID 150.1.7.7, Network Type POINT_TO_MULTIPOINT,
Cost: 1
  Transmit Delay is 1 sec, State POINT_TO_MULTIPOINT,
  Timer intervals configured, Hello 30, Dead 120, Wait 120,Retransmit 5
    oob-resync timeout 120
    Hello due in 00:00:04
  Supports Link-local Signaling (LLS)
  Index 1/1, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 2, Adjacent neighbor count is 2
    Adjacent with neighbor 150.1.1.1, cost is 1
    Adjacent with neighbor 150.1.3.3, cost is 10
  Suppress hello for 0 neighbor(s)

Rack1SW1#show ip route ospf
     173.1.0.0/16 is variably subnetted, 6 subnets, 2 masks
O       173.1.137.1/32 [110/1] via 173.1.137.1, 00:02:15, Vlan137
O       173.1.137.3/32 [110/2] via 173.1.137.1, 00:02:15, Vlan137
O IA    173.1.32.0/24 [110/129] via 173.1.137.1, 00:02:15, Vlan137
O IA    173.1.13.0/24 [110/65] via 173.1.137.1, 00:02:15, Vlan137
O IA    173.1.125.0/24 [110/65] via 173.1.137.1, 00:02:15, Vlan137
     150.1.0.0/16 is variably subnetted, 5 subnets, 2 masks
O IA    150.1.5.5/32 [110/66] via 173.1.137.1, 00:02:15, Vlan137
O IA    150.1.3.3/32 [110/3] via 173.1.137.1, 00:02:16, Vlan137
O IA    150.1.2.2/32 [110/66] via 173.1.137.1, 00:02:16, Vlan137
O IA    150.1.1.1/32 [110/2] via 173.1.137.1, 00:02:16, Vlan137
```



```
Rack1SW2#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.2.2         1   FULL/BDR        00:00:37    173.1.23.2      Vlan23
150.1.3.3         1   FULL/DR         00:00:39    173.1.23.3      Vlan23

Rack1R5#show ip route ospf
     173.1.0.0/16 is variably subnetted, 10 subnets, 2 masks
O IA    173.1.137.7/32 [110/65] via 173.1.125.1, 00:02:15,
Serial0/0.125
O IA    173.1.137.1/32 [110/64] via 173.1.125.1, 00:02:15,
Serial0/0.125
O IA    173.1.137.3/32 [110/65] via 173.1.125.1, 00:02:15,
Serial0/0.125
O       173.1.32.0/24 [110/128] via 173.1.125.2, 00:02:15,
Serial0/0.125
O       173.1.13.0/24 [110/128] via 173.1.125.1, 00:02:15,
Serial0/0.125
O IA    173.1.8.0/24 [110/66] via 173.1.125.2, 00:00:59, Serial0/0.125
O IA    173.1.23.0/24 [110/65] via 173.1.125.2, 00:01:13, Serial0/0.125
     150.1.0.0/16 is variably subnetted, 6 subnets, 2 masks
O E2    150.1.7.0/24 [110/20] via 173.1.125.1, 00:00:54, Serial0/0.125
O       150.1.3.3/32 [110/129] via 173.1.125.2, 00:02:15, Serial0/0.125
                     [110/129] via 173.1.125.1, 00:02:15, Serial0/0.125
O       150.1.2.2/32 [110/65] via 173.1.125.2, 00:02:15, Serial0/0.125
O       150.1.1.1/32 [110/65] via 173.1.125.1, 00:02:15, Serial0/0.125
O E2    150.1.8.0/24 [110/20] via 173.1.125.2, 00:00:55, Serial0/0.125

Rack1SW2#show ip ospf | beg Area 23
    Area 23
        Number of interfaces in this area is 2
        It is a NSSA area
        Area has no authentication
        SPF algorithm last executed 00:00:25.260 ago
        SPF algorithm executed 7 times
        Area ranges are
        Number of LSA 27. Checksum Sum 0x0E2136
        Number of opaque link LSA 0. Checksum Sum 0x000000
        Number of DCbitless LSA 0
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0

Rack1SW2#show ip route ospf
     173.1.0.0/16 is variably subnetted, 8 subnets, 2 masks
O IA    173.1.137.7/32 [110/11] via 173.1.23.3, 00:01:10, Vlan23
O IA    173.1.137.1/32 [110/11] via 173.1.23.3, 00:01:10, Vlan23
O IA    173.1.137.3/32 [110/1] via 173.1.23.3, 00:01:10, Vlan23
O IA    173.1.32.0/24 [110/65] via 173.1.23.2, 00:01:10, Vlan23
O IA    173.1.13.0/24 [110/129] via 173.1.23.2, 00:01:10, Vlan23
O IA    173.1.125.0/24 [110/65] via 173.1.23.2, 00:01:10, Vlan23
     150.1.0.0/16 is variably subnetted, 5 subnets, 2 masks
O IA    150.1.5.5/32 [110/66] via 173.1.23.2, 00:01:11, Vlan23
O IA    150.1.3.3/32 [110/2] via 173.1.23.3, 00:01:11, Vlan23
O IA    150.1.2.2/32 [110/2] via 173.1.23.2, 00:01:11, Vlan23
O IA    150.1.1.1/32 [110/66] via 173.1.23.2, 00:01:11, Vlan23
O*N2 0.0.0.0/0 [110/1] via 173.1.23.3, 00:01:11, Vlan23
```


Task 2.2 Solution

R5:

```
router eigrp 10
 redistribute ospf 1 metric 1500 1000 255 1 1500
!
router ospf 1
 redistribute eigrp 10 metric-type 1 subnets
```

SW1:

```
router ospf 1
 redistribute rip subnets
 summary-address 31.0.0.0 255.252.0.0
!
router rip
 redistribute ospf 1 metric 1
!
route-map CONNECTED2OSPF permit 20
 match interface Vlan73
```

Tasks 2.3 Verification

Verify that RIP routes learned from BB3 are summarized into OSPF

```
Rack1SW1#show ip route rip
     31.0.0.0/8 is variably subnetted, 5 subnets, 2 masks
R       31.3.0.0/16 [120/1] via 204.12.1.254, 00:00:08, Vlan73
R       31.2.0.0/16 [120/1] via 204.12.1.254, 00:00:08, Vlan73
R       31.1.0.0/16 [120/1] via 204.12.1.254, 00:00:08, Vlan73
R       31.0.0.0/16 [120/1] via 204.12.1.254, 00:00:08, Vlan73

Rack1R1#show ip route ospf | i 31
     31.0.0.0/14 is subnetted, 1 subnets
O E2    31.0.0.0 [110/20] via 173.1.137.7, 00:01:19, FastEthernet0/0
```


Verify that OSPF routes are advertised into RIP:

```
Rack1SW1#debug ip rip
RIP: sending v2 update to 224.0.0.9 via Vlan73 (204.12.1.7)
RIP: build update entries
31.0.0.0/14 via 0.0.0.0, metric 1, tag 0
54.1.2.0/24 via 0.0.0.0, metric 1, tag 0
150.1.1.1/32 via 0.0.0.0, metric 1, tag 0
150.1.2.2/32 via 0.0.0.0, metric 1, tag 0
150.1.3.3/32 via 0.0.0.0, metric 1, tag 0
150.1.4.0/24 via 0.0.0.0, metric 1, tag 0
150.1.5.5/32 via 0.0.0.0, metric 1, tag 0
150.1.6.0/24 via 0.0.0.0, metric 1, tag 0
150.1.8.0/24 via 0.0.0.0, metric 1, tag 0
173.1.4.0/24 via 0.0.0.0, metric 1, tag 0
173.1.5.0/24 via 0.0.0.0, metric 1, tag 0
173.1.8.0/24 via 0.0.0.0, metric 1, tag 0
173.1.13.0/24 via 0.0.0.0, metric 1, tag 0
173.1.23.0/24 via 0.0.0.0, metric 1, tag 0
173.1.32.0/24 via 0.0.0.0, metric 1, tag 0
173.1.44.0/24 via 0.0.0.0, metric 1, tag 0
173.1.46.0/24 via 0.0.0.0, metric 1, tag 0
173.1.54.0/24 via 0.0.0.0, metric 1, tag
173.1.125.0/24 via 0.0.0.0, metric 1, tag 0
173.1.137.0/24 via 0.0.0.0, metric 1, tag 0
173.1.137.1/32 via 0.0.0.0, metric 1, tag 0
173.1.137.3/32 via 0.0.0.0, metric 1, tag 0
192.10.1.0/24 via 0.0.0.0, metric 1, tag 0
200.0.0.0/24 via 0.0.0.0, metric 1, tag 0
200.0.1.0/24 via 0.0.0.0, metric 1, tag 0
200.0.2.0/24 via 0.0.0.0, metric 1, tag 0
200.0.3.0/24 via 0.0.0.0, metric 1, tag 0

Rack1SW1#undebug all
```


Verify redistribution between EIGRP and OSPF on R5:

```
Rack1R4#show ip route eigrp | i EX
D EX 204.12.1.0/24 [170/2474496] via 173.1.54.5, 00:19:31, Serial0/0/0
D EX    173.1.137.7/32 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
D EX    173.1.137.1/32 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
D EX    173.1.137.3/32 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
D EX    173.1.32.0/24 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
D EX    173.1.13.0/24 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
D EX    173.1.8.0/24 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
D EX    173.1.23.0/24 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
D EX    173.1.125.0/24 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
D EX 192.10.1.0/24 [170/2474496] via 173.1.54.5, 00:20:42, Serial0/0/0
D EX    31.0.0.0 [170/2474496] via 173.1.54.5, 00:19:31, Serial0/0/0
D EX    150.1.7.0/24 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
D EX    150.1.5.0/24 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
D EX    150.1.3.3/32 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
D EX    150.1.2.2/32 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
D EX    150.1.1.1/32 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
D EX    150.1.8.0/24 [170/2474496] via 173.1.54.5, 00:20:42,
Serial0/0/0
```

```
Rack1R1#show ip route ospf | i E1
O E1 200.0.0.0/24 [110/212] via 173.1.13.3, 00:20:40, Serial0/1
O E1    54.1.2.0 [110/212] via 173.1.13.3, 00:20:40, Serial0/1
O E1 200.0.1.0/24 [110/212] via 173.1.13.3, 00:20:40, Serial0/1
O E1 200.0.2.0/24 [110/212] via 173.1.13.3, 00:20:40, Serial0/1
O E1 200.0.3.0/24 [110/212] via 173.1.13.3, 00:20:40, Serial0/1
O E1    173.1.44.0/24 [110/212] via 173.1.13.3, 00:20:40, Serial0/1
O E1    173.1.46.0/24 [110/212] via 173.1.13.3, 00:20:40, Serial0/1
O E1    173.1.4.0/24 [110/212] via 173.1.13.3, 00:20:40, Serial0/1
O E1    150.1.6.0/24 [110/212] via 173.1.13.3, 00:20:40, Serial0/1
O E1    150.1.4.0/24 [110/212] via 173.1.13.3, 00:20:40, Serial0/1
```


Confirm full connectivity with the following Tcl script:

```
foreach i {
173.1.137.1
173.1.125.1
173.1.13.1
150.1.1.1
173.1.23.2
173.1.125.2
173.1.32.2
150.1.2.2
173.1.137.3
173.1.23.3
173.1.13.3
173.1.32.3
150.1.3.3
173.1.46.4
173.1.54.4
173.1.4.4
173.1.44.4
150.1.4.4
173.1.54.5
173.1.125.5
173.1.5.5
150.1.5.5
192.10.1.5
173.1.46.6
54.1.2.6
150.1.6.6
204.12.1.7
173.1.137.7
150.1.7.7
173.1.8.8
173.1.23.8
150.1.8.8
173.1.4.9
173.1.109.9
173.1.5.10
173.1.109.10
} { ping $i }
```


Task 2.3 Solution

R1:
```
router bgp 100
 neighbor 173.1.32.2 send-community
 neighbor 173.1.125.5 route-map NO_EXPORT in
 neighbor 173.1.137.3 send-community
 neighbor 173.1.137.7 send-community
!
route-map NO_EXPORT
 set community no-export
```

R2:
```
router bgp 100
 neighbor 173.1.13.1 send-community
 neighbor 173.1.23.3 send-community
 neighbor 173.1.125.5 route-map NO_EXPORT in
 neighbor 173.1.137.7 send-community
!
route-map NO_EXPORT
 set community no-export
```

R3:
```
router bgp 100
 neighbor 173.1.23.2 send-community
 neighbor 173.1.137.1 send-community
 neighbor 173.1.137.7 send-community
```

SW1:
```
router bgp 100
 neighbor 150.1.2.2 send-community
 neighbor 173.1.137.1 send-community
 neighbor 173.1.137.3 send-community
 neighbor 204.12.1.254 route-map NO_EXPORT in
!
route-map NO_EXPORT
 set community no-export
```

R2:
```
router bgp 100
neighbor 173.1.13.1 route-map WEIGHT in
!
ip as-path access-list 1 permit _254$
!
route-map WEIGHT permit 10
match as-path 1
set weight 1
set ip next-hop 173.1.13.1
!
route-map WEIGHT permit 20
```

R5:
```
router bgp 200
 bgp scan-time 20
 bgp update-delay 12
```


Task 2.3 Verification

```
Rack1SW1#show ip bgp q _200_
BGP table version is 28, local router ID is 150.1.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
r i192.10.1.0       150.1.2.2                0    100      0 200 i
r>i                 173.1.125.5              0    100      0 200 i
* i205.90.31.0      150.1.2.2                0    100      0 200 254 ?
*>i                 173.1.125.5              0    100      0 200 254 ?
* i220.20.3.0       150.1.2.2                0    100      0 200 254 ?
*>i                 173.1.125.5              0    100      0 200 254 ?
* i222.22.2.0       150.1.2.2                0    100      0 200 254 ?
*>i                 173.1.125.5              0    100      0 200 254 ?
```

```
Rack1SW1#show ip bgp 192.10.1.0
BGP routing table entry for 192.10.1.0/24, version 39
Paths: (2 available, best #2, table Default-IP-Routing-Table, not
advertised to EBGP peer, RIB-failure(17))
  Not advertised to any peer
  200
    150.1.2.2 (metric 67) from 150.1.2.2 (150.1.2.2)
      Origin IGP, metric 0, localpref 100, valid, internal
      Community: no-export
  200
    173.1.125.5 (metric 65) from 173.1.137.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Community: no-export
```

Check for AS54 prefixes in AS100 BGP tables:

```
Rack1R1#show ip bgp q _54_
BGP table version is 31, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   204.12.1.254             0    100      0 54 i
*>i28.119.17.0/24   204.12.1.254             0    100      0 54 i
*>i112.0.0.0        204.12.1.254             0    100      0 54 50 60 i
*>i113.0.0.0        204.12.1.254             0    100      0 54 50 60 i
*>i114.0.0.0        204.12.1.254             0    100      0 54 i
*>i115.0.0.0        204.12.1.254             0    100      0 54 i
*>i116.0.0.0        204.12.1.254             0    100      0 54 i
*>i117.0.0.0        204.12.1.254             0    100      0 54 i
*>i118.0.0.0        204.12.1.254             0    100      0 54 i
*>i119.0.0.0        204.12.1.254             0    100      0 54 i
```



```
Rack1R1#show ip bgp 28.119.16.0
BGP routing table entry for 28.119.16.0/24, version 26
Paths: (1 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
  Not advertised to any peer
  54
    204.12.1.254 (metric 20) from 173.1.137.7 (150.1.7.7)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Community: no-export

Rack1R2#show ip bgp q _254$
BGP table version is 34, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i205.90.31.0      173.1.13.1               0    100      1 200 254 ?
*                   173.1.125.5                            0 200 254 ?
*>i220.20.3.0       173.1.13.1               0    100      1 200 254 ?
*                   173.1.125.5                            0 200 254 ?
*>i222.22.2.0       173.1.13.1               0    100      1 200 254 ?
*                   173.1.125.5                            0 200 254 ?

Rack1R2#show ip bgp 205.90.31.0
BGP routing table entry for 205.90.31.0/24, version 32
Paths: (2 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
  Not advertised to any peer
  200 254
    173.1.13.1 (metric 128) from 173.1.13.1 (150.1.1.1)
      Origin incomplete, metric 0, localpref 100, weight 1, valid,
internal, best
      Community: no-export
  200 254
    173.1.125.5 from 173.1.125.5 (150.1.5.5)
      Origin incomplete, localpref 100, valid, external
      Community: no-export
```


Task 2.4 Solution

R3:

```
key chain OER
 key 1
  key-string CISCO
!
oer master
 logging
 !
 border 150.1.1.1 key-chain OER
  interface Serial0/0 external
   max-xmit-utilization percentage 60
  interface Serial0/1 internal
  interface FastEthernet0/0 internal
 !
 border 150.1.2.2 key-chain OER
  interface Serial0/0 external
   max-xmit-utilization percentage 60
  interface Serial0/1 internal
  interface FastEthernet0/0 internal
 !
 learn
  throughput
  delay
  periodic-interval 1
  monitor-period 1
  aggregation-type bgp
!
 delay threshold 200
 mode route control
 mode route metric bgp local-pref 5000
```

R1 & R2:

```
key chain OER
 key 1
  key-string CISCO
!
oer border
 local Loopback0
 master 150.1.3.3 key-chain OER
!
interface Serial 0/0
 bandwidth 64
 load-interval 30
```


R1, R2, R3:

```
!
! The below is needed because the target prefixes are being injected
! into OSPF on R5 via redistribution.
!
router ospf 1
 distance ospf external 201
```

R4:

```
router bgp 200
 network 150.1.4.0 mask 255.255.255.0
```

R6:

```
router bgp 200
 network 150.1.6.0 mask 255.255.255.0
```


Task 2.4 Verification

Verify OER configuration settings:

```
Rack1R2#show oer border
OER BR 150.1.2.2 ACTIVE, MC 150.1.3.3 UP/DOWN: UP 00:32:21,
  Auth Failures: 0
  Conn Status: SUCCESS, PORT: 3949
  Exits
  Fa0/0           INTERNAL
  Se0/0           EXTERNAL
  Se0/1           INTERNAL

Rack1R1#show oer border
OER BR 150.1.1.1 ACTIVE, MC 150.1.3.3 UP/DOWN: UP 00:45:14,
  Auth Failures: 0
  Conn Status: SUCCESS, PORT: 3949
  Exits
  Fa0/0           INTERNAL
  Se0/0           EXTERNAL
  Se0/1           INTERNAL

Rack1R3#show oer master
OER state: ENABLED and ACTIVE
  Conn Status: SUCCESS, PORT: 3949
  Number of Border routers: 2
  Number of Exits: 2
  Number of monitored prefixes: 2 (max 5000)
  Max prefixes: total 5000 learn 2500
  Prefix count: total 2, learn 2, cfg 0

Border           Status   UP/DOWN             AuthFail
150.1.2.2        ACTIVE   UP       00:33:11          0
150.1.1.1        ACTIVE   UP       00:45:52          0

Global Settings:
  max-range-utilization percent 20
  mode route metric bgp local-pref 5000
  mode route metric static tag 5000
  trace probe delay 1000
  logging
```



```
Default Policy Settings:
  backoff 300 3000 300
  delay threshold 200
  holddown 300
  periodic 0
  mode route control
  mode monitor both
  mode select-exit good
  loss relative 10
  unreachable relative 50
  resolve delay priority 11 variance 20
  resolve utilization priority 12 variance 20

Learn Settings:
  current state : STARTED
  time remaining in current state : 95 seconds
  throughput
  delay
  no protocol
  monitor-period 1
  periodic-interval 1
  aggregation-type bgp
  prefixes 100
  expire after time 720
```

To verify your configuration, start two normal ping flows from SW2 towards the IP addresses of R4's and R6's Loopback0 interfaces. Traffic should be balanced between R1 and R2:

```
Rack1SW2#ping 150.1.6.6 repeat 100000000

Type escape sequence to abort.
Sending 100000000, 100-byte ICMP Echos to 150.1.6.6, timeout is 2
seconds:
!!!!!

Rack1SW2#ping 150.1.4.4 repeat 100000000

Type escape sequence to abort.
Sending 100000000, 100-byte ICMP Echos to 150.1.4.4, timeout is 2
seconds:
!!!!
```



```
Rack1R3#show oer master prefix
OER Prefix Statistics:
 Pas - Passive, Act - Active, S - Short term, L - Long term, Dly -
Delay (ms),
 Los - Packet Loss (packets-per-million), Un - Unreachable (flows-per-
million),
 E - Egress, I - Ingress, Bw - Bandwidth (kbps), N - Not applicable
 U - unknown, * - uncontrolled, + - control more specific, @ - active
probe all

Prefix          State     Time Curr BR         CurrI/F         Protocol
              PasSDly  PasLDly   PasSUn   PasLUn  PasSLos  PasLLos
              ActSDly  ActLDly   ActSUn   ActLUn      EBw      IBw
--------------------------------------------------------------------------
150.1.4.0/24  INPOLICY      0 150.1.2.2       Se0/0           BGP
                    U      U        0        0        0        0
                    U      U        0        0       11       11
150.1.6.0/24  HOLDDOWN      86 150.1.1.1       Se0/0           BGP
                    U      U        0        0        0        0
                    U      U        0        0        9        9

Rack1R1#show oer border routes bgp
OER BR 150.1.1.1 ACTIVE, MC 150.1.3.3 UP/DOWN: UP 00:09:57,
  Auth Failures: 0
  Conn Status: SUCCESS, PORT: 3949
BGP table version is 57, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
OER Flags: C - Controlled, X - Excluded, E - Exact, N - Non-exact, I -
Injected

   Network          Next Hop        OER    LocPrf Weight Path
*>i150.1.4.0/24     173.1.32.2      XN       5000      0 200 i
*> 150.1.6.0/24     173.1.125.5     CE                 0 200 i

Rack1R1#show ip bgp 150.1.4.0
BGP routing table entry for 150.1.4.0/24, version 56
Paths: (2 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
  Not advertised to any peer
  200
    173.1.32.2 (metric 128) from 173.1.32.2 (150.1.2.2)
      Origin IGP, metric 0, localpref 5000, valid, internal, best
      Community: no-export
  200
    173.1.125.5 from 173.1.125.5 (150.1.5.5)
      Origin IGP, localpref 100, valid, external
      Community: no-export
```



```
Rack1R1#show ip route 150.1.4.0
Routing entry for 150.1.4.0/24
  Known via "bgp 100", distance 200, metric 0
  Tag 200, type internal
  Last update from 173.1.32.2 00:03:45 ago
  Routing Descriptor Blocks:
  * 173.1.32.2, from 173.1.32.2, 00:03:45 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
      Route tag 200

Rack1R2#show oer border routes bgp
OER BR 150.1.2.2 ACTIVE, MC 150.1.3.3 UP/DOWN: UP 00:10:27,
  Auth Failures: 0
  Conn Status: SUCCESS, PORT: 3949
BGP table version is 106, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
OER Flags: C - Controlled, X - Excluded, E - Exact, N - Non-exact, I -
Injected

   Network          Next Hop        OER    LocPrf Weight Path
*> 150.1.4.0/24     173.1.125.5     CE                0 200 i
*>i150.1.6.0/24     173.1.13.1      XN      5000      0 200 i

Rack1R2#show ip bgp 150.1.6.0
BGP routing table entry for 150.1.6.0/24, version 106
Paths: (2 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
Flag: 0x940
  Not advertised to any peer
  200
    173.1.13.1 (metric 128) from 173.1.13.1 (150.1.1.1)
      Origin IGP, metric 0, localpref 5000, valid, internal, best
      Community: no-export
  200
    173.1.125.5 from 173.1.125.5 (150.1.5.5)
      Origin IGP, localpref 100, valid, external
      Community: no-export
```


After this, generate a flood of ping packets on R1 towards R5 to oversubscribe the link:

```
Rack1R1#ping 173.1.125.5 repeat 10000000 timeout 0

Type escape sequence to abort.
Sending 10000000, 100-byte ICMP Echos to 173.1.125.5, timeout is 0
seconds:
 ............................................................
 ............................................................
```

And watch how OER shifts the prefixes towards R2 as the only exit:

```
Rack1R3#show oer master prefix
OER Prefix Statistics:
 Pas - Passive, Act - Active, S - Short term, L - Long term, Dly -
Delay (ms),
 Los - Packet Loss (packets-per-million), Un - Unreachable (flows-per-
million),
 E - Egress, I - Ingress, Bw - Bandwidth (kbps), N - Not applicable
 U - unknown, * - uncontrolled, + - control more specific, @ - active
probe all

Prefix          State     Time Curr BR         CurrI/F         Protocol
               PasSDly  PasLDly   PasSUn   PasLUn  PasSLos  PasLLos
               ActSDly  ActLDly   ActSUn   ActLUn      EBw      IBw
-------------------------------------------------------------------
150.1.4.0/24   INPOLICY      0 150.1.2.2       Se0/0           BGP
                     U       U         0         0        0        0
                     U       U         0         0        9        9
150.1.6.0/24   HOLDDOWN    294 150.1.2.2       Se0/0           BGP
                     U       U         0         0        0        0
                     U       U         0         0        4        4


Rack1R2#show oer border routes bgp
OER BR 150.1.2.2 ACTIVE, MC 150.1.3.3 UP/DOWN: UP 00:16:33,
  Auth Failures: 0
  Conn Status: SUCCESS, PORT: 3949
BGP table version is 107, local router ID is 150.1.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
OER Flags: C - Controlled, X - Excluded, E - Exact, N - Non-exact, I -
Injected

   Network          Next Hop        OER    LocPrf Weight Path
*> 150.1.4.0/24     173.1.125.5     CE                0 200 i
*> 150.1.6.0/24     173.1.125.5     CE                0 200 i
```



```
Rack1R2#show ip route 150.1.4.0
Routing entry for 150.1.4.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 200, type external
  Last update from 173.1.125.5 00:13:10 ago
  Routing Descriptor Blocks:
  * 173.1.125.5, from 173.1.125.5, 00:13:10 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
      Route tag 200

Rack1R2#show ip route 150.1.6.0
Routing entry for 150.1.6.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 200, type external
  Last update from 173.1.125.5 00:01:15 ago
  Routing Descriptor Blocks:
  * 173.1.125.5, from 173.1.125.5, 00:01:15 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
      Route tag 200

Rack1R1#show oer border routes bgp
OER BR 150.1.1.1 ACTIVE, MC 150.1.3.3 UP/DOWN: UP 00:17:28,
  Auth Failures: 0
  Conn Status: SUCCESS, PORT: 3949
BGP table version is 59, local router ID is 150.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
OER Flags: C - Controlled, X - Excluded, E - Exact, N - Non-exact, I -
Injected

   Network          Next Hop        OER    LocPrf Weight Path
*>i150.1.4.0/24     173.1.32.2      XN      5000      0 200 i
*>i150.1.6.0/24     173.1.32.2      XN      5000      0 200 i

Rack1R1#show ip route 150.1.4.0
Routing entry for 150.1.4.0/24
  Known via "bgp 100", distance 200, metric 0
  Tag 200, type internal
  Last update from 173.1.32.2 00:09:53 ago
  Routing Descriptor Blocks:
  * 173.1.32.2, from 173.1.32.2, 00:09:53 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
      Route tag 200
```



```
Rack1R1#show ip route 150.1.6.0
Routing entry for 150.1.6.0/24
  Known via "bgp 100", distance 200, metric 0
  Tag 200, type internal
  Last update from 173.1.32.2 00:01:58 ago
  Routing Descriptor Blocks:
  * 173.1.32.2, from 173.1.32.2, 00:01:58 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
      Route tag 200

Rack1R1#show ip bgp 150.1.4.0
BGP routing table entry for 150.1.4.0/24, version 56
Paths: (2 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
  Not advertised to any peer
  200
    173.1.32.2 (metric 128) from 173.1.32.2 (150.1.2.2)
      Origin IGP, metric 0, localpref 5000, valid, internal, best
      Community: no-export
  200
    173.1.125.5 from 173.1.125.5 (150.1.5.5)
      Origin IGP, localpref 100, valid, external
      Community: no-export

Rack1R1#show ip bgp 150.1.6.0
BGP routing table entry for 150.1.6.0/24, version 59
Paths: (2 available, best #1, table Default-IP-Routing-Table, not
advertised to EBGP peer)
  Not advertised to any peer
  200
    173.1.32.2 (metric 128) from 173.1.32.2 (150.1.2.2)
      Origin IGP, metric 0, localpref 5000, valid, internal, best
      Community: no-export
  200
    173.1.125.5 from 173.1.125.5 (150.1.5.5)
      Origin IGP, localpref 100, valid, external
      Community: no-export
```


Task 5.1 Solution

R4:

```
interface Tunnel47
 ip unnumbered FastEthernet0/0
 ip pim dense-mode
 tunnel source Loopback0
 tunnel destination 150.1.7.7
!
!
ip mroute 0.0.0.0 0.0.0.0 Tunnel47
```

SW1:

```
interface Tunnel47
 ip unnumbered Vlan73
 ip pim dense-mode
 tunnel source Loopback0
 tunnel destination 150.1.4.4
```

Task 5.1 Verification

```
Rack1R4#show ip pim interface

Address          Interface      Ver/   Nbr    Query  DR     DR
                                Mode   Count  Intvl  Prior
173.1.46.4       FastEthernet0/0    v2/D   0      30     1
173.1.46.4
173.1.46.4       Tunnel47       v2/D   1      30     1      0.0.0.0

Rack1SW1#show ip pim interface

Address          Interface      Ver/   Nbr    Query  DR     DR
                                Mode   Count  Intvl  Prior
204.12.1.7       Tunnel47       v2/D   1      30     1      0.0.0.0
204.12.1.7       Vlan73         v2/D   0      30     1      204.12.1.7
```



```
Rack1R4#show ip pim neighbor
PIM Neighbor Table
Mode: B - Bidir Capable, DR - Designated Router, N - Default DR
Priority,
      S - State Refresh Capable
Neighbor          Interface      Uptime/Expires    Ver   DR
Address                                                  Prio/Mode
204.12.1.7        Tunnel47       00:18:09/00:01:21 v2    1 / S
Rack1R4#show ip mroute static
Mroute: 0.0.0.0/0, interface: Tunnel47
  Protocol: none, distance: 0, route-map: none
```

Task 5.2 Solution

R4:

```
interface FastEthernet0/0
 ip igmp join-group 227.69.53.7
```

Task 5.2 Verification

```
Rack1SW1#debug ip icmp
ICMP packet debugging is on

Rack1SW1#ping 227.69.53.7

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 227.69.53.7, timeout is 2 seconds:

03:14:20: ICMP: echo reply rcvd, src 173.1.46.4, dst 204.12.1.7
Reply to request 0 from 173.1.46.4, 104 ms
```

```
Rack1R4#show ip igmp groups fastEthernet 0/0
IGMP Connected Group Membership
Group Address    Interface          Uptime    Expires   Last Reporter
Group Accounted
227.69.53.7      FastEthernet0/0    00:01:22  00:02:07  173.1.46.4
224.0.1.40       FastEthernet0/0    22:21:59  00:02:09  173.1.46.4
```

Task 5.3 Solution

SW1:

```
interface Tunnel47
 ip access-group MULTICAST out
!
ip access-list extended MULTICAST
 permit ip any host 227.69.53.7
 deny   ip any 224.0.0.0 15.255.255.255
 permit ip any any
```


Task 6.1 Solution

R6:

```
interface Serial0/0
 ip access-group 1 in
!
access-list 1 deny 200.0.1.2 0.0.2.24
access-list 1 permit any
```

Task 6.2 Solution

R5:

```
interface FastEthernet0/1
 rate-limit output access-group 192 496000 4000 4000 conform-action
transmit exceed-action drop
!
access-list 192 permit tcp any 173.1.5.0 0.0.0.255 eq www syn
```

Task 6.2 Verification

```
Rack1R5#show interfaces FastEthernet0/1 rate-limit
FastEthernet0/1
  Output
    matches: access-group 192
      params:  496000 bps, 4000 limit, 4000 extended limit
      conformed 0 packets, 0 bytes; action: transmit
      exceeded 0 packets, 0 bytes; action: drop
      last packet: 12704588ms ago, current burst: 0 bytes
      last cleared 00:00:35 ago, conformed 0 bps, exceeded 0 bps
```

Task 6.3 Solution

R4:

```
!
! The configuration below has been simply copied from Cisco's NAC
! documentation page example and adapted to the scenario.
!
aaa new-model
!
aaa authentication eou default group radius
!
ip admission name CISCO eapoudp inactivity-time 60
!
interface FastEthernet0/1.44
 ip access-group 102 in
 ip admission CISCO
!
ip radius source-interface Loopback0
!
radius-server host 173.1.137.252 auth-port 1645 acct-port 1646 key
CISCO
radius-server key CISCO
!
access-list 102 permit udp any any eq 21862
access-list 102 deny   ip any any
```


Task 6.3 Verification

```
Rack1R4#show ip admission configuration

Authentication Proxy Banner not configured
Consent Banner is not configured
Authentication global cache time is 60 minutes
Authentication global absolute time is 0 minutes
Authentication global init state time is 2 minutes
Authentication Proxy Session ratelimit is 100
Authentication Proxy Watch-list is disabled

Authentication Proxy Max HTTP process is 7
Authentication Proxy Auditing is disabled
Max Login attempts per user is 30

Authentication Proxy Rule Configuration
 Auth-proxy name CISCO
    eapoudp list not specified inactivity-timer 60 minute

Rack1R4#show aaa method-lists authentication
authen queue=AAA_ML_AUTHEN_LOGIN
authen queue=AAA_ML_AUTHEN_ENABLE
authen queue=AAA_ML_AUTHEN_PPP
authen queue=AAA_ML_AUTHEN_SGBP
authen queue=AAA_ML_AUTHEN_ARAP
authen queue=AAA_ML_AUTHEN_DOT1X
authen queue=AAA_ML_AUTHEN_EAPOUDP
  name=default valid=TRUE id=0 :state=ALIVE : SERVER_GROUP radius
authen queue=AAA_ML_AUTHEN_8021X
permanent lists
  name=Permanent Enable None valid=TRUE id=0 :state=ALIVE : ENABLE
NONE
  name=Permanent Enable valid=TRUE id=0 :state=ALIVE : ENABLE
  name=Permanent None valid=TRUE id=0 :state=ALIVE : NONE
  name=Permanent Local valid=TRUE id=0 :state=ALIVE : LOCAL
```



```
Rack1R4#show aaa servers

RADIUS: id 1, priority 1, host 173.1.137.252, auth-port 1645, acct-port
1646
     State: current UP, duration 86s, previous duration 0s
     Dead: total time 0s, count 0
     Quarantined: No
     Authen: request 0, timeouts 0, failover 0, retransmission 0
             Response: accept 0, reject 0, challenge 0
             Response: unexpected 0, server error 0, incorrect 0, time
0ms
             Transaction: success 0, failure 0
             Throttled: transaction 0, timeout 0, failure 0
     Author: request 0, timeouts 0, failover 0, retransmission 0
             Response: accept 0, reject 0, challenge 0
             Response: unexpected 0, server error 0, incorrect 0, time
0ms
             Transaction: success 0, failure 0
             Throttled: transaction 0, timeout 0, failure 0
     Account: request 0, timeouts 0, failover 0, retransmission 0
             Request: start 0, interim 0, stop 0
             Response: start 0, interim 0, stop 0
             Response: unexpected 0, server error 0, incorrect 0, time
0ms
             Transaction: success 0, failure 0
             Throttled: transaction 0, timeout 0, failure 0
     Elapsed time since counters last cleared: 1m
     Estimated Outstanding Access Transactions: 0
     Estimated Outstanding Accounting Transactions: 0
     Estimated Throttled Access Transactions: 0
     Estimated Throttled Accounting Transactions: 0
     Maximum Throttled Transactions: access 0, accounting 0
```


Task 6.4 Solution

R1:

```
username Rack1R3 password 0 CISCO
!
interface Serial0/1
 encapsulation ppp
 ppp chap hostname CHAPUSER
```

R2:

```
username Rack1R3 password 0 CISCO
!
interface Serial0/1
 encapsulation ppp
 ppp chap hostname CHAPUSER
```

R3:

```
username CHAPUSER password 0 CISCO
!
interface Serial1/2
 encapsulation ppp
 clockrate 64000
 ppp authentication chap
!
interface Serial1/3
 encapsulation ppp
 clockrate 64000
 ppp authentication chap
```


Task 6.4 Verification

```
Rack1R3#ping 173.1.32.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 173.1.32.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/29/32 ms

Rack1R3#ping 173.1.13.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 173.1.13.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/59/64 ms

Rack1R3#debug ppp authentication
PPP authentication debugging is on
Rack1R3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#interface Serial1/2
Rack1R3(config-if)#shutdown
Rack1R3(config-if)#no shutdown

%LINK-5-CHANGED: Interface Serial1/2, changed state to administratively
down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/2, changed
state to down
Se1/2 PPP: Using default call direction
Se1/2 PPP: Treating connection as a dedicated
Se1/2 PPP: Session handle[96000004] Session id[1]
Se1/2 PPP: Authorization required
%LINK-3-UPDOWN: Interface Serial1/2, changed state to up
Se1/2 CHAP: O CHALLENGE id 1 len 28 from "Rack1R3"
Se1/2 CHAP: I RESPONSE id 1 len 29 from "CHAPUSER"
Se1/2 PPP: Sent CHAP LOGIN Request
Se1/2 PPP: Received LOGIN Response PASS
Se1/2 PPP: Sent LCP AUTHOR Request
Se1/2 PPP: Sent IPCP AUTHOR Request
Se1/2 LCP: Received AAA AUTHOR Response PASS
Se1/2 IPCP: Received AAA AUTHOR Response PASS
Se1/2 CHAP: O SUCCESS id 1 len 4
Se1/2 PPP: Sent CDPCP AUTHOR Request
Se1/2 CDPCP: Received AAA AUTHOR Response PASS
Se1/2 PPP: Sent IPCP AUTHOR Request
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/2, changed
state to up
```


Task 6.5 Solution

SW1:

```
interface FastEthernet0/10
switchport protected
!
interface FastEthernet0/11
switchport protected
```

Task 6.5 Verification

To verify this temporarily put ports Fa0/1 and Fa0/3 on SW1 into protected mode. Before doing this, confirm connectivity between R1 and R3:

```
Rack1R1#ping 173.1.137.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 173.1.137.3, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 1/2/4 ms
```

Implement protected ports:

```
SW1:
!
interface range FastEthernet 0/1 , FastEthernet 0/3
 switchport protected
```

Verify new configuration:

```
Rack1R1#ping 173.1.137.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 173.1.137.3, timeout is 2 seconds:
...
Success rate is 0 percent (0/5)
```

Confirm that you still can ping SW1:

```
Rack1R1#ping 173.1.137.7

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 173.1.137.7, timeout is 2 seconds:
..!!!
Success rate is 60 percent (3/5), round-trip min/avg/max = 1/2/4 ms
```


Task 6.6 Solution

```
SW1:
interface FastEthernet0/10
 switchport block unicast
 switchport block multicast
!
interface FastEthernet0/11
 switchport block unicast
 switchport block multicast
```

Task 6.6 Breakdown

Port protection, as configured in the previous task, is designed to prevent devices in the same broadcast domain (VLAN) from directly exchanging traffic. Typically this configuration is used in a DMZ environment to prevent compromised devices from attacking the network from within. However there is a security flaw inherent to port protection that should not be overlooked. This flaw relates to the default processing behavior of unknown unicast and multicast MAC addresses.

When a switch receives a unicast or multicast frame it looks in the CAM table to pair the destination MAC address of the frame with the outgoing port. If there is not an entry in the CAM table the frame is treated as a broadcast frame and is flooded out all ports in the VLAN except that which it was received on. This mechanism is used to assist in discovering new hosts which have not previously sent traffic into the switch block, or those which have CAM entries which have timed out. By flooding the frame to all ports in the broadcast domain it can be reasonably assumed that the destination device will respond and the switch will be able to learn the outgoing interface for said device. In the case of port protection this behavior may not be desirable.

By sending traffic to random destination unicast and multicast MAC addresses an attacker can force a switch to flood the traffic out all interfaces. In the case that this traffic is received on a protected port the resulting behavior will be to flood the traffic out all ports in the VLAN, even to those that are protected. Since the ultimate goal of port protection is to prevent ports from communicating with each other this behavior is not acceptable. By issuing the **`switchport block unicast and switchport block multicast`** interface level commands these unknown unicast and multicast frames will not be forwarded out the interfaces they are configured on.

Task 6.6 Verification

```
Rack1SW1#show interfaces f0/10 switchport
Name: Fa0/10
Switchport: Enabled
Administrative Mode: dynamic desirable
Operational Mode: static access
Administrative Trunking Encapsulation: negotiate
Operational Trunking Encapsulation: native
Negotiation of Trunking: On
Access Mode VLAN: 4 (VLAN0004)
Trunking Native Mode VLAN: 1 (default)
Administrative Native VLAN tagging: enabled
Voice VLAN: none
Administrative private-vlan host-association: none
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
Trunking VLANs Enabled: ALL
Pruning VLANs Enabled: 2-1001
Capture Mode Disabled
Capture VLANs Allowed: ALL

Protected: true
Unknown unicast blocked: enabled
Unknown multicast blocked: enabled
Appliance trust: none
```


Task 6.7 Solution

SW2:

```
interface FastEthernet0/22
 switchport mode access
 switchport port-security
 switchport port-security maximum 5
 switchport port-security aging time 5
 switchport port-security violation protect
 switchport port-security aging type inactivity
!
interface FastEthernet0/23
 switchport mode access
 switchport port-security
 switchport port-security maximum 5
 switchport port-security aging time 5
 switchport port-security violation protect
 switchport port-security aging type inactivity
```

Task 6.7 Breakdown

Previous configurations have addressed the issue of limiting the amount of hosts that can access the network through a single switchport. However these configurations have not addressed the issue of port-security aging.

The above task describes a situation in which a maximum of five hosts are allowed to access the network through a specific port. Once traffic is received from a host it is added to the secure MAC address list. Once there are five addresses in the secure list traffic from all other hosts is dropped. This has been accomplished by issuing the **`maximum 5`** and **`violation protect`** port-security options. The issue with this configuration however is that once these addresses are learned they are not aged out of the table.
In order to ensure that inactive hosts are not taking up space in the secure MAC list the **`aging type inactivity`** and **`aging time 5`** options have been added to the above port-security configuration. These commands indicate that a MAC address in the secure list will expire once it has been inactive for more than five minutes.

Task 6.7 Verification

```
Rack1SW2#show port-security interface fa0/22
Port Security              : Enabled
Port Status                : Secure-down
Violation Mode             : Protect
Aging Time                 : 5 mins
Aging Type                 : Inactivity
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 5
Total MAC Addresses        : 0
Configured MAC Addresses   : 0
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 0000.0000.0000:0
Security Violation Count   : 0

Rack1SW2#show port-security interface fa0/23
Port Security              : Enabled
Port Status                : Secure-down
Violation Mode             : Protect
Aging Time                 : 5 mins
Aging Type                 : Inactivity
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 5
Total MAC Addresses        : 0
Configured MAC Addresses   : 0
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 0000.0000.0000:0
Security Violation Count   : 0
```


Task 6.8 Solution

R4:

```
zone security WAN-Serial
 description Covers S0/0/0 and S0/1/0
exit
!
zone security WAN-Ethernet
 description Covers Fa0/0
exit
!
zone security VLAN4
 description Covers VLAN 4 (Fa0/1)
exit
!
interface Serial0/0/0
 zone-member security WAN-Serial
!
interface Serial0/1/0
 zone-member security WAN-Serial
!
interface fastEthernet 0/0
 zone-member security WAN-Ethernet
!
interface fastEthernet 0/1
 zone-member security VLAN4
!
class-map type inspect VLAN4Web
 match protocol http
!
access-list 178 permit ip any host 150.1.5.5
```



```
class-map type inspect match-all VLAN4FTP
 match protocol ftp
 match access-group 178
!
class-map type inspect http DeepWeb
 match request uri length gt 222
!
policy-map type inspect http DeepWeb
 class type inspect http DeepWeb
  reset
!
policy-map type inspect Serial
 class VLAN4Web
  inspect
 class VLAN4FTP
  inspect
!
policy-map type inspect Ethernet
 class VLAN4Web
  inspect
  service-policy http DeepWeb
!
zone-pair security OutSerial source VLAN4 destination WAN-Serial
 service-policy type inspect Serial
!
zone-pair security OutEthernet source VLAN4 destination WAN-Ethernet
 service-policy type inspect Ethernet
```


Task 6.8 Verification

```
Rack1R4#show zone security
zone self
  Description: System defined zone


zone WAN-Serial
  Description: Covers S0/0/0 and S0/1/0
  Member Interfaces:
    Serial0/0/0
    Serial0/1/0


zone WAN-Ethernet
  Description: Covers Fa0/0
  Member Interfaces:
    FastEthernet0/0


zone VLAN4
  Description: Covers VLAN 4 (Fa0/1)
  Member Interfaces:
    FastEthernet0/1
```



```
Rack1R4#show zone-pair security
Zone-pair name OutSerial
    Source-Zone VLAN4  Destination-Zone WAN-Serial
    service-policy Serial
Zone-pair name OutEthernet
    Source-Zone VLAN4  Destination-Zone WAN-Ethernet
    service-policy Ethernet

Rack1R4#show policy-map type inspect zone-pair

policy exists on zp OutSerial
 Zone-pair: OutSerial

  Service-policy inspect : Serial

    Class-map: VLAN4Web (match-all)
      Match: protocol http

   Inspect
        Session creations since subsystem startup or last reset 0
        Current session counts (estab/half-open/terminating) [0:0:0]
        Maxever session counts (estab/half-open/terminating) [0:0:0]
        Last session created never
        Last statistic reset never
        Last session creation rate 0
        Maxever session creation rate 0
        Last half-open session total 0

    Class-map: VLAN4FTP (match-all)
      Match: protocol ftp
      Match: access-group 178

   Inspect
        Session creations since subsystem startup or last reset 0
        Current session counts (estab/half-open/terminating) [0:0:0]
        Maxever session counts (estab/half-open/terminating) [0:0:0]
        Last session created never
        Last statistic reset never
        Last session creation rate 0
        Maxever session creation rate 0
        Last half-open session total 0

    Class-map: class-default (match-any)
      Match: any
      Drop
        0 packets, 0 bytes
```



```
policy exists on zp OutEthernet
 Zone-pair: OutEthernet

  Service-policy inspect : Ethernet

    Class-map: VLAN4Web (match-all)
      Match: protocol http

   Inspect
        Session creations since subsystem startup or last reset 0
        Current session counts (estab/half-open/terminating) [0:0:0]
        Maxever session counts (estab/half-open/terminating) [0:0:0]
        Last session created never
        Last statistic reset never
        Last session creation rate 0
        Maxever session creation rate 0
        Last half-open session total 0

    Class-map: class-default (match-any)
      Match: any
      Drop
        0 packets, 0 bytes
Rack1R4#
```

Task 6.9 Solution

R4:

```
access-list 179 permit ip any any
!
class-map type inspect All
 match access-group 179
!
policy-map type inspect PermitAll
  class All
    pass
!
zone-pair security WAN-WAN1 source WAN-Serial destination WAN-Ethernet
  service-policy type inspect PermitAll
!
zone-pair security WAN-WAN2 source WAN-Ethernet destination WAN-Serial
  service-policy type inspect PermitAll
```


Task 6.9 Verification

```
Rack1R4(config-sec-zone-pair)#do sh zone-pair sec
Zone-pair name OutSerial
    Source-Zone VLAN4  Destination-Zone WAN-Serial
    service-policy Serial
Zone-pair name OutEthernet
    Source-Zone VLAN4  Destination-Zone WAN-Ethernet
    service-policy Ethernet
Zone-pair name WAN-WAN1
    Source-Zone WAN-Serial  Destination-Zone WAN-Ethernet
    service-policy PermitAll
Zone-pair name WAN-WAN2
    Source-Zone WAN-Ethernet  Destination-Zone WAN-Serial
    service-policy PermitAll

Rack1R4#show policy-map type inspect zone-pair WAN-WAN1

policy exists on zp WAN-WAN1
 Zone-pair: WAN-WAN1

  Service-policy inspect : PermitAll

    Class-map: All (match-all)
      Match: access-group 179
      Pass
        0 packets, 0 bytes

    Class-map: class-default (match-any)
      Match: any
      Drop
        0 packets, 0 bytes

Rack1R4#show policy-map type inspect zone-pair WAN-WAN2

policy exists on zp WAN-WAN2
 Zone-pair: WAN-WAN2

  Service-policy inspect : PermitAll

    Class-map: All (match-all)
      Match: access-group 179
      Pass
        0 packets, 0 bytes

    Class-map: class-default (match-any)
      Match: any
      Drop
        0 packets, 0 bytes
```


Task 7.1 Solution

R4:

```
logging rate-limit 10
```

Task 7.2 Solution

SW1:

```
ip access-list standard TELNET
 permit 173.1.0.0 0.0.255.255
 permit 150.1.0.0 0.0.15.255
 permit any log
!
logging file flash:LOCAL_LOGGING.TXT informational
!
line vty 0 15
 access-class TELNET in
```

Task 7.2 Verification

Telnet to SW1 from BB3:

```
BB3>telnet 204.12.1.7
Trying 204.12.1.7 ... Open


User Access Verification

Password: <cisco>
Rack1SW1>en
Password: <cisco>

Rack1SW1#more flash:LOCAL_LOGGING.TXT | inc SEC
03:37:41: %SEC-6-IPACCESSLOGS: list TELNET permitted 204.12.1.254 1
packet
```


Task 7.3 Solution

R5:

```
interface FastEthernet0/0
 ip nat outside
!
interface Serial0/0/0.54
 ip nat inside
!
interface Serial0/0/0.125
 ip nat inside
!
interface Serial0/1/0
 ip nat inside

!
interface FastEthernet0/1
 ip nat inside
!
interface Loopback0
 ip nat inside
!
ip nat inside source list INTERNAL_NETWORK interface FastEthernet0/0
overload
!
ip access-list standard INTERNAL_NETWORK
 permit 173.1.0.0 0.0.255.255
 permit 150.1.0.0 0.0.15.255
```


Task 7.3 Verification

```
Rack1R5#show ip nat statistics
Total active translations: 0 (0 static, 0 dynamic; 0 extended)
Outside interfaces:
  FastEthernet0/0
Inside interfaces:
  Serial0/0.54, Serial0/0.125, FastEthernet0/1, Loopback0
Hits: 0  Misses: 0
CEF Translated packets: 0, CEF Punted packets: 0
Expired translations: 0
Dynamic mappings:
-- Inside Source
[Id: 1] access-list INTERNAL_NETWORK interface FastEthernet0/0 refcount
0
Queued Packets: 0

Rack1R4#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 = 60/62/64 ms

Rack1R5#show ip nat translations
Pro Inside global      Inside local   Outside local      Outside global
icmp 192.10.1.5:4      173.1.54.4:4   192.10.1.254:4     192.10.1.254:4
```

Task 7.4 Solution

R5:

```
ip nat inside source static tcp 173.1.5.100 25 192.10.1.5 25
ip nat inside source static tcp 173.1.5.100 80 192.10.1.5 80
ip nat inside source static tcp 173.1.5.100 443 192.10.1.5 443
ip nat inside source static tcp 173.1.5.100 110 192.10.1.5 110
```

Task 7.4 Verification

```
Rack1R5#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside
global
tcp 192.10.1.5:25      173.1.5.100:25     ---                ---
tcp 192.10.1.5:80      173.1.5.100:80     ---                ---
tcp 192.10.1.5:110     173.1.5.100:110    ---                ---
tcp 192.10.1.5:443     173.1.5.100:443    ---                ---
```


Task 7.5 Solution

R1:

```
ip domain-name cisco.com
crypto key generate rsa general-keys modulus 1024
username CISCO password CISCO
!
ip ssh port 2002 rotary 1
ip ssh logging events
!
line aux 0
 rotary 1
  transport input ssh
  login local
```

Task 7.6 Solution

R5:

```
ip traffic-export profile R5-WAN
  interface fastEthernet 0/1
  bidirectional
  mac-address 0010.1731.5100
  incoming sample one-in-every 10
  outgoing sample one-in-every 10
!
interface Serial0/0/0.54
 ip traffic-export apply R5-WAN
!
interface Serial0/0/0.125
 up traffic-export apply R5-WAN
!
interface Serial0/1/0
 up traffic-export apply R5-WAN
```


Task 7.6 Verification

R5:

```
%RITE-5-ACTIVATE: Activated IP traffic export on interface Serial0/0/0
%RITE-5-ACTIVATE: Activated IP traffic export on interface Serial0/1/0

Rack1R5#show ip traffic-export
Router IP Traffic Export Parameters
Monitored Interface             Serial0/0/0
        Export Interface                FastEthernet0/1
        Destination MAC address 0010.1731.5100
        bi-directional traffic export is on
Output IP Traffic Export Information    Packets/Bytes Exported    4/218
        Packets Dropped           40
        Sampling Rate             one-in-every 10 packets
        No Access List configured
Input IP Traffic Export Information     Packets/Bytes Exported    2/99
        Packets Dropped           22
        Sampling Rate             one-in-every 10 packets
        No Access List configured
        Profile R5-WAN is Active

Monitored Interface             Serial0/1/0
        Export Interface                FastEthernet0/1
        Destination MAC address 0010.1731.5100
        bi-directional traffic export is on
Output IP Traffic Export Information    Packets/Bytes Exported    0/0
Packets Dropped
        Sampling Rate             one-in-every 10 packets
        No Access List configured
Input IP Traffic Export Information     Packets/Bytes Exported    0/0
        Packets Dropped           0
        Sampling Rate             one-in-every 10 packets
        No Access List configured
        Profile R5-WAN is Active
```

Task 8.1 Solution

R5:

```
ip cef
!
interface FastEthernet0/0
 ip nbar protocol-discovery
!
interface FastEthernet0/1
 ip nbar protocol-discovery
```


Task 8.1 Verification

```
Rack1R5#show ip nbar protocol-discovery top-n 3

 FastEthernet0/0
                Input                     Output
                -----                     ------
   Protocol     Packet Count              Packet Count
                Byte Count                Byte Count
                5min Bit Rate (bps)       5min Bit Rate (bps)
                5min Max Bit Rate (bps)   5min Max Bit Rate (bps)
   ------------ ------------------------- -----------------
   icmp         10                        5
                1140                      570
                0                         0
                0                         0
   bgp          2                         1
                167                       93
                0                         0
                0                         0
   citrix       0                         0
                0                         0
                0                         0
                0                         0
   unknown      0                         0
                0                         0
                0                         0
                0                         0
   Total        12                        6
                1307                      663
                0                         0
                0                         0
```



```
 FastEthernet0/1
                Input                     Output
                -----                     ------
   Protocol     Packet Count              Packet Count
                Byte Count                Byte Count
                5min Bit Rate (bps)       5min Bit Rate (bps)
                5min Max Bit Rate (bps)   5min Max Bit Rate (bps)
   ------------ ------------------------- -----------------
   eigrp        0                         14
                0                         1036
                0                         0
                0                         0
   bgp          0                         0
                0                         0
                0                         0
                0                         0
   citrix       0                         0
                0                         0
                0                         0
                0                         0
   unknown      0                         0
                0                         0
                0                         0
                0                         0
   Total        0                         14
                0                         1036
                0                         0
                0                         0
```

Task 8.2 Solution

R5:

```
class-map match-any PEER_TO_PEER
  match protocol kazaa2
  match protocol fasttrack
  match protocol gnutella
  !
 policy-map DROP_PEER_TO_PEER_VLAN5
  class PEER_TO_PEER
   drop
!
policy-map DROP_PEER_TO_PEER_VLAN52_IN
  class PEER_TO_PEER
   drop
!
policy-map DROP_PEER_TO_PEER_VLAN52_OUT
  class PEER_TO_PEER
   drop
!
interface FastEthernet0/0
 service-policy input DROP_PEER_TO_PEER_VLAN52_IN
 service-policy output DROP_PEER_TO_PEER_VLAN52_OUT
!
interface FastEthernet0/1
 service-policy input DROP_PEER_TO_PEER_VLAN5
 service-policy output DROP_PEER_TO_PEER_VLAN5
```


Task 8.2 Verification

```
Rack1R5#show policy-map interface FastEthernet 0/0
 FastEthernet0/0

  Service-policy input: DROP_PEER_TO_PEER

    Class-map: PEER_TO_PEER (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol kazaa2
        0 packets, 0 bytes
        5 minute rate 0 bps
      Match: protocol fasttrack
        0 packets, 0 bytes
        5 minute rate 0 bps
      Match: protocol gnutella
        0 packets, 0 bytes
        5 minute rate 0 bps
      Match: protocol napster
        0 packets, 0 bytes
        5 minute rate 0 bps
      drop

    Class-map: class-default (match-any)
      3 packets, 345 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

  Service-policy output: DROP_PEER_TO_PEER

    Class-map: PEER_TO_PEER (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol kazaa2
        0 packets, 0 bytes
        5 minute rate 0 bps
      Match: protocol fasttrack
        0 packets, 0 bytes
        5 minute rate 0 bps
      Match: protocol gnutella
        0 packets, 0 bytes
        5 minute rate 0 bps
      Match: protocol napster
        0 packets, 0 bytes
        5 minute rate 0 bps
      drop

    Class-map: class-default (match-any)
      8 packets, 619 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```


Task 8.3 Solution

R5:

```
policy-map DROP_PEER_TO_PEER_VLAN52_IN
  class class-default
   police cir 512000 bc 1920 pir 896000 be 3360
   conform-action transmit exceed-action set-dscp-transmit 0
   violate-action drop
```

Task 8.3 Verification

```
Rack1R5#show policy-map interface fastEthernet 0/0 input
 FastEthernet0/0

  Service-policy input: DROP_PEER_TO_PEER_VLAN52_IN

    Class-map: PEER_TO_PEER (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol kazaa2
        0 packets, 0 bytes
        5 minute rate 0 bps
      Match: protocol fasttrack
        0 packets, 0 bytes
        5 minute rate 0 bps
      Match: protocol gnutella
        0 packets, 0 bytes
        5 minute rate 0 bps
      drop

    Class-map: class-default (match-any)
      324 packets, 29160 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      police:
          cir 512000 bps, bc 1920 bytes
          pir 896000 bps, be 3360 bytes
        conformed 137 packets, 12330 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          set-dscp-transmit default
        violated 0 packets, 0 bytes; actions:
          drop
        conformed 0 bps, exceed 0 bps, violate 0 bps
```


Task 8.4 Solution

R5:

```
class-map match-any DLCI_501
 match fr-dlci 501
!
class-map VOICE
 match dscp EF
!
policy-map CBWFQ_DLCI_501
 class VOICE
  priority 256
 class class-default
  fair-queue
!
policy-map SHAPE_DLCI_501
 class DLCI_501
  shape average 512000
  service-policy CBWFQ_DLCI_501
!
interface Serial 0/0
 service-policy output SHAPE_DLCI_501
```


Task 8.4 Verification

```
Rack1R5#show policy-map interface serial 0/0/0.125

 Serial0/0/0.125

  Service-policy output: SHAPE_DLCI_501

    Class-map: DLCI_501 (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: fr-dlci 501
        0 packets, 0 bytes
        5 minute rate 0 bps
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 0/0
      shape (average) cir 512000, bc 2048, be 2048
      target shape rate 512000
        lower bound cir 0,  adapt to fecn 0

      Service-policy : CBWFQ_DLCI_501

        queue stats for all priority classes:

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

        Class-map: VOICE (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match:  dscp ef (46)
          Priority: 256 kbps, burst bytes 6400, b/w exceed drops: 0


        Class-map: class-default (match-any)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: any
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops/flowdrops) 0/0/0/0
          (pkts output/bytes output) 0/0
          Fair-queue: per-flow queue limit 16
          QoS Set
            fr-de
              Packets marked 0

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 0/0
```

Task 8.5 Solution

R5:

```
class-map VOIP
 match dscp EF
!
policy-map LLQ
 class VOIP
  priority 64
 class class-default
 fair-queue

policy-map SHAPE_256K
 class class-default
  shape average 256000 2560
  shape adaptive 128000
  shape fr-voice-adapt
  service-policy LLQ
!
map-class frame-relay TO_R4
 service-policy output SHAPE_256K
!

interface Serial 0/0/0.54
 frame-relay interface-dlci 504
  class TO_R4
!
interface Serial 0/0/0
 frame-relay fragment 640 end-to-end
 frame-relay fragmentation voice-adaptive
```


R4:

```
class-map VOIP
 match dscp EF
!
policy-map LLQ
 class VOIP
  priority 64
 class class-default
 fair-queue

policy-map SHAPE_256K
 class class-default
  shape average 256000 2560
  shape adaptive 128000
  shape fr-voice-adapt
  service-policy LLQ
!
map-class frame-relay TO_R5
 service-policy output SHAPE_256K
!
interface Serial 0/0/0
 frame-relay interface-dlci 405
  class TO_R5
!
interface Serial 0/0/0
 frame-relay fragment 640 end-to-end
 frame-relay fragmentation voice-adaptive
```


Task 8.5 Verification

```
Rack1R5#show policy-map interface serial 0/0/0.54
 Serial0/0/0.54: DLCI 504 -

  Service-policy output: SHAPE_256K

    Class-map: class-default (match-any)
      669 packets, 69566 bytes
      5 minute offered rate 7000 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 669/69566
      shape (average) cir 256000, bc 2560, be 2560
      target shape rate 256000
        lower bound cir 128000,  adapt to fecn 0
        Voice Adaptive Shaping inactive

      Service-policy : LLQ

        queue stats for all priority classes:

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

        Class-map: VOIP (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match:  dscp ef (46)
          Priority: 64 kbps, burst bytes 1600, b/w exceed drops: 0


        Class-map: class-default (match-any)
          669 packets, 69566 bytes
          5 minute offered rate 7000 bps, drop rate 0 bps
          Match: any
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops/flowdrops) 0/0/0/0
          (pkts output/bytes output) 669/69566
          Fair-queue: per-flow queue limit 16
```



```
Rack1R4#show policy-map interface serial 0/0/0
 Serial0/0/0: DLCI 405 -

  Service-policy output: SHAPE_256K

    Class-map: class-default (match-any)
      2222 packets, 228552 bytes
      5 minute offered rate 7000 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 2222/228552
      shape (average) cir 256000, bc 2560, be 2560
      target shape rate 256000
        lower bound cir 128000,  adapt to fecn 0
        Voice Adaptive Shaping inactive

      Service-policy : LLQ

        queue stats for all priority classes:

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

        Class-map: VOIP (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match:  dscp ef (46)
          Priority: 64 kbps, burst bytes 1600, b/w exceed drops: 0


        Class-map: class-default (match-any)
          2222 packets, 228552 bytes
          5 minute offered rate 7000 bps, drop rate 0 bps
          Match: any
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops/flowdrops) 0/0/0/0
          (pkts output/bytes output) 2222/228552
          Fair-queue: per-flow queue limit 16
```


Lab 18 Solutions

Task 2.1 Solution

R6:

```
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
!
! Note that when redistributing between EIGRP processes there is no
! need to configure specific or seed metric as it is transparent
!
router eigrp 10
 no auto-summary
 network 54.1.1.6 0.0.0.0
 redistribute eigrp 100
!
router eigrp 100
 redistribute eigrp 10
```

Task 2.1 Verification

Verify EIGRP authentication:

```
Rack1R6#show ip eigrp 10 interfaces detail
IP-EIGRP interfaces for process 10

                        Xmit Queue   Mean   Pacing Time   Multicast
Pending
Interface        Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer
Routes
Se0/0/0            1        0/0       896       0/15        5187
0
  Hello interval is 60 sec
  Next xmit serial <none>
  Un/reliable mcasts: 0/0  Un/reliable ucasts: 1/4
  Mcast exceptions: 0  CR packets: 0  ACKs suppressed: 1
  Retransmissions sent: 0  Out-of-sequence rcvd: 0
  Authentication mode is md5,  key-chain is "EIGRP"
  Use unicast
```

Check EIGRP neighbors:

```
Rack1R6#show ip eigrp 10 neighbors
IP-EIGRP neighbors for process 10
H   Address                 Interface    Hold Uptime   SRTT   RTO  Q  Seq
                                         (sec)         (ms)       Cnt Num
```

```
0   54.1.1.254              Se0/0/0          12 00:02:19  896  5000  0  2

Rack1R6#show ip eigrp 100 neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   156.1.67.7              Fa0/1             11 01:40:16   28   200  0  55
```

Check EIGRP routes:

```
Rack1R6# show ip route eigrp 10
D    200.0.0.0/24 [90/2297856] via 54.1.1.254, 00:03:50, Serial0/0/0
D    200.0.1.0/24 [90/2297856] via 54.1.1.254, 00:03:50, Serial0/0/0
D    200.0.2.0/24 [90/2297856] via 54.1.1.254, 00:03:50, Serial0/0/0
D    200.0.3.0/24 [90/2297856] via 54.1.1.254, 00:03:50, Serial0/0/0
```



```
Rack1R6# show ip route eigrp 100
D    192.10.1.0/24 [90/4735232] via 156.1.67.7, 01:34:41,
FastEthernet0/1
     156.1.0.0/16 is variably subnetted, 10 subnets, 2 masks
D       156.1.27.0/24 [90/28416] via 156.1.67.7, 01:41:59,
FastEthernet0/1
D       156.1.23.0/24 [90/2172672] via 156.1.67.7, 01:41:59,
FastEthernet0/1
D       156.1.18.0/24 [90/4735232] via 156.1.67.7, 01:41:59,
FastEthernet0/1
D       156.1.13.0/24 [90/4732672] via 156.1.67.7, 01:41:59,
FastEthernet0/1
D       156.1.3.0/24 [90/2175232] via 156.1.67.7, 01:41:59,
FastEthernet0/1
D       156.1.58.0/24 [90/4735232] via 156.1.67.7, 01:41:59,
FastEthernet0/1
D       156.1.45.0/24 [90/5244672] via 156.1.67.7, 01:41:59,
FastEthernet0/1
D       156.1.45.4/32 [90/5244672] via 156.1.67.7, 01:41:59,
FastEthernet0/1
D       156.1.35.0/24 [90/4732672] via 156.1.67.7, 01:41:59,
FastEthernet0/1
D    204.12.1.0/24 [90/4735232] via 156.1.67.7, 01:35:11,
FastEthernet0/1
     150.1.0.0/24 is subnetted, 7 subnets
D       150.1.7.0 [90/156160] via 156.1.67.7, 01:41:59, FastEthernet0/1
D       150.1.5.0 [90/4860672] via 156.1.67.7, 01:41:59,
FastEthernet0/1
D       150.1.3.0 [90/2300672] via 156.1.67.7, 01:41:59,
FastEthernet0/1
D       150.1.2.0 [90/156416] via 156.1.67.7, 01:41:59, FastEthernet0/1
D       150.1.1.0 [90/4860672] via 156.1.67.7, 01:41:59,
FastEthernet0/1
D       150.1.8.0 [90/4863232] via 156.1.67.7, 01:41:59,
FastEthernet0/1
```

Verify redistributiojn process; EIGRP AS 10 prefixes should appear in SW1 routing table:


```
Rack1SW1#show ip route eigrp | i EX
D EX 200.0.0.0/24 [170/2300416] via 156.1.67.6, 00:05:46,
FastEthernet0/21
D EX 200.0.1.0/24 [170/2300416] via 156.1.67.6, 00:05:46,
FastEthernet0/21
D EX 200.0.2.0/24 [170/2300416] via 156.1.67.6, 00:05:46,
FastEthernet0/21
D EX 200.0.3.0/24 [170/2300416] via 156.1.67.6, 00:05:46,
FastEthernet0/21
```

Verify key chain:

```
Rack1R6#show key chain
Key-chain EIGRP:
    key 1 -- text "CISCO"
        accept lifetime (always valid) - (always valid) [valid now]
        send lifetime (always valid) - (always valid) [valid now]
```

Task 2.2 Solution

R1:

```
!
interface FastEthernet0/0
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 5
```

R5:

```
!
interface FastEthernet0/1
 ip hello-interval eigrp 100 1
 ip hold-time eigrp 100 5
```

SW2:

```
!
router eigrp 100
 offset-list ODD_THIRD_OCTET in 111111111 Vlan18
 offset-list EVEN_THIRD_OCTET in 111111111 Vlan58
!
ip access-list standard EVEN_THIRD_OCTET
 permit 0.0.0.0 255.255.254.255
ip access-list standard ODD_THIRD_OCTET
 permit 0.0.1.0 255.255.254.255
```


Task 2.2 Verification

Check hello timer at R5 (output available in recent IOS versions):

```
Rack1R5#sho ip eigrp interfaces detail fastEthernet 0/1
IP-EIGRP interfaces for process 100

                        Xmit Queue   Mean   Pacing Time   Multicast
Pending
Interface        Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer
Routes
Fa0/1              1        0/0         1       0/1           50
0
  Hello interval is 1 sec
  Next xmit serial <none>
  Un/reliable mcasts: 0/15  Un/reliable ucasts: 25/14
  Mcast exceptions: 3  CR packets: 3  ACKs suppressed: 1
  Retransmissions sent: 2  Out-of-sequence rcvd: 0
  Authentication mode is not set
  Use multicast
```

Check paths to EIGRP prefixes with even third octet:

```
Rack1SW2#show ip route eigrp | include Vlan18
D    200.0.0.0/24 [90/23717376] via 156.1.18.1, 00:00:14, Vlan18
D    200.0.2.0/24 [90/23717376] via 156.1.18.1, 00:00:14, Vlan18
D       150.1.6.0 [90/23205376] via 156.1.18.1, 00:00:14, Vlan18
D       150.1.2.0 [90/23200256] via 156.1.18.1, 00:00:14, Vlan18
```

Check paths to EIGRP prefixes with odd third octect:

```
Rack1SW2#show ip route eigrp | include Vlan58
D    192.10.1.0/24 [90/28416] via 156.1.58.5, 02:08:29, Vlan58
D       54.1.1.0 [90/23589376] via 156.1.58.5, 00:05:54, Vlan58
D EX 200.0.1.0/24 [170/23717376] via 156.1.58.5, 00:05:54, Vlan58
D       156.1.27.0/24 [90/23074816] via 156.1.58.5, 00:05:54, Vlan58
D       156.1.23.0/24 [90/23072256] via 156.1.58.5, 00:05:54, Vlan58
D       156.1.13.0/24 [90/5145856] via 156.1.58.5, 00:05:53, Vlan58
D       156.1.3.0/24 [90/2588416] via 156.1.58.5, 00:05:54, Vlan58
D       156.1.45.0/24 [90/2170112] via 156.1.58.5, 02:16:14, Vlan58
D       156.1.45.4/32 [90/2170112] via 156.1.58.5, 02:16:15, Vlan58
D       156.1.35.0/24 [90/2585856] via 156.1.58.5, 02:16:15, Vlan58
D       156.1.67.0/24 [90/23077376] via 156.1.58.5, 00:05:54, Vlan58
D EX 200.0.3.0/24 [170/23717376] via 156.1.58.5, 00:05:54, Vlan58
D    204.12.1.0/24 [90/28416] via 156.1.58.5, 02:09:00, Vlan58
D       150.1.7.0 [90/23202816] via 156.1.58.5, 00:05:54, Vlan58
D       150.1.5.0 [90/130816] via 156.1.58.5, 02:16:15, Vlan58
D       150.1.3.0 [90/2713856] via 156.1.58.5, 00:05:54, Vlan58
D       150.1.1.0 [90/5273856] via 156.1.58.5, 00:05:55, Vlan58
```


Verify the hold time value for SW2's EIGRP neighbors:

```
Rack1SW2#show ip eigrp neighbors
EIGRP-IPv4:(100) neighbors for process 100
H   Address           Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                      (sec)         (ms)       Cnt Num
1   156.1.58.5        Vl58               4 02:18:11    3   200  0  54
0   156.1.18.1        Vl18               4 02:18:12    1   200  0  58
```

Task 2.3 Solution

R1:
```
interface Virtual-Template1
 ip bandwidth-percent eigrp 100 10
```

R3:
```
interface Virtual-Template1
 ip bandwidth-percent eigrp 100 10
!
interface Virtual-Template2
 ip bandwidth-percent eigrp 100 10
```

R5:
```
interface Virtual-Template1
 ip bandwidth-percent eigrp 100 10
```

Task 2.4 Solution

R1, R2, R3, R5, R6, SW1 and SW2:
```
router eigrp 10
 timers active-time 5
```

Task 2.5 Solution

R5:
```
router odr
!
router eigrp 100
redistribute odr metric 1500 1000 255 1 1500
```


Task 2.5 Verification

Verify CDP configuration:

```
Rack1R4#show cdp neighbors serial 0/1/0
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater

Device ID    Local Intrfce     Holdtme    Capability  Platform  Port ID
Rack1R5      Ser 0/1/0          129          R S I     1841      Ser 0/1/0

Rack1R4#show ip protocols

Rack1R4#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS
level-2
       ia - IS-IS inter area, * - candidate default, U - per-user
static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 156.1.45.5 to network 0.0.0.0

     156.1.0.0/16 is variably subnetted, 4 subnets, 2 masks
C       156.1.4.0/24 is directly connected, FastEthernet0/0
C       156.1.45.0/24 is directly connected, Serial0/1/0
C       156.1.44.0/24 is directly connected, FastEthernet0/1
C       156.1.45.5/32 is directly connected, Serial0/1/0
     150.1.0.0/24 is subnetted, 1 subnets
C       150.1.4.0 is directly connected, Loopback0
o*   0.0.0.0/0 [160/1] via 156.1.45.5, 00:00:43, Serial0/1/0

Rack1R4#debug cdp packets
CDP packet info debugging is on

Rack1R4#debug cdp events
CDP events debugging is on

Rack1R4#debug cdp ip
CDP IP info debugging is on
```



```
Rack1R4#
 CDP-EV: Unrecognized type (16) seen in TLV
 CDP-PA: Packet received from Rack1SW1 on interface FastEthernet0/0
 **Entry  found in cache**
 CDP-EV: Lookup for ip phone with idb= FastEthernet0/0 ip= 156.1.27.7
mac= 000f.8fe0.3504 platform= Cisco WS-C3550-24
 CDP-IP: Writing prefix 150.1.4.0/24
 CDP-IP: Writing prefix 156.1.45.0/24
 CDP-IP: Writing prefix 156.1.44.0/24
 CDP-PA: version 2 packet sent out on FastEthernet0/0
 CDP-IP: Writing prefix 150.1.4.0/24
 CDP-IP: Writing prefix 156.1.4.0/24
 CDP-IP: Writing prefix 156.1.45.0/24
 CDP-PA: version 2 packet sent out on FastEthernet0/1
 CDP-IP: Writing prefix 150.1.4.0/24
 CDP-IP: Writing prefix 156.1.4.0/24
 CDP-IP: Writing prefix 156.1.44.0/24
 CDP-PA: version 2 packet sent out on Serial0/1/0
 CDP-PA: Packet received from Rack1R5 on interface Serial0/1/0
 **Entry  found in cache**
 CDP-EV: Lookup for ip phone with idb= Serial0/1/0 ip= 156.1.45.5 mac=
0000.0000.0000 platform= Cisco 1841
CDP-IP: Reading default route 156.1.45.5 via Serial0/1/0
CDP-IP: Updating default route 156.1.45.5 in routing table

Rack1R5#show ip route odr
     156.1.0.0/16 is variably subnetted, 12 subnets, 2 masks
o       156.1.4.0/24 [160/1] via 156.1.45.4, 00:00:57, Serial0/1/0
o       156.1.44.0/24 [160/1] via 156.1.45.4, 00:00:57, Serial0/1/0
     150.1.0.0/24 is subnetted, 8 subnets
o       150.1.4.0 [160/1] via 156.1.45.4, 00:00:57, Serial0/1/0
```

Test connectivity between ODR/EIGRP domains:

```
Rack1R1#ping 150.1.4.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.4.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 116/119/120
ms
```

Task 2.6 Solution

R5:

```
router bgp 100
 neighbor 192.10.1.254 remote-as 254
 neighbor 192.10.1.254 local-as 200 no-prepend
 neighbor 192.10.1.254 password CISCO
```


Task 2.6 Verification

Verify BGP peering:

```
Rack1R5#show ip bgp neighbors 192.10.1.254
BGP neighbor is 192.10.1.254,  remote AS 254,  local AS 200 no-prepend,
external link
  BGP version 4, remote router ID 222.22.2.1
  BGP state = Established, up for 00:00:14
  Last read 00:00:14, last write 00:00:14, hold time is 180, keepalive
interval is 60 seconds
<output omitted>
```

Verify BGP routes:

```
Rack1R5#show ip bgp q _254$
BGP table version is 14, 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
*> 205.90.31.0      192.10.1.254             0             0 254 ?
*> 220.20.3.0       192.10.1.254             0             0 254 ?
*> 222.22.2.0       192.10.1.254             0             0 254 ?
```


Task 2.7 Solution

```
R5:
router bgp 100
 redistribute eigrp 100 route-map INTERNAL_TO_BGP
!
ip prefix-list INTERNAL seq 5 permit 156.1.0.0/16 le 32
ip prefix-list INTERNAL seq 10 permit 150.1.0.0/16 le 32
!
route-map INTERNAL_TO_BGP permit 10
match ip address prefix-list INTERNAL
```

```
R6:
router bgp 100
redistribute eigrp 100 route-map INTERNAL_TO_BGP

ip prefix-list INTERNAL seq 5 permit 156.1.0.0/16 le 32
ip prefix-list INTERNAL seq 10 permit 150.1.0.0/16 le 32
!
route-map INTERNAL_TO_BGP permit 10
match ip address prefix-list INTERNAL
```


Task 2.7 Verification

Verify BGP prefixes advertisement:

```
Rack1R6#show ip bgp q ^$
BGP table version is 68, 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
*> 150.1.1.0/24     156.1.67.7         4860672         32768 ?
* i                 150.1.5.5          5273600    100      0 ?
*> 150.1.2.0/24     156.1.67.7          156416         32768 ?
* i                 150.1.5.5         23200000    100      0 ?
*> 150.1.3.0/24     156.1.67.7         2300672         32768 ?
* i                 150.1.5.5          2713600    100      0 ?
*> 150.1.4.0/24     156.1.67.7         5037312         32768 ?
*> 150.1.5.0/24     156.1.67.7         4860672         32768 ?
* i                 150.1.5.5                0    100      0 ?
*> 150.1.6.0/24     0.0.0.0                  0         32768 ?
* i                 150.1.5.5         23205120    100      0 ?
*> 150.1.7.0/24     156.1.67.7          156160         32768 ?
* i                 150.1.5.5         23202560    100      0 ?
*> 150.1.8.0/24     156.1.67.7         4863232         32768 ?
* i                 150.1.5.5           409600    100      0 ?
*> 156.1.3.0/24     156.1.67.7         2198272         32768 ?
* i                 150.1.5.5          2841600    100      0 ?
*> 156.1.4.0/24     156.1.67.7         5037312         32768 ?
*> 156.1.8.0/24     156.1.67.7         4735488         32768 ?
* i                 150.1.5.5           281856    100      0 ?
*> 156.1.13.0/24    156.1.67.7         4732672         32768 ?
* i                 150.1.5.5          5145600    100      0 ?
*> 156.1.18.0/24    156.1.67.7         4735232         32768 ?
* i                 150.1.5.5           281856    100      0 ?
*> 156.1.23.0/24    156.1.67.7         2172672         32768 ?
* i                 150.1.5.5         23072000    100      0 ?
*> 156.1.27.0/24    156.1.67.7           28416         32768 ?
* i                 150.1.5.5         23074560    100      0 ?
*> 156.1.35.0/24    156.1.67.7         4732672         32768 ?
* i                 150.1.5.5                0    100      0 ?
*> 156.1.44.0/24    156.1.67.7         5037312         32768 ?
*> 156.1.45.0/24    156.1.67.7         5244672         32768 ?
* i                 150.1.5.5                0    100      0 ?
*> 156.1.58.0/24    156.1.67.7         4735488         32768 ?
* i                 150.1.5.5                0    100      0 ?
*> 156.1.67.0/24    0.0.0.0                  0         32768 ?
* i                 150.1.5.5         23077120    100      0 ?
```



```
Rack1R5#show ip bgp q ^$
BGP table version is 37, 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
* i150.1.1.0/24     150.1.6.6          4860672    100      0 ?
*>                  156.1.35.3         5273600             32768 ?
* i150.1.2.0/24     150.1.6.6           156416    100      0 ?
*>                  156.1.35.3        23200000             32768 ?
* i150.1.3.0/24     150.1.6.6          2300672    100      0 ?
*>                  156.1.35.3         2713600             32768 ?
r>i150.1.4.0/24     150.1.6.6          5037312    100      0 ?
* i150.1.5.0/24     150.1.6.6          4860672    100      0 ?
*>                  0.0.0.0                  0             32768 ?
* i150.1.6.0/24     150.1.6.6                0    100      0 ?
*>                  156.1.35.3        23205120             32768 ?
* i150.1.7.0/24     150.1.6.6           156160    100      0 ?
*>                  156.1.35.3        23202560             32768 ?
* i150.1.8.0/24     150.1.6.6          4863232    100      0 ?
*>                  156.1.58.8          409600             32768 ?
* i156.1.3.0/24     150.1.6.6          2198272    100      0 ?
*>                  156.1.35.3         2841600             32768 ?
r>i156.1.4.0/24     150.1.6.6          5037312    100      0 ?
* i156.1.8.0/24     150.1.6.6          4735488    100      0 ?
*>                  156.1.58.8          281856             32768 ?
* i156.1.13.0/24    150.1.6.6          4732672    100      0 ?
*>                  156.1.35.3         5145600             32768 ?
* i156.1.18.0/24    150.1.6.6          4735232    100      0 ?
*>                  156.1.58.8          281856             32768 ?
* i156.1.23.0/24    150.1.6.6          2172672    100      0 ?
*>                  156.1.35.3        23072000             32768 ?
* i156.1.27.0/24    150.1.6.6            28416    100      0 ?
*>                  156.1.35.3        23074560             32768 ?
* i156.1.35.0/24    150.1.6.6          4732672    100      0 ?
*>                  0.0.0.0                  0             32768 ?
r>i156.1.44.0/24    150.1.6.6          5037312    100      0 ?
* i156.1.45.0/24    150.1.6.6          5244672    100      0 ?
*>                  0.0.0.0                  0             32768 ?
* i156.1.58.0/24    150.1.6.6          4735488    100      0 ?
*>                  0.0.0.0                  0             32768 ?
* i156.1.67.0/24    150.1.6.6                0    100      0 ?
*>                  156.1.35.3        23077120             32768 ?
```


Task 2.8 Solution

R5:

```
interface FastEthernet0/1
 ip summary-address eigrp 100 0.0.0.0 0.0.0.0 5 leak-map LEAK
!
interface Virtual-Template1
 ip summary-address eigrp 100 0.0.0.0 0.0.0.0 5 leak-map LEAK
!
ip prefix-list BACKBONES seq 5 permit 192.10.1.0/24
ip prefix-list BACKBONES seq 10 permit 204.12.1.0/24
!
ip prefix-list INTERNAL seq 5 permit 156.1.0.0/16 le 32
ip prefix-list INTERNAL seq 10 permit 150.1.0.0/16 le 32
!
route-map LEAK permit 10
 match ip address prefix-list INTERNAL
!
route-map LEAK permit 20
 match ip address prefix-list BACKBONES
```

R6:

```
interface FastEthernet0/1
 ip summary-address eigrp 100 0.0.0.0 0.0.0.0 5 leak-map LEAK
!
ip prefix-list EIGRP_LEARNED_FROM_BB1 seq 5 permit 200.0.0.0/21 le 24
ip prefix-list EIGRP_LEARNED_FROM_BB1 seq 10 permit 54.1.1.0/24
!
ip prefix-list INTERNAL seq 5 permit 156.1.0.0/16 le 32
ip prefix-list INTERNAL seq 10 permit 150.1.0.0/16 le 32
!
route-map LEAK permit 10
 match ip address prefix-list INTERNAL
!
route-map LEAK permit 20
 match ip address prefix-list EIGRP_LEARNED_FROM_BB1
```


Task 2.8 Verification

Verify the default route advertisement. Confirm that native EIGRP routes are not suppressed but leaked:

```
Rack1R3#show ip route 0.0.0.0
Routing entry for 0.0.0.0/0, supernet
  Known via "eigrp 100", distance 90, metric 2588160, candidate default
path, type internal
  Redistributing via eigrp 100
  Last update from 156.1.35.5 on Virtual-Access3, 00:01:33 ago
  Routing Descriptor Blocks:
  * 156.1.35.5, from 156.1.35.5, 00:01:33 ago, via Virtual-Access3
      Route metric is 2588160, traffic share count is 1
      Total delay is 100100 microseconds, minimum bandwidth is 100000
Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1

Rack1R3#show ip eigrp 100 topology 0.0.0.0 0.0.0.0
IP-EIGRP (AS 100): Topology entry for 0.0.0.0/0
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is
2588160
  Routing Descriptor Blocks:
  156.1.35.5 (Virtual-Access3), from 156.1.35.5, Send flag is 0x0
      Composite metric is (2588160/28160), Route is Internal
      Vector metric:
        Minimum bandwidth is 100000 Kbit
        Total delay is 100100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
      Exterior flag is set
  156.1.23.2 (Serial1/3), from 156.1.23.2, Send flag is 0x0
      Composite metric is (20519680/33280), Route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 20300 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 3
      Exterior flag is set
```



```
Rack1R3#show ip route eigrp
D EX 200.0.0.0/24 [170/21157120] via 156.1.23.2, 00:03:27, Serial1/3
D EX 200.0.1.0/24 [170/21157120] via 156.1.23.2, 00:03:27, Serial1/3
     156.1.0.0/16 is variably subnetted, 13 subnets, 2 masks
D       156.1.27.0/24 [90/20514560] via 156.1.23.2, 1d14h, Serial1/3
D       156.1.18.0/24 [90/2588160] via 156.1.13.1, 00:04:10, Virtual-
Access2
D       156.1.8.0/24 [90/2588416] via 156.1.35.5, 00:04:10, Virtual-
Access3
                     [90/2588416] via 156.1.13.1, 00:04:10, Virtual-
Access2
D EX    156.1.4.0/24 [170/4522496] via 156.1.35.5, 00:04:19, Virtual-
Access3
D       156.1.58.0/24 [90/2588160] via 156.1.35.5, 00:04:10, Virtual-
Access3
D       156.1.45.0/24 [90/4729856] via 156.1.35.5, 00:04:20, Virtual-
Access3
D EX    156.1.44.0/24 [170/4522496] via 156.1.35.5, 00:04:19, Virtual-
Access3
D       156.1.45.4/32 [90/4729856] via 156.1.35.5, 00:04:20, Virtual-
Access3
D       156.1.67.0/24 [90/20517120] via 156.1.23.2, 1d14h, Serial1/3
D EX 200.0.2.0/24 [170/21157120] via 156.1.23.2, 00:03:27, Serial1/3
D EX 200.0.3.0/24 [170/21157120] via 156.1.23.2, 00:03:28, Serial1/3
     150.1.0.0/24 is subnetted, 8 subnets
D       150.1.7.0 [90/20642560] via 156.1.23.2, 1d14h, Serial1/3
D       150.1.6.0 [90/20645120] via 156.1.23.2, 00:03:28, Serial1/3
D       150.1.5.0 [90/2713600] via 156.1.35.5, 00:04:21, Virtual-
Access3
D EX    150.1.4.0 [170/4522496] via 156.1.35.5, 00:04:21, Virtual-
Access3
D       150.1.2.0 [90/20640000] via 156.1.23.2, 1d14h, Serial1/3
D       150.1.1.0 [90/2713600] via 156.1.13.1, 1d12h, Virtual-Access2
D       150.1.8.0 [90/2716160] via 156.1.35.5, 00:04:11, Virtual-
Access3
                  [90/2716160] via 156.1.13.1, 00:04:11, Virtual-
Access2
D*   0.0.0.0/0 [90/2588160] via 156.1.35.5, 00:03:39, Virtual-Access3
```

Verify the default route advertisement. Check that native EIGRP routes are not suppressed but leaked:

```
Rack1R3#show ip route 0.0.0.0
Routing entry for 0.0.0.0/0, supernet
  Known via "eigrp 100", distance 90, metric 2588160, candidate default
path, type internal
  Redistributing via eigrp 100
  Last update from 156.1.35.5 on Virtual-Access3, 00:01:33 ago
  Routing Descriptor Blocks:
  * 156.1.35.5, from 156.1.35.5, 00:01:33 ago, via Virtual-Access3
      Route metric is 2588160, traffic share count is 1
      Total delay is 100100 microseconds, minimum bandwidth is 100000
Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
```



```
Rack1R3#show ip eigrp 100 topology 0.0.0.0 0.0.0.0
IP-EIGRP (AS 100): Topology entry for 0.0.0.0/0
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is
2588160
  Routing Descriptor Blocks:
  156.1.35.5 (Virtual-Access3), from 156.1.35.5, Send flag is 0x0
      Composite metric is (2588160/28160), Route is Internal
      Vector metric:
        Minimum bandwidth is 100000 Kbit
        Total delay is 100100 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
      Exterior flag is set
  156.1.23.2 (Serial1/3), from 156.1.23.2, Send flag is 0x0
      Composite metric is (20519680/33280), Route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 20300 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 3
      Exterior flag is set

Rack1R3#show ip route eigrp
D EX 200.0.0.0/24 [170/21157120] via 156.1.23.2, 00:03:27, Serial1/3
D EX 200.0.1.0/24 [170/21157120] via 156.1.23.2, 00:03:27, Serial1/3
     156.1.0.0/16 is variably subnetted, 13 subnets, 2 masks
D       156.1.27.0/24 [90/20514560] via 156.1.23.2, 1d14h, Serial1/3
D       156.1.18.0/24 [90/2588160] via 156.1.13.1, 00:04:10, Virtual-
Access2
D       156.1.8.0/24 [90/2588416] via 156.1.35.5, 00:04:10, Virtual-
Access3
                     [90/2588416] via 156.1.13.1, 00:04:10, Virtual-
Access2
D EX    156.1.4.0/24 [170/4522496] via 156.1.35.5, 00:04:19, Virtual-
Access3
D       156.1.58.0/24 [90/2588160] via 156.1.35.5, 00:04:10, Virtual-
Access3
D       156.1.45.0/24 [90/4729856] via 156.1.35.5, 00:04:20, Virtual-
Access3
D EX    156.1.44.0/24 [170/4522496] via 156.1.35.5, 00:04:19, Virtual-
Access3
D       156.1.45.4/32 [90/4729856] via 156.1.35.5, 00:04:20, Virtual-
Access3
D       156.1.67.0/24 [90/20517120] via 156.1.23.2, 1d14h, Serial1/3
D EX 200.0.2.0/24 [170/21157120] via 156.1.23.2, 00:03:27, Serial1/3
D EX 200.0.3.0/24 [170/21157120] via 156.1.23.2, 00:03:28, Serial1/3
     150.1.0.0/24 is subnetted, 8 subnets
D       150.1.7.0 [90/20642560] via 156.1.23.2, 1d14h, Serial1/3
D       150.1.6.0 [90/20645120] via 156.1.23.2, 00:03:28, Serial1/3
D       150.1.5.0 [90/2713600] via 156.1.35.5, 00:04:21, Virtual-
Access3
```



```
D EX    150.1.4.0 [170/4522496] via 156.1.35.5, 00:04:21, Virtual-
Access3
D       150.1.2.0 [90/20640000] via 156.1.23.2, 1d14h, Serial1/3
D       150.1.1.0 [90/2713600] via 156.1.13.1, 1d12h, Virtual-Access2
D       150.1.8.0 [90/2716160] via 156.1.35.5, 00:04:11, Virtual-
Access3
                  [90/2716160] via 156.1.13.1, 00:04:11, Virtual-
Access2
D*   0.0.0.0/0 [90/2588160] via 156.1.35.5, 00:03:39, Virtual-Access3
```

Pitfall

R6 has some BGP prefixes installed with the nex-hop IP of R5 which equal to 150.1.5.5. The problem is that devices on the path between R6 and R5 will use the default route to recursively resolve the next-hop, and route packets back to R6. This will create a routing loop, which is not uncommon in situations where you use BGP with default routing.

```
Rack1SW1#traceroute 220.20.3.1

Type escape sequence to abort.
Tracing the route to 220.20.3.1

  1 156.1.67.6 0 msec 8 msec 0 msec
  2 156.1.67.7 0 msec 0 msec 9 msec
  3 156.1.67.6 0 msec 0 msec 8 msec
  4 156.1.67.7 0 msec 0 msec 0 msec
  5 156.1.67.6 8 msec 0 msec 0 msec
  6 156.1.67.7 9 msec 0 msec 0 msec
  7 156.1.67.6 0 msec 8 msec 0 msec
  8 156.1.67.7 0 msec 0 msec 8 msec
  9 156.1.67.6 0 msec 9 msec 0 msec
 10 156.1.67.7 0 msec 0 msec 8 msec
 11 156.1.67.6 0 msec 0 msec 9 msec
 12 156.1.67.7 0 msec 0 msec 0 msec
 13 156.1.67.6 8 msec 0 msec 0 msec
 14 156.1.67.7 8 msec 0 msec 0 msec
 15 156.1.67.6 9 msec 0 msec 0 msec
```

One solution would be to leak AS54 BGP routes on R5 into EIGRP.

R5:

```
!
router eigrp 100
 redistribute bgp 100 route-map BGP_LEAK metric 1500 1000 255 1 1500
!
route-map BGP_LEAK permit 10
 match ip address prefix-list BGP_LEAK
!
ip prefix-list BGP_LEAK seq 5 permit 222.22.2.0/24
ip prefix-list BGP_LEAK seq 10 permit 220.20.3.0/24
ip prefix-list BGP_LEAK seq 15 permit 205.90.31.0/24
!
ip prefix-list BACKBONES seq 15 permit 222.22.2.0/24
ip prefix-list BACKBONES seq 20 permit 220.20.3.0/24
ip prefix-list BACKBONES seq 25 permit 205.90.31.0/24
```

Now you can test reachability to external BGP prefixes:

```
Rack1SW1#traceroute 222.22.2.1

Type escape sequence to abort.
Tracing the route to 222.22.2.1

  1 156.1.27.2 0 msec 8 msec 0 msec
  2 156.1.23.3 17 msec 17 msec 16 msec
  3 156.1.35.5 42 msec 51 msec 41 msec
  4 192.10.1.254 51 msec *  42 msec

Rack1SW1#traceroute 220.20.3.1

Type escape sequence to abort.
Tracing the route to 220.20.3.1

  1 156.1.27.2 0 msec 0 msec 0 msec
  2 156.1.23.3 17 msec 17 msec 17 msec
  3 156.1.35.5 50 msec 42 msec 50 msec
  4 192.10.1.254 42 msec *  42 msec

Rack1SW1#traceroute 205.90.31.1

Type escape sequence to abort.
Tracing the route to 205.90.31.1

  1 156.1.27.2 0 msec 9 msec 0 msec
  2 156.1.23.3 17 msec 16 msec 17 msec
  3 156.1.35.5 42 msec 42 msec 42 msec
  4 192.10.1.254 42 msec *  42 msec
```


Task 3.1 Solution

R3:
```
interface FastEthernet0/0
 ipv6 nat
!
interface FastEthernet0/1
ipv6 nat
!
ipv6 nat v4v6 source 156.1.8.100 2001:CC1E:FFFF::100
ipv6 nat v6v4 source 2001:CC1E:1:3::100 156.1.8.50
ipv6 nat prefix 2001:CC1E:FFFF::/96
```

Task 3.1 Verification

Simulate IPv6 host on VLAN3 with unusued R6's interface:

R6:
```
interface FastEthernet0/0
 no shutdown
 ipv6 address 2001:CC1E:1:3::100/64
```

R3:
```
!
interface FastEthernet0/1
 no shutdown
 ip address 156.1.8.3 255.255.255.0
!
router eigrp 10
 passive-interface FastEthernet0/1
```

SW2:
```
interface FastEthernet0/6
 switchport access vlan 3
!
interface Vlan8
 ip address 156.1.8.100 255.255.255.100
```

SW3:
```
!
interface FastEthernet0/3
 switchport access vlan 8
```


Test basic configuration:

```
Rack1R3#ping 156.1.8.100

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 156.1.8.100, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms

Rack1R3#ping 2001:CC1E:1:3::100

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:CC1E:1:3::100, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/1/4 ms
```

Check local IP aliases:

```
Rack1R3#show ip aliases
Address Type             IP Address      Port
Interface                156.1.23.3
Interface                156.1.13.3
Interface                156.1.13.3
Interface                156.1.8.3  Interface
150.1.3.3  Interface                156.1.3.3
Interface                156.1.35.3  Inter-
face                156.1.35.3

Rack1SW2#ping 156.1.8.50

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 156.1.8.50, timeout is 2 seconds:
  ...
Success rate is 0 percent (0/5)
```

Note that 156.1.8.50 is NOT listed in IP aliases. That is, IPv6 NAT-PT does not create IPv4 alias automatically. There are three ways to resolve this issue:

1. Assign 156.1.8.50 as the secondary IP to FastEthernet 0/1 of R3
2. Create static ARP entry at SW2, pointing at R3
3. Create static route at SW2 for 156.1.8.50/32, pointing at R3

Following the first approach, assign 156.1.8.50 as secondary IP:


```
R3:
interface FastEthernet0/1
 ip address 156.1.8.50 255.255.255.0 secondary
```

Test the new configuration:

```
Rack1R3#debug ipv6 nat
IPv6 NAT-PT debugging is on

Rack1R6#debug ipv6 icmp
ICMP packet debugging is on

Rack1SW2#ping 156.1.8.50

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 156.1.8.50, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 4/4/4 ms

Rack1R3#
IPv6 NAT: icmp src (156.1.8.100) -> (2001:CC1E:FFFF::100), dst
(156.1.8.50) -> (2001:CC1E:1:3::100)
IPv6 NAT: icmp src (2001:CC1E:1:3::100) -> (156.1.8.50), dst
(2001:CC1E:FFFF::100) -> (156.1.8.100)
IPv6 NAT: icmp src (156.1.8.100) -> (2001:CC1E:FFFF::100), dst
(156.1.8.50) -> (2001:CC1E:1:3::100)
IPv6 NAT: icmp src (2001:CC1E:1:3::100) -> (156.1.8.50), dst
(2001:CC1E:FFFF::100) -> (156.1.8.100)
IPv6 NAT: icmp src (156.1.8.100) -> (2001:CC1E:FFFF::100), dst
(156.1.8.50) -> (2001:CC1E:1:3::100)
IPv6 NAT: icmp src (2001:CC1E:1:3::100) -> (156.1.8.50), dst
(2001:CC1E:FFFF::100) -> (156.1.8.100)
IPv6 NAT: icmp src (156.1.8.100) -> (2001:CC1E:FFFF::100), dst
(156.1.8.50) -> (2001:CC1E:1:3::100)
IPv6 NAT: icmp src (2001:CC1E:1:3::100) -> (156.1.8.50), dst
(2001:CC1E:FFFF::100) -> (156.1.8.100)
IPv6 NAT: icmp src (156.1.8.100) -> (2001:CC1E:FFFF::100), dst
(156.1.8.50) -> (2001:CC1E:1:3::100)
IPv6 NAT: icmp src (2001:CC1E:1:3::100) -> (156.1.8.50), dst
(2001:CC1E:FFFF::100) -> (156.1.8.100)
```

```
Rack1R6#
ICMPv6: Received echo request, Src=2001:CC1E:FFFF::100,
Dst=2001:CC1E:1:3::100
ICMPv6: Sent echo reply, Src=2001:CC1E:1:3::100,
Dst=2001:CC1E:FFFF::100
ICMPv6: Received echo request, Src=2001:CC1E:FFFF::100,
Dst=2001:CC1E:1:3::100
ICMPv6: Sent echo reply, Src=2001:CC1E:1:3::100,
Dst=2001:CC1E:FFFF::100
ICMPv6: Received echo request, Src=2001:CC1E:FFFF::100,
Dst=2001:CC1E:1:3::100
ICMPv6: Sent echo reply, Src=2001:CC1E:1:3::100,
Dst=2001:CC1E:FFFF::100
ICMPv6: Received echo request, Src=2001:CC1E:FFFF::100,
Dst=2001:CC1E:1:3::100
ICMPv6: Sent echo reply, Src=2001:CC1E:1:3::100,
Dst=2001:CC1E:FFFF::100
ICMPv6: Received echo request, Src=2001:CC1E:FFFF::100,
Dst=2001:CC1E:1:3::100
ICMPv6: Sent echo reply, Src=2001:CC1E:1:3::100,
Dst=2001:CC1E:FFFF::100
```

Task 5.1 Solution

R1:

```
ip pim autorp listener
!
ip pim send-rp-announce FastEthernet0/0 scope 16 group-list 1
!
access-list 1 permit 224.0.0.0 7.255.255.255
```

R5:

```
ip pim autorp listener
!
ip pim send-rp-announce FastEthernet0/1 scope 16 group-list 1
!
access-list 1 permit 232.0.0.0 7.255.255.255
```

SW2:

```
interface loopback0
 ip pim sparse-mode
!
ip pim send-rp-discovery loopback0 scope 16
```


Task 5.1 Verification

Verify RP mappings:

```
Rack1R1#show ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP (Auto-RP)

Group(s) 224.0.0.0/5
  RP 156.1.18.1 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:01:40, expires: 00:02:15
Group(s) 232.0.0.0/5
  RP 156.1.58.5 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:01:40, expires: 00:02:19

Rack1R5#show ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP (Auto-RP)

Group(s) 224.0.0.0/5
  RP 156.1.18.1 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:03:12, expires: 00:02:42
Group(s) 232.0.0.0/5
  RP 156.1.58.5 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:03:12, expires: 00:02:45

Rack1R3#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/5
  RP 156.1.18.1 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:01:53, expires: 00:02:02
Group(s) 232.0.0.0/5
  RP 156.1.58.5 (?), v2v1
    Info source: 150.1.8.8 (?), elected via Auto-RP
         Uptime: 00:01:53, expires: 00:02:02
```

Task 5.2 Solution

R3:

```
interface FastEthernet0/0
 ip igmp join-group 224.24.24.24
 ip igmp join-group 232.32.32.32
```


Task 5.2 Verification

Ping multicast groups from SW2:

```
Rack1SW2#debug ip icmp
ICMP packet debugging is on

Rack1SW2#ping 224.24.24.24 source vlan8

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 224.24.24.24, timeout is 2 seconds:
Packet sent with a source address of 156.1.8.8

Reply to request 0 from 156.1.13.3, 67 ms
Reply to request 0 from 156.1.13.3, 143 ms
Reply to request 0 from 156.1.13.3, 109 ms
Reply to request 0 from 156.1.13.3, 101 ms
ICMP: echo reply rcvd, src 156.1.13.3, dst 156.1.18.8
ICMP: echo reply rcvd, src 156.1.13.3, dst 156.1.8.8
ICMP: echo reply rcvd, src 156.1.13.3, dst 150.1.8.8
ICMP: echo reply rcvd, src 156.1.13.3, dst 156.1.58.8

Rack1SW2#ping 232.32.32.32 source vlan 8
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 232.32.32.32, timeout is 2 seconds:
Packet sent with a source address of 156.1.8.8

Reply to request 0 from 156.1.35.3, 67 ms
Reply to request 0 from 156.1.35.3, 134 ms
Reply to request 0 from 156.1.35.3, 109 ms
Reply to request 0 from 156.1.35.3, 92 ms
ICMP: echo reply rcvd, src 156.1.35.3, dst 156.1.58.8
ICMP: echo reply rcvd, src 156.1.35.3, dst 156.1.8.8
ICMP: echo reply rcvd, src 156.1.35.3, dst 156.1.18.8
ICMP: echo reply rcvd, src 156.1.35.3, dst 150.1.8.8
```

Task 5.3 Solution

R5:

```
access-list 10 deny 224.0.1.39
access-list 10 deny 224.0.1.40
access-list 10 permit any
!
interface FastEthernet0/0.2
ip multicast boundary 10
```


Task 5.3 Verification

Temporarily enable PIM on the FastEthernet interface connected to BB3. Check mroutes on R5 before applying the solution:

```
Rack1R5#show ip mroute 224.0.1.39
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.0.1.39), 00:02:19/stopped, RP 0.0.0.0, flags: DC
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0.2, Forward/Sparse, 00:00:07/00:00:00
    FastEthernet0/1, Forward/Sparse, 00:02:19/00:00:00
    Virtual-Access1, Forward/Sparse, 00:02:19/00:00:00

(156.1.18.1, 224.0.1.39), 00:02:19/00:02:44, flags: T
  Incoming interface: FastEthernet0/1, RPF nbr 156.1.58.8
  Outgoing interface list:
    FastEthernet0/0.2, Forward/Sparse, 00:00:08/00:00:00
    Virtual-Access1, Prune/Sparse, 00:00:19/00:02:43, A

(156.1.58.5, 224.0.1.39), 00:01:56/00:02:03, flags: T
  Incoming interface: FastEthernet0/1, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/0.2, Forward/Sparse, 00:00:08/00:00:00
    Virtual-Access1, Forward/Sparse, 00:01:56/00:00:00
```



```
Rack1R5#show ip mroute 224.0.1.40
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.0.1.40), 00:03:56/stopped, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/1, Forward/Sparse, 00:03:56/00:00:00
    FastEthernet0/0.2, Forward/Sparse, 00:03:56/00:00:00
    Virtual-Access1, Forward/Sparse, 00:03:56/00:00:00

(150.1.8.8, 224.0.1.40), 00:03:14/00:02:50, flags: LT
  Incoming interface: FastEthernet0/1, RPF nbr 156.1.58.8
  Outgoing interface list:
    FastEthernet0/0.2, Forward/Sparse, 00:03:15/00:00:00
    Virtual-Access1, Prune/Sparse, 00:02:13/00:00:49, A
```

Apply the solution and check mroutes again:

```
Rack1R5#show ip mroute 224.0.1.39
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.0.1.39), 00:05:33/stopped, RP 0.0.0.0, flags: DC
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/1, Forward/Sparse, 00:05:33/00:00:00
    Virtual-Access1, Forward/Sparse, 00:05:33/00:00:00
```



```
(156.1.18.1, 224.0.1.39), 00:05:33/00:00:26, flags: T
  Incoming interface: FastEthernet0/1, RPF nbr 156.1.58.8
  Outgoing interface list:
    Virtual-Access1, Forward/Sparse, 00:00:30/00:00:00, A

(156.1.58.5, 224.0.1.39), 00:05:10/00:02:49, flags: T
  Incoming interface: FastEthernet0/1, RPF nbr 0.0.0.0
  Outgoing interface list:
    Virtual-Access1, Forward/Sparse, 00:00:05/00:00:00, A

Rack1R5#show ip mroute 224.0.1.40
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.0.1.40), 00:06:04/stopped, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    FastEthernet0/1, Forward/Sparse, 00:06:04/00:00:00
    Virtual-Access1, Forward/Sparse, 00:06:04/00:00:00

(150.1.8.8, 224.0.1.40), 00:05:22/00:02:45, flags: PLT
  Incoming interface: FastEthernet0/1, RPF nbr 156.1.58.8
  Outgoing interface list:
    Virtual-Access1, Prune/Sparse, 00:00:23/00:02:39, A
```


Task 6.1 Solution

SW1 & SW2:

```
username SSH password 0 CISCO
!
ip domain-name ine.com
!
crypto key generate rsa usage-keys modulus 2048
!
line vty 0 15
 login local
 transport input ssh
```

Taks 6.1 Verification

Verify SSH status:

```
Rack1SW1#show ip ssh
SSH Enabled - version 1.99
Authentication timeout: 120 secs; Authentication retries: 3

Rack1R6#ssh -l SSH 150.1.7.7

Password: <CISCO>

Rack1SW1>exit

Rack1R6#telnet 150.1.7.7
Trying 150.1.7.7 ...
% Connection refused by remote host
```


Task 6.2 Solution

R4:

```
interface FastEthernet0/0
 ip access-group VLAN4 in
!
interface FastEthernet0/1
 ip access-group VLAN44 in
!
ip access-list extended VLAN4
 permit ip 156.1.4.0 0.0.0.255 156.1.44.0 0.0.0.255
 permit tcp 156.1.4.0 0.0.0.255 any eq www
 permit tcp 156.1.4.0 0.0.0.255 any eq 443
 permit tcp 156.1.4.0 0.0.0.255 any eq 8080
 permit tcp host 156.1.4.40 eq ftp-data any gt 1023
 permit tcp host 156.1.4.40 eq ftp any gt 1023
 permit tcp 156.1.4.0 0.0.0.255 any eq 1720
 permit udp 156.1.4.0 0.0.0.255 range 16384 32767 any range 16384 32767
 deny   ip any any
ip access-list extended VLAN44
 permit ip 156.1.44.0 0.0.0.255 156.1.4.0 0.0.0.255
 permit tcp 156.1.44.0 0.0.0.255 any eq www
 permit tcp 156.1.44.0 0.0.0.255 any eq 443
 permit tcp 156.1.44.0 0.0.0.255 any eq 8080
 permit tcp 156.1.44.0 0.0.0.255 any eq 1720
 permit udp 156.1.44.0 0.0.0.255 range 16384 32767 any range 16384
32767
 deny   ip any any
```

Task 6.2 Verification

```
Rack1R4#show ip interface fastEthernet 0/0 | section Inbound
  Inbound  access list is VLAN4

Rack1R4#show ip interface fastEthernet 0/1 | section Inbound
  Inbound  access list is VLAN44
```

Task 6.3 Solution

R4:

```
interface Serial0/1/0
 encapsulation ppp
 ppp chap refuse
 ppp pap sent-username ROUTER4 password 0 CISCO
 no peer neighbor-route
```

R5:

```
username ROUTER4 password 0 CISCO
!
interface Serial0/1/0
 encapsulation ppp
 clockrate 64000
 ppp authentication chap pap
 no peer neighbor-route
```


Task 6.3 Verification

Verify PPP authentication process:

```
Rack1R4#debug ppp negotiation
PPP protocol negotiation debugging is on

Rack1R4#debug ppp authentication
PPP authentication debugging is on

Rack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R4(config)#interface s0/1/0
Rack1R4(config-if)#shutdown
%LINK-5-CHANGED: Interface Serial0/1/0, changed state to
administratively down
 Se0/1/0 PPP: Sending Acct Event[Down] id[5]
 Se0/1/0 CDPCP: State is Closed
 Se0/1/0 IPCP: Remove link info for cef entry 156.1.45.5
 Se0/1/0 IPCP: State is Closed
 Se0/1/0 PPP: Phase is TERMINATING
 Se0/1/0 LCP: State is Closed
 Se0/1/0 PPP: Phase is DOWN
 Se0/1/0 IPCP: Remove route to 156.1.45.5
 Se0/1/0 IPCP: Remove default route thru 156.1.45.5
 %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1, changed
state to down

Rack1R4(config-if)#no shutdown
Se0/1/0 PPP: Phase is ESTABLISHING, Active Open
Se0/1/0 PPP: Authorization required
Se0/1/0 LCP: O CONFREQ [Closed] id 14 len 10
Se0/1/0 LCP:    MagicNumber 0x30D0BD58 (0x050630D0BD58)
 Se0/1/0 LCP: I CONFREQ [REQsent] id 7 len 15
 Se0/1/0 LCP:    AuthProto CHAP (0x0305C22305)
 Se0/1/0 LCP:    MagicNumber 0x08281AF9 (0x050608281AF9)
 Se0/1/0 LCP: O CONFNAK [REQsent] id 7 len 9
 Se0/1/0 LCP:    AuthProto MS-CHAP (0x0305C22380)
 Se0/1/0 LCP: I CONFACK [REQsent] id 14 len 10
 Se0/1/0 LCP:    MagicNumber 0x30D0BD58 (0x050630D0BD58)
 Se0/1/0 LCP: I CONFREQ [ACKrcvd] id 8 len 14
 Se0/1/0 LCP:    AuthProto PAP (0x0304C023)
 Se0/1/0 LCP:    MagicNumber 0x08281AF9 (0x050608281AF9)
 Se0/1/0 LCP: O CONFACK [ACKrcvd] id 8 len 14
 Se0/1/0 LCP:    AuthProto PAP (0x0304C023)
 Se0/1/0 LCP:    MagicNumber 0x08281AF9 (0x050608281AF9)
 Se0/1/0 LCP: State is Open
 Se0/1/0 PPP: No authorization without authentication
Se0/1/0 PPP: Phase is AUTHENTICATING, by the peer
 Se0/1/0 PAP: Using hostname from interface PAP
 Se0/1/0 PAP: Using password from interface PAP
 Se0/1/0 PAP: O AUTH-REQ id 3 len 18 from "ROUTER4"
 Se0/1/0 PAP: I AUTH-ACK id 3 len 5
Se0/1/0 PPP: Phase is FORWARDING, Attempting Forward
 Se0/1/0 PPP: Phase is ESTABLISHING, Finish LCP
```



```
 Se0/1/0 PPP: Phase is UP
 Se0/1/0 IPCP: O CONFREQ [Closed] id 1 len 10
 Se0/1/0 IPCP:    Address 156.1.45.4 (0x03069C012D04)
 Se0/1/0 CDPCP: O CONFREQ [Closed] id 1 len 4
 Se0/1/0 PPP: Process pending ncp packets
 Se0/1/0 CDPCP: I CONFREQ [REQsent] id 1 len 4
 Se0/1/0 CDPCP: O CONFACK [REQsent] id 1 len 4
 Se0/1/0 IPCP: I CONFREQ [REQsent] id 1 len 10
 Se0/1/0 IPCP:    Address 156.1.45.5 (0x03069C012D05)
 Se0/1/0 IPCP: O CONFACK [REQsent] id 1 len 10
 Se0/1/0 IPCP:    Address 156.1.45.5 (0x03069C012D05)
 Se0/1/0 CDPCP: I CONFACK [ACKsent] id 1 len 4
 Se0/1/0 CDPCP: State is Open
 Se0/1/0 IPCP: I CONFACK [ACKsent] id 1 len 10
 Se0/1/0 IPCP:    Address 156.1.45.4 (0x03069C012D04)
 Se0/1/0 IPCP: State is Open
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1, changed
      state to up
```


Task 7.1 Solution

R2:

```
logging 156.1.8.100
logging facility local2
logging trap critical
!
service sequence-numbers
```

Task 7.2 Solution

R2:

```
logging count
```

Tasks 7.1 – 7.2 Verification

Verify sequence numbers:

```
Rack1R2(config)#interface fastEthernet 0/0
Rack1R2(config-if)#shutdown
000035: *Mar  1 04:14:01.854: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 10:
Neighbor 156.1.27.7 (FastEthernet0/0) is down: interface down
Rack1R2(config-if)#
000036: *Mar  1 04:14:01.858: destroy peer: 156.1.27.7
Rack1R2(config-if)#
000037: *Mar  1 04:14:03.834: %LINK-5-CHANGED: Interface
FastEthernet0/0, changed state to administratively down
000038: *Mar  1 04:14:04.834: %LINEPROTO-5-UPDOWN: Line protocol on
Interface FastEthernet0/0, changed state to down
```

Verify logging count:

```
Rack1R2#show logging count
Facility       Message Name                 Sev Occur      Last Time
=============================================================================
SYS            CONFIG_I                      5    2 May 29 07:04:19.570
-------------  ----------------------------  ----------------------
SYS TOTAL                                       2

LINEPROTO      UPDOWN                        5    2 May 29 07:04:23.577
-------------  ----------------------------  ----------------------
LINEPROTO TOTAL                                 2

DUAL           NBRCHANGE                     5    2 May 29 07:04:53.446
-------------  ----------------------------  ----------------------
DUAL TOTAL                                      2

LINK           UPDOWN                        3    1 May 29 07:04:21.045
LINK           CHANGED                       5    1 May 29 07:04:17.968
-------------  ----------------------------  ----------------------
LINK TOTAL                                             2
```


Task 7.3 Solution

R1, R2, R3, R4, R5, R6, SW1, SW2, SW3, SW4:

```
ip domain-lookup
!
ip name-server 150.1.6.6
ip domain-name ine.com
```

R6:

```
ip dns server
!
ip host Rack1R1.ine.com 150.1.1.1
ip host Rack1R2.ine.com 150.1.2.2
ip host Rack1R3.ine.com 150.1.3.3
ip host Rack1R4.ine.com 150.1.4.4
ip host Rack1R5.ine.com 150.1.5.5
ip host Rack1R6.ine.com 150.1.6.6
ip host Rack1SW1.ine.com 150.1.7.7
ip host Rack1SW2.ine.com 150.1.8.8
ip host Rack1SW2.ine.com 150.1.9.9
ip host Rack1SW2.ine.com 150.1.10.10
```


Task 7.3 Verification

```
Rack1R3#debug domain
Domain Name System debugging is on
Rack1R3#
Rack1R3#ping Rack1R1

Translating "Rack1R1"...domain server (150.1.6.6) [OK]

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 = 1/2/4 ms
Rack1R3#
Domain: query for Rack1R1.ine.com type 1 to 150.1.6.6
DOM: dom2cache: hostname is Rack1R1.ine.com, RR type=1, class=1,
ttl=10, n=4Reply received ok
Rack1R3#

Rack1R6#debug domain
Domain Name System debugging is on
Rack1R6#
DNS: Incoming UDP query (id#31)
DNS: Type 1 DNS query (id#31) for host 'Rack1R1.ine.com' from
156.1.23.3(53481)
DNS: Finished processing query (id#31) in 0.004 secs

Rack1R3#show host
Default domain is ine.com
Name/address lookup uses domain service
Name servers are 150.1.6.6

Codes: UN - unknown, EX - expired, OK - OK, ?? - revalidate
       temp - temporary, perm - permanent
       NA - Not Applicable None - Not defined

Host                      Port  Flags      Age Type   Address(es)
Rack1R1.ine.com           None  (temp, OK)  0   IP    150.1.1.1
Rack1R3#
```


Task 7.4 Solution

R1:

```
interface Loopback0
 ip nat outside
!
interface Loopback1
 description arbitrary address
 ip address 1.1.1.1 255.255.255.255
 ip nat inside
 ip policy route-map POLICY1
!
route-map POLICY permit 10
 match ip address 100
 set interface Loopback1
!
route-map POLICY1 permit 10
 set interface Loopback0
!
access-list 100 permit icmp any any time-exceeded
access-list 100 permit icmp any any port-unreachable
!
ip nat inside source list 100 interface Loopback0 overload
!
ip local policy route-map POLICY
```

Task 7.4 Verification

Confirm that R1 will always reply to traceroute with Loopback0 source address:

```
Rack1R5#traceroute 150.1.1.1

Type escape sequence to abort.
Tracing the route to Rack1R1.ine.com (150.1.1.1)

  1 156.1.35.3 28 msec 28 msec 28 msec
  2 Rack1R1.ine.com (150.1.1.1) 36 msec *  24 msec

Rack1SW1#traceroute 150.1.1.1

Type escape sequence to abort.
Tracing the route to Rack1R1.ine.com (150.1.1.1)

  1 156.1.27.2 8 msec 0 msec 0 msec
  2 156.1.23.3 17 msec 8 msec 9 msec
  3 Rack1R1.ine.com (150.1.1.1) 50 msec *  42 msec
```



```
Rack1SW2#traceroute 150.1.1.1

Type escape sequence to abort.
Tracing the route to Rack1R1.ine.com (150.1.1.1)

  1 156.1.58.5 0 msec 0 msec 0 msec
  2 156.1.35.3 33 msec 33 msec 34 msec
  3 Rack1R1.ine.com (150.1.1.1) 25 msec *  25 msec
```

Task 7.5 Solution

```
R6:
snmp-server chassis-id Rack1-R6

rmon alarm 1 ifOutOctets.3 4 delta rising 1000 1 fall 1000
rmon event 1 log description Whoah!

event manager applet EIGRP-Load
 event syslog pattern "RMON-5-RISINGTRAP: Rising trap is generated
because the value of ifOutOctets.3 exceeded the rising-threshold value
1000"
 action 1.0 cli command "enable"
 action 1.1 cli command "configure terminal"
 action 1.2 cli command "router eigrp 100"
 action 1.3 cli command "metric weights 0 1 1 1 0 0"
exit
```

Task 7.5 Verification

```
Rack1R6#show rmon alarms
Alarm 1 is active, owned by config
 Monitors ifOutOctets.3 every 4 second(s)
 Taking delta samples, last value was 0
 Rising threshold is 1000, assigned to event 1
 Falling threshold is 1000, assigned to event 0
 On startup enable rising or falling alarm

Rack1R6#show event manager policy registered
No. Class   Type  Event Type  Trap  Time Registered          Name
1   applet  user  syslog      Off   Sat May 29 07:42:40 2010 EIGRP-Load
 pattern {RMON-5-RISINGTRAP: Rising trap is generated because the value
of ifOutOctets.3 exceeded the rising-threshold value 1000}
 maxrun 20.000
 action 1.0 cli command "enable"
 action 1.1 cli command "configure terminal"
 action 1.2 cli command "router eigrp 100"
 action 1.3 cli command "metric weights 0 1 1 1 0 0"
```


Task 7.6 Solution

R1, R2, R3, R5, SW1, SW2

```
!
event manager applet EIGRP-Load
 event syslog pattern "K-value mismatch"
 action 1.0 cli command "enable"
 action 1.1 cli command "configure terminal"
 action 1.2 cli command "router eigrp 100"
 action 1.3 cli command "metric weights 0 1 1 1 0 0"
exit
```

Task 7.6 Verification

```
Rack1R1#show event manager policy registered
No. Class  Type   Event Type  Trap Time Registered           Name
1   applet system syslog      Off  Sat May 29 07:42:40 2010  EIGRP-Load
 pattern {K-value mismatch}
 action 1.0 cli command "enable"
 action 1.1 cli command "configure terminal"
 action 1.2 cli command "router eigrp 100"
 action 1.3 cli command "metric weights 0 1 1 1 0 0"
```


Tasks 7.5 - 7.6 Breakdown

The best thing to do is to run "ping 54.X.1.254 size 1500 time 0 repeat 1000" from R6. That will be enough to trigger the RMON. You should then see every router lose EIGRP after R6 makes the change, and the cascading effect of the drop/modify/restore EIGRP functionality.

Task 8.1 Solution

Verify policy-map configuration:

```
Rack1R5#show policy-map interface Fa0/0.1
 FastEthernet0/0.1

  Service-policy output: 2.5Mbps

    Class-map: class-default (match-any)
      4 packets, 262 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 4/312
      shape (average) cir 2500000, bc 10000, be 10000
      target shape rate 2500000

Rack1R5#show policy-map interface Fa0/0.2
 FastEthernet0/0.2

  Service-policy output: 3Mbps

    Class-map: class-default (match-any)
      13 packets, 840 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 13/984
      shape (average) cir 3000000, bc 12000, be 12000
      target shape rate 3000000
```


Task 8.2 Solution

```
R4:
class-map VoIP
 match access-group name VoIP
!
policy-map VOIP_PRIORITY
 class VoIP
 priority 64
!
interface Serial0/1
 service-policy output VOIP_PRIORITY
!
ip access-list extended VoIP
 permit tcp any any eq 1720
 permit udp any any range 16384 32767
```

```
R5:
class-map VoIP
 match access-group name VoIP
!
policy-map VOIP_PRIORITY
 class VoIP
  priority 64
!
policy-map QOS_BB2
 class VoIP
  priority 64
!
policy-map 2.5Mbps
 class class-default
  service-policy QOS_BB2
!
interface Serial0/1
 service-policy output VOIP_PRIORITY
!
ip access-list extended VoIP
 permit tcp any any eq 1720
 permit udp any any range 16384 32767
```


Task 8.2 Verification

Verify QoS configuration:

```
Rack1R5#show policy-map interface s0/1/0
 Serial0/1/0

  Service-policy output: VOIP_PRIORITY

    queue stats for all priority classes:

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

    Class-map: VoIP (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name VoIP
      Priority: 64 kbps, burst bytes 1600, b/w exceed drops: 0


    Class-map: class-default (match-any)
      7 packets, 652 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 9/664
```



```
Rack1R5#show policy-map interface fastEthernet 0/0.1
 FastEthernet0/0.1

  Service-policy output: 2.5Mbps

    Class-map: class-default (match-any)
      106 packets, 7481 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 17/1345
      shape (average) cir 2500000, bc 10000, be 10000
      target shape rate 2500000

      Service-policy : QOS_BB2

        queue stats for all priority classes:

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

        Class-map: VoIP (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: access-group name VoIP
          Priority: 64 kbps, burst bytes 1600, b/w exceed drops: 0


        Class-map: class-default (match-any)
          17 packets, 1057 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: any

          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 17/1345
```

Task 8.3 Solution

R5:

```
class-map ICMP
 match protocol icmp
!
 policy-map QOS_BB2
  class ICMP
   police cir 16000
!
policy-map QOS_BB3
 class ICMP
   police cir 16000
!
policy-map 3Mbps
  class class-default
    service-policy QOS_BB3
```

Task 8.3 Verification

Simulate ping flood from SW2:

```
Rack1SW2#ping
Protocol [ip]:
Target IP address: 204.12.1.254
Repeat count [5]: 10000
Datagram size [100]: 1400
Timeout in seconds [2]: 0
Extended commands [n]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 10000, 1400-byte ICMP Echos to 204.12.1.254, timeout is 0
seconds:
 ..............................................................
<output omitted>
```


Check policy-map at R5:

```
Rack1R5#show policy-map interface fastEthernet 0/0.2
 FastEthernet0/0.2

  Service-policy output: 3Mbps

    Class-map: class-default (match-any)
      5342 packets, 7562724 bytes
      5 minute offered rate 210000 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 21/17715
      shape (average) cir 3000000, bc 12000, be 12000
      target shape rate 3000000

      Service-policy : QOS_BB3

        Class-map: ICMP (match-all)
          5333 packets, 7562194 bytes
          5 minute offered rate 210000 bps, drop rate 209000 bps
          Match: protocol icmp
          police:
              cir 16000 bps, bc 1500 bytes
            conformed 12 packets, 17016 bytes; actions:
              transmit
            exceeded 5321 packets, 7545178 bytes; actions:
              drop
            conformed 5000 bps, exceed 1728000 bps

        Class-map: class-default (match-any)
          9 packets, 530 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
```


Task 8.4 Solution

R5:

```
policy-map 2.5Mbps
  class class-default
   set dscp ef
!
policy-map 3Mbps
  class class-default
   set dscp ef
```

Task 8.4 Verification

Verify marking:

```
Rack1R5#show policy-map interface fastEthernet 0/0.1
 FastEthernet0/0.1

  Service-policy output: 2.5Mbps

    Class-map: class-default (match-any)
      4289 packets, 463346 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 4134/347663
      shape (average) cir 2500000, bc 10000, be 10000
      target shape rate 2500000
      QoS Set
        dscp ef
          Packets marked 8

      Service-policy : QOS_BB2

        queue stats for all priority classes:

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

        Class-map: VoIP (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: access-group name VoIP
          Priority: 64 kbps, burst bytes 1600, b/w exceed drops: 0


        Class-map: ICMP (match-all)
          89 packets, 120782 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: protocol icmp
          police:
              cir 16000 bps, bc 1500 bytes
            conformed 23 packets, 21794 bytes; actions:
              transmit
            exceeded 66 packets, 98988 bytes; actions:
              drop
            conformed 0 bps, exceed 0 bps
```

```
Class-map: class-default (match-any)
  4111 packets, 336140 bytes
  5 minute offered rate 0 bps, drop rate 0 bps
  Match: any

  queue limit 64 packets
  (queue depth/total drops/no-buffer drops) 0/0/0
  (pkts output/bytes output) 4134/347663
```

Task 8.5 Solution

SW1:

```
mls qos
!
class-map match-any EF_AND_CS5
match ip dscp ef cs5
!
policy-map RATE_LIMIT
class EF_AND_CS5
police 1000000 16000 exceed-action drop
!
interface FastEthernet0/10
service-policy input RATE_LIMIT
!
interface FastEthernet0/11
service-policy input RATE_LIMIT
```

Task 8.5 Verification

Temporarily apply policy map to Fa 0/5 at SW1 and configure dscp monitoring:

SW1:

```
!
interface FastEthernet0/5
 service-policy input RATE_LIMIT
```

Verify statistics:

```
Rack1SW1#show mls qos interface fastEthernet 0/5 statistics
FastEthernet0/5 (All statistics are in packets)

  dscp: incoming
-------------------------------

  0 -  4 :       13740          0            0            0            0
  5 -  9 :           0          0            0            0            0
 10 - 14 :           0          0            0            0            0
 15 - 19 :           0          0            0            0            0
 20 - 24 :           0          0            0            0            0
 25 - 29 :           0          0            0            0            0
 30 - 34 :           0          0            0            0            0
 35 - 39 :           0          0            0            0            0
 40 - 44 :           0          0            0            0            0
 45 - 49 :           0        228            0       115694            0
 50 - 54 :           0          0            0            0            0
 55 - 59 :           0          0            0            0            0
 60 - 64 :           0          0            0            0
  dscp: outgoing
-------------------------------
```



```
  0 -  4 :       14316          0              0              0              0
  5 -  9 :           0          0              0              0              0
 10 - 14 :           0          0              0              0              0
 15 - 19 :           0          0              0              0              0
 20 - 24 :           0          0              0              0              0
 25 - 29 :           0          0              0              0              0
 30 - 34 :           0          0              0              0              0
 35 - 39 :           0          0              0              0              0
 40 - 44 :           0          0              0              0              0
 45 - 49 :           0          0              0         723102              0
 50 - 54 :           0          0              0              0              0
 55 - 59 :           0      62077              0              0              0
 60 - 64 :           0          0              0              0
  cos: incoming
------------------------------

  0 -  4 :      165531          0              0              0              0
  5 -  7 :           0          0              0
  cos: outgoing
------------------------------

  0 -  4 :      785851          0              0              0              0
  5 -  7 :       28308     138819             14
Policer: Inprofile:       101 OutofProfile:              0
```

Task 8.6 Solution

SW1:

```
!
! Enable QoS and change markdown settings
!
mls qos
mls qos map policed-dscp 32 to 8
!
! Class-map to match the specific port
!
class-map PORT_TO_R5
 match input-interface FastEthernet 0/5
!
! Access-lists and class-maps to match the traffic
!
ip access-list extended ICMP
 permit icmp any any
!
ip access-list extended TCP
 permit tcp any any
!
class-map ICMP
 match access-group name ICMP
!
class-map TCP
 match access-group name TCP

!
! Interface-level policers - policing only
!
policy-map POLICE_256
 class PORT_TO_R5
  police 256000 16000
!
policy-map POLICE_512
 class PORT_TO_R5
  police 512000 32000 exceed policed-dscp-transmit
```



```
!
! VLAN level policers - marking only
!
policy-map VLAN_52_POLICY
 class ICMP
  set ip precedence 3
  service-policy POLICE_256
!
policy-map VLAN_53_POLICY
 class TCP
  set ip precedence 4
  service-policy POLICE_512
!
interface Vlan 52
 service-policy input VLAN_52_POLICY
!
interface Vlan 53
 service-policy input VLAN_53_POLICY

!
! Enable VLAN-based QoS on the port
!
interface FastEthernet 0/5
 mls qos vlan-based
```


Task 8.6 Verification

```
Rack1SW1#show mls qos interface fastEthernet 0/5
FastEthernet0/5
trust state: not trusted
trust mode: not trusted
trust enabled flag: ena
COS override: dis
default COS: 0
DSCP Mutation Map: Default DSCP Mutation Map
Trust device: none
qos mode: vlan-based

Rack1SW1#show mls qos vlan 52
Vlan52
Attached policy-map for Ingress: VLAN_52_POLICY

Rack1SW1#show mls qos vlan 53
Vlan53
Attached policy-map for Ingress: VLAN_53_POLICY

Rack1SW1#show mls qos maps policed-dscp
   Policed-dscp map:
     d1 :  d2 0  1  2  3  4  5  6  7  8  9
     ---------------------------------------
      0 :    00 01 02 03 04 05 06 07 08 09
      1 :    10 11 12 13 14 15 16 17 18 19
      2 :    20 21 22 23 24 25 26 27 28 29
      3 :    30 31 08 33 34 35 36 37 38 39
      4 :    40 41 42 43 44 45 46 47 48 49
      5 :    50 51 52 53 54 55 56 57 58 59
      6 :    60 61 62 63
```


Lab 19 Solutions

Task 1.1 Solution

SW1:

```
!
! The following VLAN definitions were missing from the configuration
!
Vlan 7,77
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
 channel-group 1 mode on
!
interface FastEthernet0/14
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
 channel-group 1 mode on
!
interface FastEthernet0/15
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
 channel-group 1 mode on
```


SW2:

```
!
! The following VLAN definitions were missing from the configuration
!
Vlan 8,88
!
port-channel load-balance dst-ip
!
interface FastEthernet0/13
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
 channel-group 1 mode on
!
interface FastEthernet0/14
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
 channel-group 1 mode on
!
interface FastEthernet0/15
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
 channel-group 1 mode on
!
interface FastEthernet0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
 channel-group 2 mode on

interface FastEthernet0/17
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
 channel-group 2 mode on
!
interface FastEthernet0/18
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
 channel-group 2 mode on
```


SW3:

```
interface FastEthernet0/16
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
 channel-group 1 mode on
!
interface FastEthernet0/17
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
 channel-group 1 mode on
!
interface FastEthernet0/18
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport nonegotiate
 channel-group 1 mode on
```

SW1:

```
vlan 10
 remote-span
!
monitor session 1 source vlan 127 rx
monitor session 1 destination remote vlan 10
```

SW3:

```
monitor session 1 destination interface Fa0/10
monitor session 1 source remote vlan 10
```


Task 1.1 Verification

```
RSRack1SW2#show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 2
Number of aggregators:           2

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------
1      Po1(SU)          -        Fa0/13(P)   Fa0/14(P)   Fa0/15(P)
2      Po2(SU)          -        Fa0/16(P)   Fa0/17(P)   Fa0/18(P)

RSRack1SW2#show etherchannel protocol
                Channel-group listing:
                ----------------------

Group: 1
----------
Protocol:   -   (Mode ON)

Group: 2
----------
Protocol:   -   (Mode ON)

RSRack1SW2#show interfaces trunk

Port        Mode             Encapsulation  Status        Native vlan
Po1         on               802.1q         trunking      1
Po2         on               802.1q         trunking      1

Port        Vlans allowed on trunk
Po1         1-4094
Po2         1-4094

Port        Vlans allowed and active in management domain
Po1         1,7-8,10,32-33,44,77,88,127,568
Po2         1,7-8,10,32-33,44,77,88,127,568

Port        Vlans in spanning tree forwarding state and not pruned
Po1         1,7-8,10,32-33,44,77,88,127,568
Po2         1,7-8,10,32-33,44,77,88,127,568
```



```
Rack1SW2#show etherchannel load-balance
EtherChannel Load-Balancing Operational State (dst-ip):
Non-IP: Destination MAC address
  IPv4: Destination IP address
  IPv6: Destination IP address

Rack1SW1#show vlan id 10

VLAN Nam                        Status    Ports
---- ------------------------- --------- -------------------------------
10   RSPAN                      active    Fa0/19, Po1

VLAN Type SAID MTU Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ---- ---- ---- ----- ------ -------- ---- -------- ------ ------
10   enet 100010 1500  -      -      -        -    -        0      0

Remote SPAN VLAN
----------------
Enabled

Rack1SW1#show monitor session 1
Session 1
---------
Type              : Remote Source Session
Source VLANs      :
    RX Only       : 127
Dest RSPAN VLAN   : 10

Rack1SW3#show monitor session 1
Session 1
---------
Type              : Remote Destination Session
Source RSPAN VLAN : 10
Destination Ports : Fa0/10
    Encapsulation : Native
          Ingress : Disabled
```


Task 1.2 Solution

R3:

```
frame-relay switching
!
interface Serial 1/2
 clock rate 64000
 encapsulation frame-relay
 frame-relay intf-type dce
 no shutdown
!
interface Serial 1/3
 clock rate 64000
 encapsulation frame-relay
 frame-relay intf-type dce
 no shutdown
!
connect R1_R2 Serial 1/2 132 Serial 1/3 231
```

R1:

```
interface Serial 0/1
 encapsulation frame-relay
 no frame-relay inverse-arp
 ip address 149.1.12.1 255.255.255.0
 frame-relay map ip 149.1.12.2 132
 no shutdown
```

R2:

```
interface Serial 0/1
 encapsulation frame-relay
 no frame-relay inverse-arp
 ip address 149.1.12.2 255.255.255.0
 frame-relay map ip 149.1.12.1 231
 no shutdown
```


Task 1.2 Verification

```
RSRack1R1#show frame-relay pvc interface serial 0/1

PVC Statistics for interface Serial0/1 (Frame Relay DTE)

             Active     Inactive      Deleted       Static
  Local         1            0            0            0
  Switched      0            0            0            0
  Unused        0            0            0            0

DLCI = 132, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/1

  input pkts 5             output pkts 5            in bytes 520
  out bytes 520            dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 0         out bcast bytes 0
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 00:23:59, last time pvc status changed 00:22:10

RSRack1R3#show frame-relay pvc 132

PVC Statistics for interface Serial1/2 (Frame Relay DCE)

DLCI = 132, DLCI USAGE = SWITCHED, PVC STATUS = ACTIVE, INTERFACE =
Serial1/2

  input pkts 5             output pkts 5            in bytes 520
  out bytes 520            dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 0         out bcast bytes 0
  30 second input rate 0 bits/sec, 0 packets/sec
  30 second output rate 0 bits/sec, 0 packets/sec
  switched pkts 5
  Detailed packet drop counters:
  no out intf 0            out intf down 0          no out PVC 0
  in PVC down 0            out PVC down 0           pkt too big 0
  shaping Q full 0         pkt above DE 0           policing drop 0
  pvc create time 00:24:49, last time pvc status changed 00:24:35
```



```
RSRack1R3#show frame-relay pvc 231

PVC Statistics for interface Serial1/3 (Frame Relay DCE)

DLCI = 231, DLCI USAGE = SWITCHED, PVC STATUS = ACTIVE, INTERFACE =
Serial1/3

  input pkts 5             output pkts 5            in bytes 520
  out bytes 520            dropped pkts 0           in pkts dropped 0
  out pkts dropped 0                 out bytes dropped 0
  in FECN pkts 0           in BECN pkts 0           out FECN pkts 0
  out BECN pkts 0          in DE pkts 0             out DE pkts 0
  out bcast pkts 0         out bcast bytes 0
  30 second input rate 0 bits/sec, 0 packets/sec
  30 second output rate 0 bits/sec, 0 packets/sec
  switched pkts 5
  Detailed packet drop counters:
  no out intf 0            out intf down 0          no out PVC 0
  in PVC down 0            out PVC down 0           pkt too big 0
  shaping Q full 0         pkt above DE 0           policing drop 0
  pvc create time 00:24:54, last time pvc status changed 00:24:35

RSRack1R1#ping 149.1.12.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 149.1.12.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/60 ms
```

Task 1.3 Solution

R4:

```
aaa new-model
aaa authentication ppp default group MYTACACS local
aaa authentication login CONSOLE none
!
username R5PPP password CISCO
!
aaa group server tacacs+ MYTACACS
 server-private 149.1.4.100 key CISCO
 ip tacacs source-interface Loopback0
!
interface Serial 0/1/0
 ppp authentication pap
!
line console 0
 login authentication CONSOLE
```

R5:

```
interface Serial 0/1/0
 ppp pap sent-username R5PPP password CISCO
```


Task 1.3 Verification

Observe the authentication procedure:

```
RSRack1R4#debug tacacs packet
TACACS+ packets debugging is on

RSRack1R4#debug tacacs
TACACS access control debugging is on

RSRack1R4#debug ppp negotiation
PPP protocol negotiation debugging is on

RSRack1R4#debug ppp authentication
PPP authentication debugging is on

RSRack1R4#debug aaa authentication
AAA Authentication debugging is on

RSRack1R4#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
RSRack1R4(config)#int serial 0/1/0
RSRack1R4(config-if)#shutdown

Se0/1/0 PPP: Sending Acct Event[Down] id[4]
Se0/1/0 CDPCP: State is Closed
Se0/1/0 IPCP: State is Closed
Se0/1/0 PPP: Phase is TERMINATING
Se0/1/0 LCP: State is Closed
Se0/1/0 PPP: Phase is DOWN
Se0/1/0 IPCP: Remove route to 149.1.45.5
```



```
RSRack1R4(config-if)#no shutdown

%LINK-3-UPDOWN: Interface Serial0/1/0, changed state to up
AAA/BIND(00000005): Bind i/f Serial0/1/0
Se0/1/0 PPP: Using default call direction
Se0/1/0 PPP: Treating connection as a dedicated line
Se0/1/0 PPP: Session handle[4D000005] Session id[4]
Se0/1/0 PPP: Phase is ESTABLISHING, Active Open
Se0/1/0 PPP: Authorization NOT required
Se0/1/0 LCP: O CONFREQ [Closed] id 4 len 14
Se0/1/0 LCP:    AuthProto PAP (0x0304C023)
Se0/1/0 LCP:    MagicNumber 0x15D8FB40 (0x050615D8FB40)
Se0/1/0 LCP: I CONFREQ [REQsent] id 4 len 10
Se0/1/0 LCP:    MagicNumber 0x139125B7 (0x0506139125B7)
Se0/1/0 LCP: O CONFACK [REQsent] id 4 len 10
Se0/1/0 LCP:    MagicNumber 0x139125B7 (0x0506139125B7)
Se0/1/0 LCP: I CONFACK [ACKsent] id 4 len 14
Se0/1/0 LCP:    AuthProto PAP (0x0304C023)
Se0/1/0 LCP:    MagicNumber 0x15D8FB40 (0x050615D8FB40)
Se0/1/0 LCP: State is Open
Se0/1/0 PPP: Phase is AUTHENTICATING, by this end
Se0/1/0 PAP: I AUTH-REQ id 3 len 16 from "R5PPP"
Se0/1/0 PAP: Authenticating peer R5PPP
Se0/1/0 PPP: Phase is FORWARDING, Attempting Forward
Se0/1/0 PPP: Phase is AUTHENTICATING, Unauthenticated User
AAA/AUTHEN/PPP (00000005): Pick method list 'default'
Se0/1/0 PPP: Sent PAP LOGIN Request

TPLUS: Queuing AAA Authentication request 5 for processing
TPLUS: processing authentication start request id 5
TPLUS: Authentication start packet created for 5(R5PPP)
TPLUS: Using server 149.1.4.100

TPLUS(00000005)/0/NB_WAIT/6608CF18: Started 5 sec timeout
Se0/1/0 PPP: Outbound ip packet dropped
Se0/1/0 PPP: Outbound cdp packet dropped
Se0/1/0 PPP: Outbound cdp packet dropped

TPLUS(00000005)/0/NB_WAIT/6608CF18: timed out
TPLUS(00000005)/0/NB_WAIT/6608CF18: timed out, clean up
TPLUS(00000005)/0/6608CF18: Processing the reply packet
```



```
Se0/1/0 PPP: Received LOGIN Response PASS
Se0/1/0 PPP: Phase is FORWARDING, Attempting Forward
Se0/1/0 PPP: Phase is AUTHENTICATING, Authenticated User
Se0/1/0 PAP: O AUTH-ACK id 3 len 5
Se0/1/0 PPP: Phase is UP
Se0/1/0 IPCP: O CONFREQ [Closed] id 1 len 10
Se0/1/0 IPCP:    Address 149.1.45.4 (0x030695012D04)
Se0/1/0 CDPCP: O CONFREQ [Closed] id 1 len 4
Se0/1/0 PPP: Process pending ncp packets
Se0/1/0 CDPCP: I CONFREQ [REQsent] id 1 len 4
Se0/1/0 CDPCP: O CONFACK [REQsent] id 1 len 4
Se0/1/0 CDPCP: I CONFACK [ACKsent] id 1 len 4
Se0/1/0 CDPCP: State is Open
Se0/1/0 IPCP: I CONFREQ [REQsent] id 1 len 10
Se0/1/0 IPCP:    Address 149.1.45.5 (0x030695012D05)
Se0/1/0 IPCP: O CONFACK [REQsent] id 1 len 10
Se0/1/0 IPCP:    Address 149.1.45.5 (0x030695012D05)
Se0/1/0 IPCP: I CONFACK [ACKsent] id 1 len 10
Se0/1/0 IPCP:    Address 149.1.45.4 (0x030695012D04)
Se0/1/0 IPCP: State is Open
Se0/1/0 IPCP: Install route to 149.1.45.5
```


Task 2.1 Solution

R3:
```
interface Serial1/1
 ip ospf network point-to-multipoint
!
router ospf 1
 router-id 150.1.3.3
 network 149.1.254.3 0.0.0.0 area 0
```

R4:
```
interface Serial0/0/0
 ip ospf network point-to-multipoint
!
router ospf 1
 router-id 150.1.4.4
 network 149.1.254.4 0.0.0.0 area 0
 network 149.1.44.4 0.0.0.0 area 0
```

R5:
```
interface Serial0/0/0
 ip ospf network point-to-multipoint
!
router ospf 1
 router-id 150.1.5.5
 network 149.1.254.5 0.0.0.0 area 0
```

R5:
```
interface FastEthernet0/1
 ip ospf network non-broadcast
 ip ospf message-digest-key 1 md5 0 CISCO
!
router ospf 1
 area 568 authentication message-digest
 neighbor 149.1.0.6
 neighbor 149.1.0.8
```

R6:
```
interface FastEthernet0/0
 ip ospf network non-broadcast
 ip ospf message-digest-key 1 md5 0 CISCO
!
router ospf 1
 area 568 authentication message-digest
 neighbor 149.1.0.5
 neighbor 149.1.0.8
```


SW2:

```
interface Vlan568
 ip ospf network non-broadcast
 ip ospf message-digest-key 1 md5 0 CISCO
!
router ospf 1
 area 568 authentication message-digest
```

R4 and R5:

```
interface Serial0/1
 ip ospf retransmit-interval 10
 ip ospf transmit-delay 5
```

Task 2.1 Verification

Verify OSPF neighbors:

```
RSRack1R5#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.4.4         0   FULL/  -        00:01:34    149.1.254.4     Serial0/0/0
150.1.3.3         0   FULL/  -        00:01:30    149.1.254.3     Serial0/0/0
150.1.4.4         0   FULL/  -        00:00:39    149.1.45.4      Serial0/1/0
150.1.6.6         1   FULL/DR         00:01:35    149.1.0.6     FastEthernet0/1
150.1.8.8         0   FULL/DROTHER    00:01:51    149.1.0.8     FastEthernet0/1
```

Verify OSPF routes and test connectivity:

```
RSRack1R3#show ip route ospf
     54.0.0.0/24 is subnetted, 1 subnets
O E2    54.1.2.0 [110/20] via 149.1.254.5, 00:08:41, Serial1/1
     149.1.0.0/16 is variably subnetted, 7 subnets, 2 masks
O       149.1.254.4/32 [110/845] via 149.1.254.5, 00:14:35, Serial1/1
O       149.1.254.5/32 [110/781] via 149.1.254.5, 00:14:35, Serial1/1
O IA    149.1.0.0/24 [110/782] via 149.1.254.5, 00:14:10, Serial1/1
O       149.1.44.0/24 [110/846] via 149.1.254.5, 00:14:35, Serial1/1
O       149.1.45.0/24 [110/10780] via 149.1.254.5, 00:14:35, Serial1/1
     150.1.0.0/24 is subnetted, 5 subnets
O E2    150.1.6.0 [110/20] via 149.1.254.5, 00:08:41, Serial1/1
O E2    150.1.5.0 [110/20] via 149.1.254.5, 00:08:41, Serial1/1
O E2    150.1.4.0 [110/20] via 149.1.254.5, 00:08:41, Serial1/1
O E2    150.1.8.0 [110/20] via 149.1.254.5, 00:08:41, Serial1/1
```

```
Rack1R3#ping 149.1.254.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 149.1.254.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 116/118/128
ms
```


Confirm OSPF network type and authentication:

```
RSRack1R5#show ip ospf interface FastEthernet 0/1
FastEthernet0/1 is up, line protocol is up
  Internet Address 149.1.0.5/24, Area 568
  Process ID 1, Router ID 150.1.5.5, Network Type NON_BROADCAST, Cost:
1
  Transmit Delay is 1 sec, State BDR, Priority 2
  Designated Router (ID) 150.1.6.6, Interface address 149.1.0.6
  Backup Designated router (ID) 150.1.5.5, Interface address 149.1.0.5
  Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit
5
    oob-resync timeout 120
    Hello due in 00:00:25
  Supports Link-local Signaling (LLS)
  Cisco NSF helper support enabled
  IETF NSF helper support enabled
  Index 1/1, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 5
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 2, Adjacent neighbor count is 2
    Adjacent with neighbor 150.1.6.6  (Designated Router)
    Adjacent with neighbor 150.1.8.8
  Suppress hello for 0 neighbor(s)
  Message digest authentication enabled
    Youngest key id is 1
```

Verify interface-level timers:

```
RSRack1R5#show ip ospf interface serial 0/1/0
Serial0/1/0 is up, line protocol is up
  Internet Address 149.1.45.5/24, Area 0
  Process ID 1, Router ID 150.1.5.5, Network Type POINT_TO_POINT, Cost:
9999
  Transmit Delay is 5 sec, State POINT_TO_POINT
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 10
    oob-resync timeout 40
    Hello due in 00:00:07
  Supports Link-local Signaling (LLS)
  Cisco NSF helper support enabled
  IETF NSF helper support enabled
  Index 1/2, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 3
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.4.4
  Suppress hello for 0 neighbor(s)
```


Task 2.2 Solution

R1:
```
router bgp 300
 redistribute connected route-map CONNECTED2BGP
!
route-map CONNECTED2BGP permit 10
 match interface Loopback0
```

R2:
```
router bgp 300
 redistribute connected route-map CONNECTED2BGP
!
route-map CONNECTED2BGP permit 10
 match interface Loopback0
```

R3:
```
router bgp 200
 redistribute connected route-map CONNECTED2BGP
!
route-map CONNECTED2BGP permit 10
 match interface Serial1/0
```

R4:
```
router bgp 100
 redistribute connected route-map CONNECTED2BGP
!
route-map CONNECTED2BGP permit 10
 match interface FastEthernet0/1
```

R5:
```
router bgp 100
 redistribute connected route-map CONNECTED2BGP
!
route-map CONNECTED2BGP permit 10
 match interface FastEthernet0/0
```

SW1:
```
router bgp 300
 redistribute connected route-map CONNECTED2BGP
!
route-map CONNECTED2BGP permit 10
 match interface Vlan7 Vlan77 Vlan127 Loopback0
```

SW2:
```
router bgp 100
 redistribute connected route-map CONNECTED2BGP
!
route-map CONNECTED2BGP permit 10
 match interface Vlan8 Vlan88
```


R3:
```
router bgp 200
 neighbor 149.1.123.1 default-originate
 neighbor 149.1.123.1 filter-list 1 out
 neighbor 149.1.123.2 default-originate
 neighbor 149.1.123.2 filter-list 1 out
!
ip as-path access-list 1 permit ^([0-9]+)?$
```

R1:
```
router bgp 300
 neighbor 149.1.123.3 route-map R3_OUT out
!
ip prefix-list VLAN_7 seq 5 permit 149.1.7.0/24
!
ip prefix-list VLAN_77 seq 5 permit 149.1.77.0/24
!
route-map R3_OUT permit 10
 match ip address prefix-list VLAN_7
 set metric 1
!
route-map R3_OUT permit 20
 match ip address prefix-list VLAN_77
 set metric 2
!
route-map R3_OUT permit 30
```

R2:
```
router bgp 300
 neighbor 149.1.123.3 route-map R3_OUT out
!
ip prefix-list VLAN_7 seq 5 permit 149.1.7.0/24
!
ip prefix-list VLAN_77 seq 5 permit 149.1.77.0/24
!
route-map R3_OUT permit 10
 match ip address prefix-list VLAN_7
 set metric 2
!
route-map R3_OUT permit 20
 match ip address prefix-list VLAN_77
 set metric 1
!
route-map R3_OUT permit 30
```

R3:
```
router bgp 200
 maximum-paths 2
```


Task 2.2 Verification

Verify BGP prefix origination:

```
RSRack1R3#show ip bgp q _(300||100)$
BGP table version is 71, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 149.1.4.0/24     149.1.254.4                           0 100 ?
*> 149.1.5.0/24     149.1.254.5              0            0 100 ?
*  149.1.7.0/24     149.1.123.2              2            0 300 ?
*>                  149.1.123.1              1            0 300 ?
*> 149.1.8.0/24     149.1.254.5                           0 100 ?
*> 149.1.77.0/24    149.1.123.2              1            0 300 ?
*                   149.1.123.1              2            0 300 ?
*> 149.1.88.0/24    149.1.254.5                           0 100 ?
*> 149.1.123.0/24   0.0.0.0                  0        32768 ?
*  149.1.127.0/24   149.1.123.2                          0 300 ?
*>                  149.1.123.1                           0 300 ?
*  150.1.1.0/24     149.1.123.2                           0 300 ?
*>                  149.1.123.1              0            0 300 ?
*  150.1.2.0/24     149.1.123.1                           0 300 ?
*>                  149.1.123.2              0            0 300 ?
*  150.1.7.0/24     149.1.123.2                           0 300 ?
*>                  149.1.123.1                           0 300 ?
```


Verify BGP table at SW1 before applying solution:

```
Rack1SW1#show ip bgp
BGP table version is 25, local router ID is 150.1.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop        Metric LocPrf Weight Path
* i28.119.16.0/24   149.1.127.2          0    100      0 200 54 i
*>i                 149.1.127.1          0    100      0 200 54 i
* i28.119.17.0/24   149.1.127.2          0    100      0 200 54 i
*>i                 149.1.127.1          0    100      0 200 54 i
* i112.0.0.0        149.1.127.2          0    100      0 200 54 50 60 i
*>i                 149.1.127.1          0    100      0 200 54 50 60 i
* i113.0.0.0        149.1.127.2          0    100      0 200 54 50 60 i
*>i                 149.1.127.1          0    100      0 200 54 50 60 i
* i114.0.0.0        149.1.127.2          0    100      0 200 54 i
*>i                 149.1.127.1          0    100      0 200 54 i
* i115.0.0.0        149.1.127.2          0    100      0 200 54 i
*>i                 149.1.127.1          0    100      0 200 54 i
* i116.0.0.0        149.1.127.2          0    100      0 200 54 i
*>i                 149.1.127.1          0    100      0 200 54 i
* i117.0.0.0        149.1.127.2          0    100      0 200 54 i
*>i                 149.1.127.1          0    100      0 200 54 i
* i118.0.0.0        149.1.127.2          0    100      0 200 54 i
*>i                 149.1.127.1          0    100      0 200 54 i
* i119.0.0.0        149.1.127.2          0    100      0 200 54 i
*>i                 149.1.127.1          0    100      0 200 54 i
* i149.1.4.0/24     149.1.127.2          0    100      0 200 100 ?
*>i                 149.1.127.1          0    100      0 200 100 ?
* i149.1.5.0/24     149.1.127.2          0    100      0 200 100 ?
*>i                 149.1.127.1          0    100      0 200 100 ?
*> 149.1.7.0/24     0.0.0.0              0         32768 ?
* i149.1.8.0/24     149.1.127.2          0    100      0 200 100 ?
*>i                 149.1.127.1          0    100      0 200 100 ?
*> 149.1.77.0/24    0.0.0.0              0         32768 ?
* i149.1.88.0/24    149.1.127.2          0    100      0 200 100 ?
*>i                 149.1.127.1          0    100      0 200 100 ?
* i149.1.123.0/24   149.1.127.2          0    100      0 200 ?
*>i                 149.1.127.1          0    100      0 200 ?
*> 149.1.127.0/24   0.0.0.0              0         32768 ?
*>i150.1.1.0/24     149.1.127.1          0    100      0 ?
*>i150.1.2.0/24     149.1.127.2          0    100      0 ?
*> 150.1.7.0/24     0.0.0.0              0         32768 ?
* i205.90.31.0      149.1.127.2          0    100      0 200 254 ?
*>i                 149.1.127.1          0    100      0 200 254 ?
* i220.20.3.0       149.1.127.2          0    100      0 200 254 ?
*>i                 149.1.127.1          0    100      0 200 254 ?
* i222.22.2.0       149.1.127.2          0    100      0 200 254 ?
*>i                 149.1.127.1          0    100      0 200 254 ?
```


Compare it with BGP table after you applied solution:

```
Rack1SW1#show ip bgp
BGP table version is 30, local router ID is 150.1.7.7
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i0.0.0.0         149.1.127.2              0    100      0 200 i
*>i               149.1.127.1              0    100      0 200 i
* i28.119.16.0/24  149.1.127.2              0    100      0 200 54 i
*>i               149.1.127.1              0    100      0 200 54 i
* i28.119.17.0/24  149.1.127.2              0    100      0 200 54 i
*>i               149.1.127.1              0    100      0 200 54 i
* i114.0.0.0       149.1.127.2              0    100      0 200 54 i
*>i               149.1.127.1              0    100      0 200 54 i
* i115.0.0.0       149.1.127.2              0    100      0 200 54 i
*>i               149.1.127.1              0    100      0 200 54 i
* i116.0.0.0       149.1.127.2              0    100      0 200 54 i
*>i               149.1.127.1              0    100      0 200 54 i
* i117.0.0.0       149.1.127.2              0    100      0 200 54 i
*>i               149.1.127.1              0    100      0 200 54 i
* i118.0.0.0       149.1.127.2              0    100      0 200 54 i
*>i               149.1.127.1              0    100      0 200 54 i
* i119.0.0.0       149.1.127.2              0    100      0 200 54 i
*>i               149.1.127.1              0    100      0 200 54 i
* i149.1.4.0/24    149.1.127.2              0    100      0 200 100 ?
*>i               149.1.127.1              0    100      0 200 100 ?
* i149.1.5.0/24    149.1.127.2              0    100      0 200 100 ?
*>i               149.1.127.1              0    100      0 200 100 ?
*> 149.1.7.0/24    0.0.0.0                  0         32768 ?
* i149.1.8.0/24    149.1.127.2              0    100      0 200 100 ?
*>i               149.1.127.1              0    100      0 200 100 ?
*> 149.1.77.0/24   0.0.0.0                  0         32768 ?
* i149.1.88.0/24   149.1.127.2              0    100      0 200 100 ?
*>i               149.1.127.1              0    100      0 200 100 ?
* i149.1.123.0/24  149.1.127.2              0    100      0 200 ?
*>i               149.1.127.1              0    100      0 200 ?
*> 149.1.127.0/24  0.0.0.0                  0         32768 ?
*>i150.1.1.0/24   149.1.127.1              0    100      0 ?
*>i150.1.2.0/24   149.1.127.2              0    100      0 ?
*> 150.1.7.0/24    0.0.0.0                  0         32768 ?
* i205.90.31.0     149.1.127.2              0    100      0 200 254 ?
*>i               149.1.127.1              0    100      0 200 254 ?
* i220.20.3.0      149.1.127.2              0    100      0 200 254 ?
*>i               149.1.127.1              0    100      0 200 254 ?
* i222.22.2.0      149.1.127.2              0    100      0 200 254 ?
*>i               149.1.127.1              0    100      0 200 254 ?
```


Verify BGP best-paths for VLAN7, VLAN7 and VLAN127 prefixes:

```
RSRack1R3#show ip bgp 149.1.7.0
BGP routing table entry for 149.1.7.0/24, version 70
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Multipath: eBGP
  Advertised to update-groups:
     1           2           3
  300
    149.1.123.2 from 149.1.123.2 (150.1.2.2)
      Origin incomplete, metric 2, localpref 100, valid, external
  300
    149.1.123.1 from 149.1.123.1 (150.1.1.1)
      Origin incomplete, metric 1, localpref 100, valid, external, best

RSRack1R3#show ip bgp 149.1.77.0
BGP routing table entry for 149.1.77.0/24, version 71
Paths: (2 available, best #1, table Default-IP-Routing-Table)
Multipath: eBGP
  Advertised to update-groups:
     1           2           3
  300
    149.1.123.2 from 149.1.123.2 (150.1.2.2)
      Origin incomplete, metric 1, localpref 100, valid, external, best
  300
    149.1.123.1 from 149.1.123.1 (150.1.1.1)
      Origin incomplete, metric 2, localpref 100, valid, external


RSRack1R3#show ip bgp 149.1.127.0
BGP routing table entry for 149.1.127.0/24, version 64
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Multipath: eBGP
  Advertised to update-groups:
     1           2           3
  300
    149.1.123.2 from 149.1.123.2 (150.1.2.2)
      Origin incomplete, localpref 100, valid, external, multipath
  300
    149.1.123.1 from 149.1.123.1 (150.1.1.1)
      Origin incomplete, localpref 100, valid, external, multipath,
best
```



```
RSRack1R3#sh ip route 149.1.127.0
Routing entry for 149.1.127.0/24
  Known via "bgp 200", distance 20, metric 0
  Tag 300, type external
  Last update from 149.1.123.1 00:45:15 ago
  Routing Descriptor Blocks:
    149.1.123.2, from 149.1.123.2, 00:45:15 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
      Route tag 300
  * 149.1.123.1, from 149.1.123.1, 00:45:15 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
      Route tag 300
```

Task 2.3 Solution

R3:

```
!
router bgp 200
 aggregate-address 149.1.0.0 255.255.128.0 as-set summary-only
   advertise-map ONLY_AS_100
!
ip as-path access-list 2 permit ^100$
!
route-map ONLY_AS_100 permit 10
 match as-path 2
```

R3:

```
router bgp 200
 neighbor 149.1.254.5 unsuppress-map UNSUPPRESS
 neighbor 149.1.254.5 route-map R5_OUT out
!
ip prefix-list AGGREGATE seq 5 permit 149.1.0.0/17
!
route-map UNSUPPRESS permit 10
!
route-map R5_OUT deny 10
 match ip address prefix-list AGGREGATE
!
route-map R5_OUT permit 1000
```


Task 2.3 Verification

Verify summary generation:

```
RSRack1R3#show ip bgp | section 0.0.0.0
*> 149.1.0.0/17     0.0.0.0                        100  32768 100 ?
s> 149.1.123.0/24   0.0.0.0                  0         32768 ?
```

Confirm that R3 does not send summary prefix to AS100, only specifics:

```
RSRack1R3#show ip bgp neighbors 149.1.254.5 advertised-routes
BGP table version is 82, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 28.119.16.0/24   204.12.1.254             0             0 54 i
*> 28.119.17.0/24   204.12.1.254             0             0 54 i
*> 112.0.0.0        204.12.1.254                           0 54 50 60 i
*> 113.0.0.0        204.12.1.254                           0 54 50 60 i
*> 114.0.0.0        204.12.1.254                           0 54 i
*> 115.0.0.0        204.12.1.254                           0 54 i
*> 116.0.0.0        204.12.1.254                           0 54 i
*> 117.0.0.0        204.12.1.254                           0 54 i
*> 118.0.0.0        204.12.1.254                           0 54 i
*> 119.0.0.0        204.12.1.254                           0 54 i
s> 149.1.7.0/24     149.1.123.1              1             0 300 ?
s> 149.1.77.0/24    149.1.123.2              1             0 300 ?
s> 149.1.123.0/24   0.0.0.0                  0         32768 ?
s> 149.1.127.0/24   149.1.123.1                            0 300 ?
*> 150.1.1.0/24     149.1.123.1              0             0 300 ?
*> 150.1.2.0/24     149.1.123.2              0             0 300 ?
*> 150.1.7.0/24     149.1.123.1                            0 300 ?
*> 205.90.31.0      192.10.1.254             0             0 254 ?
*> 220.20.3.0       192.10.1.254             0             0 254 ?
*> 222.22.2.0       192.10.1.254             0             0 254 ?

Total number of prefixes 20
```


Confirm that R3 still sends summary to other neighbors:

```
RSRack1R3#show ip bgp neighbors 204.12.1.254 advertised-routes
BGP table version is 82, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 149.1.0.0/17     0.0.0.0                  100  32768 100 ?
*> 150.1.1.0/24     149.1.123.1              0             0 300 ?
*> 150.1.2.0/24     149.1.123.2              0             0 300 ?
*> 150.1.7.0/24     149.1.123.1                            0 300 ?
*> 205.90.31.0      192.10.1.254             0             0 254 ?
*> 220.20.3.0       192.10.1.254             0             0 254 ?
*> 222.22.2.0       192.10.1.254             0             0 254 ?

Total number of prefixes 7
```

Verify new summary prefix:

```
Rack1R3#show ip bgp 149.1.0.0
BGP routing table entry for 149.1.0.0/17, version 46
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Multipath: eBGP
Flag: 0x820
  Advertised to update-groups:
     1           2           3
  100, (aggregated by 200 150.1.3.3)
    0.0.0.0 from 0.0.0.0 (150.1.3.3)
      Origin incomplete, localpref 100, weight 32768, valid,
aggregated, local, atomic-aggregate, best
```

Confirm that AS300 sees that prefix in BGP tables:

```
RSRack1SW1#show ip bgp 149.1.0.0
BGP routing table entry for 149.1.0.0/17, version 81
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Not advertised to any peer
  200 100, (aggregated by 200 150.1.3.3)
    149.1.127.1 from 149.1.127.1 (150.1.1.1)
      Origin incomplete, metric 0, localpref 100, valid, internal,
atomic-aggregate, best
  200 100, (aggregated by 200 150.1.3.3)
    149.1.127.2 from 149.1.127.2 (150.1.2.2)
      Origin incomplete, metric 0, localpref 100, valid, internal,
atomic-aggregate
```


And AS100 does not see it:

```
Rack1SW2#show ip bgp 149.1.0.0
% Network not in table
```

Task 3.1 Solution

R1:

```
ipv6 unicast-routing
!
interface FastEthernet0/0
 ipv6 address 2001:149:1:127::/64 eui-64
!
interface Serial0/0
 ipv6 address 2001:149:1:123::1/64
 ipv6 address fe80::1 link-local
 frame-relay map ipv6 2001:149:1:123::2 103 broadcast
 frame-relay map ipv6 FE80::3 103
 frame-relay map ipv6 FE80::2 103
 frame-relay map ipv6 2001:149:1:123::3 103
```

R2:

```
ipv6 unicast-routing
!
interface FastEthernet0/0
 ipv6 address 2001:149:1:127::/64 eui-64
 ipv6 nd suppress-ra
!
interface Serial0/0
 ipv6 address 2001:149:1:123::2/64
 ipv6 address fe80::2 link-local
 frame-relay map ipv6 2001:149:1:123::3 203 broadcast
 frame-relay map ipv6 FE80::1 203
 frame-relay map ipv6 FE80::3 203
 frame-relay map ipv6 2001:149:1:123::1 203
```

R3:

```
ipv6 unicast-routing
!
interface Loopback100
 ipv6 address 2001:220:20:3::3/64
!
interface Serial1/0
 ipv6 address 2001:149:1:123::3/64
 ipv6 address fe80::3 link-local
 frame-relay map ipv6 FE80::2 302
 frame-relay map ipv6 FE80::1 301
 frame-relay map ipv6 2001:149:1:123::2 302 broadcast
 frame-relay map ipv6 2001:149:1:123::1 301 broadcast
```


Task 3.1 Verification

Verify IPv6 address assignment:

```
RSRack1R3#show ipv6 interface brief
FastEthernet0/0            [up/up]
FastEthernet0/1            [up/up]
Serial1/0                  [up/up]
    FE80::3
    2001:149:1:123::3
Serial1/1                  [up/up]
Serial1/2                  [up/up]
Serial1/3                  [up/up]
Loopback0                  [up/up]
Loopback100                [up/up]
    FE80::211:20FF:FE93:E560
    2001:220:20:3::3

RSRack1R1#show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::213:80FF:FEAD:7A80
    2001:149:1:127:213:80FF:FEAD:7A80
Serial0/0                  [up/up]
    FE80::1
    2001:149:1:123::1
Serial0/1                  [up/up]
Loopback0                  [up/up]

RSRack1R2#show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::211:20FF:FEFD:6E00
    2001:149:1:127:211:20FF:FEFD:6E00
Serial0/0                  [up/up]
    FE80::2
    2001:149:1:123::2
Serial0/1                  [up/up]
Loopback0                  [up/up]
```


Verify L3 to L2 mappings:

```
RSRack1R3#show frame-relay map
Serial1/0 (up): ipv6 FE80::2 dlci 302(0x12E,0x48E0), static,
              CISCO, status defined, active
Serial1/0 (up): ipv6 2001:149:1:123::1 dlci 301(0x12D,0x48D0), static,
              broadcast,
              CISCO, status defined, active
Serial1/0 (up): ipv6 2001:149:1:123::2 dlci 302(0x12E,0x48E0), static,
              broadcast,
              CISCO, status defined, active
Serial1/0 (up): ip 149.1.123.1 dlci 301(0x12D,0x48D0), static,
              broadcast,
              CISCO, status defined, active
Serial1/0 (up): ip 149.1.123.2 dlci 302(0x12E,0x48E0), static,
              broadcast,
              CISCO, status defined, active
Serial1/0 (up): ipv6 FE80::1 dlci 301(0x12D,0x48D0), static,
              CISCO, status defined, active
Serial1/1 (up): ip 149.1.254.5 dlci 315(0x13B,0x4CB0), static,
              broadcast,
              CISCO, status defined, active

RSRack1R2#show frame-relay map
Serial0/0 (up): ipv6 FE80::3 dlci 203(0xCB,0x30B0), static,
              CISCO, status defined, active
Serial0/0 (up): ipv6 2001:149:1:123::1 dlci 203(0xCB,0x30B0), static,
              CISCO, status defined, active
Serial0/0 (up): ip 149.1.123.1 dlci 203(0xCB,0x30B0), static,
              CISCO, status defined, active
Serial0/0 (up): ipv6 2001:149:1:123::3 dlci 203(0xCB,0x30B0), static,
              broadcast,
              CISCO, status defined, active
Serial0/0 (up): ip 149.1.123.3 dlci 203(0xCB,0x30B0), static,
              broadcast,
              CISCO, status defined, active
Serial0/0 (up): ipv6 FE80::1 dlci 203(0xCB,0x30B0), static,
              CISCO, status defined, active
Serial0/1 (up): ip 149.1.12.1 dlci 231(0xE7,0x3870), static,
              CISCO, status defined, active
```



```
RSRack1R1#show frame-relay map
Serial0/0 (up): ipv6 FE80::2 dlci 103(0x67,0x1870), static,
              CISCO, status defined, active
Serial0/0 (up): ipv6 FE80::3 dlci 103(0x67,0x1870), static,
              CISCO, status defined, active
Serial0/0 (up): ipv6 2001:149:1:123::2 dlci 103(0x67,0x1870), static,
              broadcast,
              CISCO, status defined, active
Serial0/0 (up): ipv6 2001:149:1:123::3 dlci 103(0x67,0x1870), static,
              CISCO, status defined, active
Serial0/0 (up): ip 149.1.123.2 dlci 103(0x67,0x1870), static,
              CISCO, status defined, active
Serial0/0 (up): ip 149.1.123.3 dlci 103(0x67,0x1870), static,
              broadcast,
              CISCO, status defined, active
Serial0/1 (up): ip 149.1.12.2 dlci 132(0x84,0x2040), static,
              CISCO, status defined, active
```

Test connectivity:

```
Rack1R1#ping 2001:149:1:123::3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:149:1:123::3, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/32 ms

Rack1R1#ping 2001:149:1:123::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:149:1:123::2, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 108/110/112
ms
```


Check for ICMPv6 ND RA:

```
RSRack1R1#show ipv6 interface fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::213:80FF:FEAD:7A80
  Global unicast address(es):
    2001:149:1:127:213:80FF:FEAD:7A80, subnet is 2001:149:1:127::/64
[EUI]
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FFAD:7A80
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 200 seconds
  ND router advertisements live for 1800 seconds
  Hosts use stateless autoconfig for addresses.

RSRack1R2#show ipv6 interface fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::211:20FF:FEFD:6E00
  Global unicast address(es):
    2001:149:1:127:211:20FF:FEFD:6E00, subnet is 2001:149:1:127::/64
[EUI]
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FFFD:6E00
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
  Hosts use stateless autoconfig for addresses.
```


Task 3.2 Solution

R1:

```
interface FastEthernet0/0
 ipv6 rip RIPng enable
!
interface Serial0/0
 ipv6 rip RIPng enable
```

R2:

```
interface FastEthernet0/0
 ipv6 rip RIPng enable
!
interface Serial0/0
 ipv6 rip RIPng enable
 ipv6 rip RIPng metric-offset 3
```

R3:

```
ipv6 router rip RIPng
  no split-horizon
!
interface Loopback100
ipv6 rip RIPng enable
!
interface Serial1/0
ipv6 rip RIPng enable
```

Task 3.2 Verification

```
RSRack1R1#show ipv6 route rip
IPv6 Routing Table - 7 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   2001:220:20:3::/64 [120/2]
     via FE80::3, Serial0/0

RSRack1R2#show ipv6 route rip
IPv6 Routing Table - 7 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   2001:220:20:3::/64 [120/3]
     via FE80::213:80FF:FEAD:7A80, FastEthernet0/0
```


Shutdown serial interface on R1 and wait for the routing tables to converge:

```
RSRack1R1#show ipv6 route rip
IPv6 Routing Table - 6 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS
summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF
ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   2001:149:1:123::/64 [120/2]
     via FE80::211:20FF:FEFD:6E00, FastEthernet0/0
R   2001:220:20:3::/64 [120/5]
     via FE80::211:20FF:FEFD:6E00, FastEthernet0/0
```

Task 5.1 Solution

```
R3:
interface Loopback0
 ip pim sparse-mode
!
ip pim bsr-candidate Loopback0 0

SW1:
interface Loopback0
 ip pim sparse-mode
!
ip pim rp-candidate Loopback0
```

Task 5.1 Verification

```
RSRack1SW1#show ip pim bsr-router
PIMv2 Bootstrap information
  BSR address: 150.1.3.3 (?)
  Uptime:      01:52:17, BSR Priority: 0, Hash mask length: 0
  Expires:     00:02:25
  Candidate RP: 150.1.7.7(Loopback0)
    Advertisement interval 60 seconds
    Next advertisement in 00:00:56

RSRack1R5#show ip pim bsr-router
PIMv2 Bootstrap information
  BSR address: 150.1.3.3 (?)
  Uptime:      01:53:55, BSR Priority: 0, Hash mask length: 0
  Expires:     00:01:27

Rack1R4#show ip pim bsr-router
PIMv2 Bootstrap information
```

Check RP mapping information:

```
RSRack1R5#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/4
  RP 150.1.7.7 (?), v2
    Info source: 150.1.3.3 (?), via bootstrap, priority 0, holdtime 210
         Uptime: 02:02:30, expires: 00:02:55

RSRack1R1#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/4
  RP 150.1.7.7 (?), v2
    Info source: 150.1.3.3 (?), via bootstrap, priority 0, holdtime 210
         Uptime: 02:03:06, expires: 00:03:21
```


Task 5.2 Solution

R4:

```
interface FastEthernet0/1
 ip igmp join-group 224.1.1.1
```

R5:

```
interface FastEthernet0/0
 ip igmp join-group 224.1.1.1
!
interface Serial0/0/0
 ip pim nbma-mode
```

Task 5.2 Verification

Confirm that R4 has RP mapping information via BSR:

```
Rack1R4#show ip pim bsr-router
PIMv2 Bootstrap information
  BSR address: 150.1.3.3 (?)
  Uptime:      00:00:27, BSR Priority: 0, Hash mask length: 0
  Expires:     00:02:02

RSRack1R4#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/4
  RP 150.1.7.7 (?), v2
    Info source: 150.1.3.3 (?), via bootstrap, priority 0, holdtime 210
         Uptime: 07:23:27, expires: 00:02:48
```

Confirm multicast reachability:

```
Rack1R1#ping 224.1.1.1

Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 224.1.1.1, timeout is 2 seconds:

Reply to request 0 from 149.1.254.5, 92 ms
Reply to request 0 from 149.1.254.4, 188 ms
Reply to request 0 from 149.1.254.4, 164 ms
Reply to request 0 from 149.1.254.5, 116 ms
```



```
Rack1R5#show ip mroute 224.1.1.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.1.1.1), 00:04:55/00:03:10, RP 150.1.7.7, flags: SJCL
  Incoming interface: Serial0/0/0, RPF nbr 149.1.254.3
  Outgoing interface list:
    Serial0/0/0, 149.1.254.4, Forward/Sparse, 00:04:14/00:03:10
    FastEthernet0/0, Forward/Sparse, 00:04:22/00:02:40

(149.1.123.1, 224.1.1.1), 00:01:26/00:02:23, flags: LT
  Incoming interface: Serial0/0/0, RPF nbr 149.1.254.3
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:01:27/00:02:39
    Serial0/0/0, 149.1.254.4, Forward/Sparse, 00:01:27/00:03:09

(149.1.127.1, 224.1.1.1), 00:01:27/00:02:22, flags: LT
  Incoming interface: Serial0/0/0, RPF nbr 149.1.254.3
  Outgoing interface list:
    FastEthernet0/0, Forward/Sparse, 00:01:27/00:02:39
    Serial0/0/0, 149.1.254.4, Forward/Sparse, 00:01:27/00:03:09
```



```
RSRack1R4#show ip mroute 224.1.1.1
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group,
       V - RD & Vector, v - Vector
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.1.1.1), 07:26:37/stopped, RP 150.1.7.7, flags: SJCL
  Incoming interface: Serial0/0/0, RPF nbr 149.1.254.5
  Outgoing interface list:
    FastEthernet0/1, Forward/Sparse, 07:26:37/00:02:29

(149.1.123.1, 224.1.1.1), 00:00:59/00:02:01, flags: LJT
  Incoming interface: Serial0/0/0, RPF nbr 149.1.254.5
  Outgoing interface list:
    FastEthernet0/1, Forward/Sparse, 00:00:59/00:02:29

(149.1.127.1, 224.1.1.1), 00:00:59/00:02:01, flags: LJT
  Incoming interface: Serial0/0/0, RPF nbr 149.1.254.5
  Outgoing interface list:
    FastEthernet0/1, Forward/Sparse, 00:00:59/00:02:29
```



```
RSRack1SW1#ping
Protocol [ip]:
Target IP address: 224.1.1.1
Repeat count [1]: 5
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Interface [All]: Vlan127
Time to live [255]:
Source address: 149.1.7.7
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 224.1.1.1, timeout is 2 seconds:
Packet sent with a source address of 149.1.7.7

Reply to request 0 from 149.1.254.5, 176 ms
Reply to request 0 from 149.1.254.4, 495 ms
Reply to request 0 from 149.1.254.4, 436 ms
Reply to request 0 from 149.1.254.5, 269 ms
Reply to request 1 from 149.1.254.5, 176 ms
Reply to request 1 from 149.1.254.4, 436 ms
Reply to request 2 from 149.1.254.5, 176 ms
Reply to request 2 from 149.1.254.4, 436 ms
Reply to request 3 from 149.1.254.5, 176 ms
Reply to request 3 from 149.1.254.4, 436 ms
Reply to request 4 from 149.1.254.5, 176 ms
Reply to request 4 from 149.1.254.4, 436 ms
```

Task 5.3 Solution

R4:

```
access-list 44 permit 224.1.1.1
!
interface FastEthernet0/1

 ip multicast rate-limit out group-list 44 512000
 ip multicast rate-limit out 2000000
```

R5:

```
access-list 55 permit 224.1.1.1
!
interface FastEthernet0/0

 ip multicast rate-limit out group-list 55 512000
 ip multicast rate-limit out 2000000
```


Task 5.3 Verification

```
RSRack1R5#show ip mroute
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C -
Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP
Advertisement,
       U - URD, I - Received Source Specific Host Report,
       Z - Multicast Tunnel, z - MDT-data group sender,
       Y - Joined MDT-data group, y - Sending to MDT-data group,
       V - RD & Vector, v - Vector
Outgoing interface flags: H - Hardware switched, A - Assert winner
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.1.1.1), 07:54:02/00:02:29, RP 150.1.7.7, flags: SJCL
  Incoming interface: Serial0/0/0, RPF nbr 149.1.254.3
  Outgoing interface list:
    Serial0/0/0, 149.1.254.4, Forward/Sparse, 07:53:36/00:02:34
    FastEthernet0/0, Forward/Sparse, 07:54:02/00:02:29, limit 512000
kbps

(*, 224.0.1.40), 2d10h/00:02:50, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial0/0/0, 224.0.1.40, Forward/Sparse, 2d10h/00:02:50
```


Task 6.1 Solution

R6:

```
class-map match-all ICMP
  match protocol icmp
!
policy-map POLICE_DoS
  class ICMP
   police cir 8000 bc 1000
    conform-action transmit
    exceed-action drop
!
interface Serial0/0/0
 service-policy input POLICE_DoS
```

Task 6.1 Verification

Verify policy-map configuration with extended pings from BB1:

```
BB1#ping 54.1.2.6 size 1000 repeat 10

Type escape sequence to abort.
Sending 10, 1000-byte ICMP Echos to 54.1.2.6, timeout is 2 seconds:
 ........
Success rate is 0 percent (0/10)

BB1#ping 54.1.2.6 size 500 repeat 10

Type escape sequence to abort.
Sending 10, 500-byte ICMP Echos to 54.1.2.6, timeout is 2 seconds:
!!.!!.!!.!
Success rate is 70 percent (7/10), round-trip min/avg/max = 132/132/136
ms
```



```
Rack1R6#show policy-map interface serial 0/0/0

 Serial0/0/0

  Service-policy input: POLICE_DoS

    Class-map: ICMP (match-all)
      20 packets, 15080 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol icmp
      police:
          cir 8000 bps, bc 1000 bytes
        conformed 7 packets, 3528 bytes; actions:
          transmit
        exceeded 13 packets, 11552 bytes; actions:
          drop
        conformed 0 bps, exceed 0 bps

    Class-map: class-default (match-any)
      269 packets, 21611 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```


Task 6.2 Solution

R6:

```
interface Serial0/0/0
 ip access-group PRIVATE_ADDRESS_SPACE in
!
ip access-list standard PRIVATE_ADDRESS_SPACE
 deny   10.0.0.0 0.255.255.255
 deny   172.16.0.0 0.15.255.255
 deny   192.168.0.0 0.0.255.255
 permit any
```

Task 6.2 Verification

```
RSRack1R6#show ip interface serial 0/0/0 | section Inbound
  Inbound  access list is PRIVATE_ADDRESS_SPACE

RSRack1R6#show access-lists PRIVATE_ADDRESS_SPACE
Standard IP access list PRIVATE_ADDRESS_SPACE
    10 deny   10.0.0.0, wildcard bits 0.255.255.255
    20 deny   172.16.0.0, wildcard bits 0.15.255.255
    30 deny   192.168.0.0, wildcard bits 0.0.255.255
    40 permit any (21 matches)
```

Task 7.1 Solution

R3:

```
ip ftp source-interface Loopback0
ip ftp username R3FTP
ip ftp password CISCO
```

Task 7.1 Verification

Configure R5 as a FTP server

```
Rack1R5#copy startup-config flash:test.txt
Destination filename [test.txt]?
Erase flash: before copying? [confirm]n
Verifying checksum...  OK (0xF6D)
2643 bytes copied in 0.292 secs (9051 bytes/sec)
```

Notice that recent version of IOS do not have the FTP server feature – the test was performed using an older IOS version.

```
Rack1R5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R5(config)#ftp-server enable
Rack1R5(config)#ftp-server topdir flash:
Rack1R5#debug ftpserver
FTPServer Events debugging is on
```

Perform a file transfer:

```
Rack1R3#copy ftp: null:
Address or name of remote host []? 149.1.254.5
Source filename []? test.txt
Accessing ftp://149.1.254.5/test.txt...
Loading test.txt !
[OK - 2643/4096 bytes]

2643 bytes copied in 5.184 secs (510 bytes/sec)

Rack1R5#
%FTPSERVER-6-NEWCONN: FTP Server - new connection made.
-Process= "TCP/FTP Server", ipl= 0, pid= 66FTPSRV_DEBUG:FTP Server file
path: 'flash:/'
FTPSRV_DEBUG:FTPSERVER: Input path passed Top-dir(flash:) test.
FTPSRV_DEBUG:FTP Server file path: 'flash:/'
FTPSRV_DEBUG:FTPSERVER: Input path passed Top-dir(flash:) test.
FTPSRV_DEBUG:(REPLY)  220
FTPSRV_DEBUG:Rack1R5 IOS-FTP server (version 1.00) ready.
FTPSRV_DEBUG:FTP Server Command received: 'USER R3FTP
FTPSRV_DEBUG:(REPLY)  331
FTPSRV_DEBUG:Password required for 'R3FTP'.
FTPSRV_DEBUG:FTP Server Command received: 'PASS CISCO
FTPSRV_DEBUG:(REPLY)  230
FTPSRV_DEBUG:Logged in.
FTPSRV_DEBUG:FTP Server Command received: 'TYPE I
FTPSRV_DEBUG:(REPLY)  200
FTPSRV_DEBUG:Type set to I.
FTPSRV_DEBUG:FTP Server Command received: 'PASV
FTPSRV_DEBUG:(REPLY)  227
FTPSRV_DEBUG:Entering Passive Mode (149,1,254,5,218,53)
FTPSRV_DEBUG:FTP Server Command received: 'RETR test.txt
FTPSRV_DEBUG:FTP Server file path: 'flash:/test.txt'
FTPSRV_DEBUG:FTPSERVER: Input path passed Top-dir(flash:) test.
FTPSRV_DEBUG:(REPLY)  150
FTPSRV_DEBUG:Opening BINARY mode data connection for test.txt (2643
bytes).
FTPSRV_DEBUG:(REPLY)  226
FTPSRV_DEBUG:Transfer complete.
FTPSRV_DEBUG:FTP Server Command received: 'QUIT
FTPSRV_DEBUG:(REPLY)  221
FTPSRV_DEBUG:Goodbye.
FTPSRV_DEBUG:FTP Server Exit Signal for Process(66). Freeing instance
data.
```


Task 7.2 Solution

R3 and R6:

```
logging rate-limit console 5
logging console informational
logging monitor informational
logging buffered warnings
```

Task 7.2 Verification

Verify logging configuration:

```
RSRack1R3#show logging | section Syslog
Syslog logging: enabled (11 messages dropped, 0 messages rate-limited,
                0 flushes, 0 overruns, xml disabled, filtering
disabled)
    Console logging: level informational, 90 messages logged, xml
disabled,
                     filtering disabled
    Monitor logging: level informational, 0 messages logged, xml
disabled,
                     filtering disabled
    Buffer logging: level warnings, 0 messages logged, xml disabled,
                    filtering disabled
    Logging Exception size (4096 bytes)
    Count and timestamp logging messages: disabled
```

Task 7.3 Solution

Assign a rotary group to VTY 0:

R5:

```
!
line vty 0
 rotary 1
```

Try telneting to the specific line (VTY 0):

```
Rack1R5#telnet 150.1.5.5 3001
Trying 150.1.5.5, 3001 ... Open


User Access Verification

Password:
Rack1R5>telnet 150.1.5.5 3001
Trying 150.1.5.5, 3001 ... Open
Try back in 10 minutes
[Connection to 150.1.5.5 closed by foreign host]
```


Task 7.4 Solution

R5:

```
banner exec ^R5 is for use by authorized users only. You are on line
number: $(line)^
```

Task 7.4 Verification

Telnet to R5:

```
Rack1R5#telnet 150.1.5.5
Trying 150.1.5.5 ... Open


User Access Verification

Password: <cisco>
R5 is for use by authorized users only. You are on line
number: 130
```

Task 7.5 Solution

R3:

```
interface Tunnel0
 ip unnumbered Loopback0
 keepalive 10 3
 tunnel source 149.1.123.3
 tunnel destination 149.1.123.1
```

R1:

```
interface Tunnel0
 ip unnumbered Loopback0
 keepalive 10 3
 tunnel source 149.1.123.1
 tunnel destination 149.1.123.3

interface FastEthernet0/0
standby 1 ip 149.1.127.254
standby 1 priority 110
standby 1 preempt
standby 1 name HSRP
standby 1 track Tunnel0 20
```

R2:

```
interface FastEthernet0/0
standby 1 ip
standby 1 preempt
standby 1 name HSRP
```


Task 7.5 Verification

```
RSRack1R2#show standby
FastEthernet0/0 - Group 1
  State is Standby
    3 state changes, last state change 00:00:03
  Virtual IP address is 149.1.127.254 (learnt)
  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 2.387 secs
  Preemption enabled
  Active router is 149.1.127.1, priority 110 (expires in 7.095 sec)
  Standby router is local
  Priority 100 (default 100)
  IP redundancy name is "HSRP" (cfgd)

RSRack1R2#show standby brief
                     P indicates configured to preempt.
                     |
Interface   Grp  Prio P State    Active          Standby         Virtual IP
Fa0/0       1    100  P Standby  149.1.127.1     local       149.1.127.254

Rack1R3(config)#interface serial1/0
Rack1R3(config-if)#shutdown

Rack1R1#debug standby events
HSRP Events debugging is on

HSRP: Fa0/0 Grp 1 Track 1 object changed, state Up -> Down
HSRP: Fa0/0 Grp 1 Priority 110 -> 90
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state
to down
HSRP: Fa0/0 Grp 1 Ignoring Coup (100/149.1.127.2 < 110/149.1.127.1)
HSRP: Fa0/0 Grp 1 Hello  in  149.1.127.2 Active  pri 100 vIP
149.1.127.254
HSRP: Fa0/0 Grp 1 Active router is 149.1.127.2, was local
HSRP: Fa0/0 Grp 1 Standby router is unknown, was 149.1.127.2
HSRP: Fa0/0 Grp 1 Active: g/Hello rcvd from higher pri Active router
(100/149.1.127.2)
HSRP: Fa0/0 Grp 1 Active -> Speak
%HSRP-6-STATECHANGE: FastEthernet0/0 Grp 1 state Active -> Speak
HSRP: Fa0/0 Grp 1 Redundancy "HSRP" state Active -> Speak
HSRP: Fa0/0 API MAC address update

Rack1R1#show interface tu0
Tunnel0 is up, line protocol is down
  Hardware is Tunnel
  Interface is unnumbered. Using address of Loopback0 (150.1.1.1)
  MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive set (10 sec), retries 3
  Tunnel source 149.1.123.1, destination 149.1.123.3
  Tunnel protocol/transport GRE/IP, key disabled, sequencing disabled
```


Task 7.6 Solution

R1 and R2:

```
interface FastEthernet0/0
ip helper-address 149.1.5.50 redundancy HSRP
```

Task 7.6 Verification

Verify VRG configuration:

```
Rack1R2#show ip helper-address
Interface                  Helper-Address  VPN VRG Name        VRG State
FastEthernet0/0            149.1.5.50      0   HSRP              STANDBY

Rack1R1#show ip helper-address
Interface                  Helper-Address  VPN VRG Name        VRG State
FastEthernet0/0            149.1.5.50      0   HSRP                ACTIVE
```

Task 7.7 Solution

R5:

```
!
! Find out the default values using show interaces dampening
!
interface FastEthernet 0/1
 dampening 5 2000 1000 15
```

Task 7.7 Verification

```
Rack1R5#show interfaces dampening
FastEthernet0/1
  Flaps Penalty    Supp ReuseTm   HalfL  ReuseV   SuppV  MaxSTm    MaxP
Restart
      0       0   FALSE       0       5    2000    1000      15   16000
```

Task 7.8 Solution

R5:

```
!
! This setting will make R5 bring the interface down as soon
! as the carrier signal from the switch port is lost.
!
interface FastEthernet0/0
 carrier-delay msec 0
```


Task 7.9 Solution

R5:

```
!
! Time based redistribution wont work with BGP in newer IOS versions
! Thus we resort to creating static routes that track certain objects
! The object's state depend on the rachability based on the time-ranged
! access-list.
!
time-range WEEKDAY
 periodic weekday 9:00 to 17:59
!
no access-list 100
access-list 100 permit ip any any time-range WEEKDAY
!
interface Loopback1000
 ip address 6.6.6.6 255.255.255.255
 ip access-group 100 in
!
ip sla 1
 icmp-echo 6.6.6.6
 timeout 100
 frequency 1
!
ip sla schedule 1 start now life forever
!
track 1 rtr 1 reachability
!
ip route 150.1.66.0 255.255.255.0 Loopback0 6.6.6.254 tag 100 track 1
!
router bgp 100
 redistribute static route-map STATIC_TO_BGP
!
route-map STATIC_TO_BGP permit 10
 match tag 100
```


Task 7.9 Verification

Check the prefix advertisement while the time-range is not active.

```
RSRack1R6#show ip access-lists 100
Extended IP access list 100
    10 permit ip any any time-range WEEKDAY (inactive)

RSRack1R6#show ip bgp 150.1.66.0
% Network not in table

RSRack1R6#show track 1
Track 1
  IP SLA 1 reachability
  Reachability is Down
    3 changes, last change 00:00:29
  Latest operation return code: Timeout
  Tracked by:
    STATIC-IP-ROUTING 0
```

Change the clock to a moment within the “valid” time-range:

```
RSRack1R6#clock set 11:00:00 Jan 1 2020
%TRACKING-5-STATE: 1 ip sla 1 reachability Down->Up

RSRack1R6#show track
Track 1
  IP SLA 1 reachability
  Reachability is Up
    4 changes, last change 00:00:06
  Latest operation return code: OK
  Latest RTT (millisecs) 1
  Tracked by:
    STATIC-IP-ROUTING 0

RSRack1R6#show ip access-list 100
Extended IP access list 100
    10 permit ip any any time-range WEEKDAY (active) (12 matches)

RSRack1R6#show ip bgp 150.1.66.0
BGP routing table entry for 150.1.66.0/24, version 14
Paths: (1 available, best #1, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
        2
  Local
    0.0.0.0 from 0.0.0.0 (150.1.6.6)
      Origin incomplete, metric 0, localpref 100, weight 32768, valid,
sourced, best
```



```
RSRack1R6#clock set 19:00:00 Jan 1 2020
%TRACKING-5-STATE: 1 ip sla 1 reachability Up->Down

RSRack1R6#show track
Track 1
  IP SLA 1 reachability
  Reachability is Down
    3 changes, last change 00:00:22
  Latest operation return code: Timeout
  Tracked by:
    STATIC-IP-ROUTING 0

RSRack1R6#show ip access-lists 100
Extended IP access list 100
    10 permit ip any any time-range WEEKDAY (inactive) (91 matches)

RSRack1R6#show ip bgp 150.1.66.0
% Network not in table
```

Task 8.1 Solution

```
R3:
interface Serial1/0
 frame-relay traffic-shaping
 frame-relay interface-dlci 302
  class 64Kbps
!
map-class frame-relay 64Kbps
 frame-relay cir 64000
 frame-relay bc 8000
 frame-relay be 4000
```

Task 8.1 Verification

Verify traffic-shaping parameters:

```
Rack1R3#show frame-relay pvc 302

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

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

  input pkts 438           output pkts 589          in bytes 26377
  out bytes 36093          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 421       out bcast bytes 25134
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 01:14:14, last time pvc status changed 01:14:14
  cir 64000     bc 8000      be 4000      byte limit 1500  interval 125
  mincir 32000     byte increment 1000  Adaptive Shaping none
  pkts 13        bytes 1107      pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/40, 0 drop, 0 dequeued
```

Task 8.2 Solution

```
R3:
interface Serial1/0
 frame-relay interface-dlci 301
 class FULL_T1
!
map-class frame-relay FULL_T1
 frame-relay cir 1536000
```


Task 8.2 Verification

Verify traffic-shaping parameters:

```
Rack1R3#show frame-relay pvc 301

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

DLCI = 301, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial1/0

  input pkts 425           output pkts 559          in bytes 22636
  out bytes 34890          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 396       out bcast bytes 23114
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 01:16:24, last time pvc status changed 00:04:04
  cir 1536000   bc 1536000   be 0         byte limit 6144   interval 32
  mincir 768000    byte increment 6144  Adaptive Shaping none
  pkts 36        bytes 2913      pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: fifo
  Output queue 0/40, 0 drop, 0 dequeued
```

Task 8.3 Solution

R4:

```
map-class frame-relay SHAPE_512
 frame-relay cir 512000
 frame-relay bc 5120
 frame-relay fair-queue 64 128 4
!
interface Serial 0/0/0
 frame-relay traffic-shaping
 frame-relay interface-dlci 405
  class SHAPE_512
 ip rsvp bandwidth 256 64
 ip rsvp resource-provider wfq pvc
```

R5:

```
map-class frame-relay SHAPE_512
 frame-relay cir 512000
 frame-relay bc 5120
 frame-relay fair-queue 64 128 4
!
interface Serial 0/0/0
 frame-relay traffic-shaping
 frame-relay interface-dlci 504
  class SHAPE_512
 ip rsvp bandwidth 256 64
 ip rsvp resource-provider wfq pvc
```


Task 8.3 Verification

Check Per-VC queueing:

```
RSRack1R4#show frame-relay pvc 405

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

DLCI = 405, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0/0

  input pkts 353           output pkts 509          in bytes 28255
  out bytes 33827          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 470       out bcast bytes 29820
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  pvc create time 02:26:45, last time pvc status changed 02:20:13
  cir 512000    bc 5120      be 0         byte limit 640    interval 10
  mincir 256000    byte increment 640   Adaptive Shaping none
  pkts 40        bytes 3454      pkts delayed 0         bytes delayed 0
  shaping inactive
  traffic shaping drops 0
  Queueing strategy: weighted fair
  Current fair queue configuration:
   Discard     Dynamic      Reserved
   threshold   queue count  queue count
    64          128          8
  Output queue size 0/max total 600/drops 0
```

Check for RSVP neighbors:

```
RSRack1R4#show ip rsvp neighbor
Neighbor         Encapsulation  Time since msg rcvd/sent
149.1.254.5      Raw IP         00:00:26   00:00:27
```

Pitfall

* Neighbors inactive for more than one hour are not shown.Use the "inactive" keyword to display them.

Task 8.4 Solution

R4:

```
ip rsvp sender-host 150.1.5.5 150.1.4.4 udp 16384 16384 64 8
```

R5:

```
ip rsvp reservation-host 150.1.5.5 150.1.4.4 UDP 16384 16384 FF LOAD 64
8
```

Task 8.4 Verificaiton

Check the RSVP sender and reservation status (notice we check the sender at the “reserving” host and reservation at the “sending” host).

```
RSRack1R5#show ip rsvp sender detail
PATH:
  Destination 150.1.5.5, Protocol_Id 17, Don't Police , DstPort 16384
  Sender address: 150.1.4.4, port: 16384
  Path refreshes:
    arriving: from PHOP 149.1.254.4 on Se0/0/0 every 30000 msecs
  Traffic params - Rate: 64K bits/sec, Max. burst: 8K bytes
    Min Policed Unit: 0 bytes, Max Pkt Size 2147483647 bytes
  Path ID handle: 01000402.
  Incoming policy: Accepted. Policy source(s): Default
  Status: Proxy-terminated

RSRack1R5#show ip rsvp reservation detail

RSVP Reservation. Destination is 150.1.5.5, Source is 150.1.4.4,
  Protocol is UDP, Destination port is 16384, Source port is 16384
  Next Hop: none
  Reservation Style is Fixed-Filter, QoS Service is Controlled-Load
  Resv ID handle: 01000403.
  Created: 08:26:57 UTC Thu Jun 3 2010
  Average Bitrate is 64K bits/sec, Maximum Burst is 8K bytes
  Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Status: Proxied
  Policy: Forwarding. Policy source(s): Default

RSRack1R4#show ip rsvp reservation detail

RSVP Reservation. Destination is 150.1.5.5, Source is 150.1.4.4,
  Protocol is UDP, Destination port is 16384, Source port is 16384
  Next Hop: 149.1.254.5 on Serial0/0/0
  Reservation Style is Fixed-Filter, QoS Service is Controlled-Load
  Resv ID handle: 01000404.
  Created: 08:27:29 UTC Thu Jun 3 2010
  Average Bitrate is 64K bits/sec, Maximum Burst is 8K bytes
  Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Status:
  Policy: Forwarding. Policy source(s): Default
```



```
RSRack1R4#show ip rsvp sender detail
PATH:
  Destination 150.1.5.5, Protocol_Id 17, Don't Police , DstPort 16384
  Sender address: 150.1.4.4, port: 16384
  Path refreshes:
  Traffic params - Rate: 64K bits/sec, Max. burst: 8K bytes
    Min Policed Unit: 0 bytes, Max Pkt Size 2147483647 bytes
  Path ID handle: 01000402.
  Incoming policy: Accepted. Policy source(s): Default
  Status: Proxied
  Output on Serial0/0/0. Policy status: Forwarding. Handle: 03000401
    Policy source(s): Default
```


Lab 20 Solutions

Task 1.1 Solution

SW1-SW4:

```
spanning-tree mode mst
!
spanning-tree mst configuration
 name IESTP
 revision 10
 instance 1 vlan 1, 5, 12, 107
 instance 2 vlan 27, 34, 58
 instance 3 vlan 46, 89, 363
```

Task 1.1 Verification

Verify of MST configuration and VLANs to MST instances mapping:

```
Rack1SW1#show spanning-tree mst configuration
Name      [IESTP]
Revision  10    Instances configured 4

Instance  Vlans mapped
--------  ----------------------------------------------------------------
0         2-4,6-11,13-26,28-33,35-45,47-57,59-88,90-106,108-362,364-
4094
1         1,5,12,107
2         27,34,58
3         46,89,363
-------------------------------------------------------------------------
```



```
Rack1SW1#show spanning-tree mst 1

##### MST1  vlans mapped:   1,5,12,107
Bridge      address 0019.56c8.4e80  priority      32769 (32768 sysid 1)
Root        address 0009.433c.a380  priority      32769 (32768 sysid 1)
            port    Fa0/20          cost          200000    rem hops 19

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/1            Desg FWD 200000    128.3    P2p
Fa0/5            Desg FWD 200000    128.7    P2p
Fa0/13           Desg FWD 200000    128.15   P2p
Fa0/16           Altn BLK 200000    128.18   P2p
Fa0/17           Altn BLK 200000    128.19   P2p
Fa0/18           Altn BLK 200000    128.20   P2p
Fa0/20           Root FWD 200000    128.22   P2p
Po1              Altn BLK 100000    128.56   P2p

Rack1SW1#show spanning-tree mst 2

##### MST2  vlans mapped:   27,34,58
Bridge      address 0019.56c8.4e80  priority      32770 (32768 sysid 2)
Root        address 0009.433c.a380  priority      32770 (32768 sysid 2)
            port    Fa0/20          cost          200000    rem hops 19

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/3            Desg FWD 200000    128.5    P2p
Fa0/13           Desg FWD 200000    128.15   P2p
Fa0/16           Altn BLK 200000    128.18   P2p
Fa0/17           Altn BLK 200000    128.19   P2p
Fa0/18           Altn BLK 200000    128.20   P2p
Fa0/20           Root FWD 200000    128.22   P2p
Po1              Altn BLK 100000    128.56   P2p

Rack1SW1#show spanning-tree mst 3

##### MST3  vlans mapped:   46,89,363
Bridge      address 0019.56c8.4e80  priority      32771 (32768 sysid 3)
Root        address 0009.433c.a380  priority      32771 (32768 sysid 3)
            port    Fa0/20          cost          200000    rem hops 19

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/13           Desg FWD 200000    128.15   P2p
Fa0/16           Altn BLK 200000    128.18   P2p
Fa0/17           Altn BLK 200000    128.19   P2p
Fa0/18           Altn BLK 200000    128.20   P2p
Fa0/20           Root FWD 200000    128.22   P2p
Po1              Altn BLK 100000    128.56   P2p
```


Task 1.2 Solution

SW1:

```
spanning-tree mst 1 priority 0
spanning-tree mst 3 priority 0
!
interface FastEthernet0/20
 spanning-tree mst 2 cost 20000000
!
interface range FastEthernet 0/16 - 18
 spanning-tree mst 2 cost 20000000
!
interface Port-Channel1
 spanning-tree mst 2 cost 20000000
```

SW4:

```
spanning-tree mst 0 priority 0
spanning-tree mst 2 priority 0
```

Task 1.2 Verification

Verify if SW1 and SW4 are STP root bridges:

```
Rack1SW1#show spanning-tree mst 1

##### MST1    vlans mapped:   1,5,12,107
Bridge        address 0019.56c8.4e80  priority      1     (0 sysid 1)
Root          this switch for MST1

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/1            Desg FWD 200000    128.3    P2p
Fa0/5            Desg FWD 200000    128.7    P2p
Fa0/13           Desg FWD 200000    128.15   P2p
Fa0/16           Desg FWD 200000    128.18   P2p
Fa0/17           Desg FWD 200000    128.19   P2p
Fa0/18           Desg FWD 200000    128.20   P2p
Fa0/20           Desg FWD 200000    128.22   P2p
Po1              Desg FWD 100000    128.56   P2p
```



```
Rack1SW1#show spanning-tree mst 3

##### MST3    vlans mapped:   46,89,363
Bridge        address 0019.56c8.4e80  priority      3     (0 sysid 3)
Root          this switch for MST3

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/13           Desg FWD 200000    128.15   P2p
Fa0/16           Desg FWD 200000    128.18   P2p
Fa0/17           Desg FWD 200000    128.19   P2p
Fa0/18           Desg FWD 200000    128.20   P2p
Fa0/20           Desg FWD 200000    128.22   P2p
Po1              Desg FWD 100000    128.56   P2p

Rack1SW4#show spanning-tree mst 0

##### MST0    vlans mapped:   2-4,6-11,13-26,28-33,35-45,47-57,59-
88,90-106
                             108-362,364-4094
Bridge        address 0009.433c.a380  priority      0     (0 sysid 0)
Root          this switch for the CIST
Operational   hello time 2 , forward delay 15, max age 20, txholdcount
6
Configured    hello time 2 , forward delay 15, max age 20, max hops
20

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/4            Desg FWD 200000    128.4    P2p
Fa0/6            Desg FWD 200000    128.6    P2p
Fa0/13           Desg FWD 200000    128.13   P2p
Fa0/14           Desg FWD 200000    128.14   P2p
Fa0/15           Desg FWD 200000    128.15   P2p
Fa0/19           Desg FWD 200000    128.19   P2p
Fa0/20           Desg FWD 200000    128.20   P2p
Fa0/21           Desg FWD 200000    128.21   P2p

Rack1SW4#show spanning-tree mst 2

##### MST2    vlans mapped:   27,34,58
Bridge        address 0009.433c.a380  priority      2     (0 sysid 2)
Root          this switch for MST2

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/4            Desg FWD 200000    128.4    P2p
Fa0/14           Desg FWD 200000    128.14   P2p
Fa0/20           Desg FWD 200000    128.20   P2p
```


Check that VLAN 27 traffic passes on the trunk between SW1 and SW2:

```
Rack1SW1#show spanning-tree vlan 27 interface fastEthernet 0/13

Vlan                Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- ----------------------
MST2                Root FWD 200000    128.15   P2p


Rack1SW2#show spanning-tree vlan 27 interface fastEthernet 0/13

Vlan                Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- ----------------------
MST2                Desg FWD 200000    128.15   P2p

Rack1SW1#show arp | i 142.1.27.2
Internet  142.1.27.2              25   0012.0120.4460  ARPA   Vlan27

Rack1SW1#show mac  address-table | i 0012.0120.4460
  27    0012.0120.4460    DYNAMIC     Fa0/13
```

Task 1.3 Solution

```
SW3:
interface FastEthernet0/17
 spanning-tree mst 3 port-priority 32
!
interface FastEthernet0/18
 spanning-tree mst 3 port-priority 16
```


Task 1.3 Verification

```
Rack1SW2#show spanning-tree mst 3

##### MST3  vlans mapped:   46,89,363
Bridge      address 0019.aa7e.ea00  priority      32771 (32768 sysid 3)
Root        address 0019.56c8.4e80  priority      3     (0 sysid 3)
            port    Fa0/13          cost          200000    rem hops 19

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/4            Desg FWD 200000    128.6    P2p
Fa0/6            Desg FWD 200000    128.8    P2p
Fa0/13           Root FWD 200000    128.15   P2p
Fa0/16           Altn BLK 200000    128.18   P2p
Fa0/17           Altn BLK 200000    128.19   P2p
Fa0/18           Altn BLK 200000    128.20   P2p

Rack1SW2(config)#interface fastEthernet 0/13
Rack1SW2(config-if)#shutdown

Rack1SW2#show spanning-tree mst 3

##### MST3  vlans mapped:   46,89,363
Bridge      address 0019.aa7e.ea00  priority      32771 (32768 sysid 3)
Root        address 0019.56c8.4e80  priority      3     (0 sysid 3)
            port    Fa0/18          cost          300000    rem hops 18

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/4            Desg BLK 200000    128.6    P2p
Fa0/6            Desg BLK 200000    128.8    P2p
Fa0/16           Altn BLK 200000    128.18   P2p
Fa0/17           Altn BLK 200000    128.19   P2p
Fa0/18           Root FWD 200000    128.20   P2p

Rack1SW3#show spanning-tree mst 3

##### MST3  vlans mapped:   46,89,363
Bridge      address 000a.f4f3.e780  priority      32771 (32768 sysid 3)
Root        address 0019.56c8.4e80  priority      3     (0 sysid 3)
            port    Po1             cost          100000    rem hops 19

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------
Fa0/3            Desg FWD 200000    128.3    P2p
Fa0/13           Altn BLK 200000    128.13   P2p
Fa0/14           Altn BLK 200000    128.14   P2p
Fa0/15           Altn BLK 200000    128.15   P2p
Fa0/16           Desg FWD 200000    128.16   P2p
Fa0/17           Desg FWD 200000     32.17   P2p
Fa0/18           Desg FWD 200000     16.18   P2p
Fa0/20           Desg FWD 200000    128.20   P2p
Fa0/24           Desg FWD 2000000   128.24   Shr
Po1              Root FWD 100000    128.65   P2p
```


Task 1.4 Solution

R1:

```
bridge crb
!
interface Serial0/0
 no shutdown
 encapsulation frame-relay
!
interface Serial0/0.102 multipoint
 frame-relay map bridge 102 broadcast
 bridge-group 1
!
interface Serial0/0.103 multipoint
 frame-relay map bridge 103 broadcast
 bridge-group 1
!
bridge 1 protocol ieee
```

R2:

```
bridge irb
!
interface Serial0/0
no ip address
encapsulation frame-relay
!
interface Serial0/0
no ip address
encapsulation frame-relay
frame-relay map bridge 201 broadcast
bridge-group 1
!
interface BVI1
ip address 142.1.23.2 255.255.255.254
!
bridge 1 protocol ieee
bridge 1 route ip
```

R3:

```
bridge irb
!
interface Serial1/0
 no ip address
 encapsulation frame-relay
 frame-relay map bridge 301 broadcast
 bridge-group 1
!
interface BVI1
 ip address 142.1.23.3 255.255.255.254
!
bridge 1 protocol ieee
bridge 1 route ip
```


Task 1.4 Verification

Verify spanning-tree status:

```
Rack1R1#show spanning-tree 1

 Bridge group 1 is executing the ieee compatible Spanning Tree protocol
  Bridge Identifier has priority 32768, address 0000.0cdc.40b8
  Configured hello time 2, max age 20, forward delay 15
  Current root has priority 32768, address 0000.0c03.f370
  Root port is 9 (Serial0/0.102), cost of root path is 647
  Topology change flag not set, detected flag not set
  Number of topology changes 1 last change occurred 00:02:09 ago
          from Serial0/0.102
  Times:  hold 1, topology change 35, notification 2
          hello 2, max age 20, forward delay 15
  Timers: hello 0, topology change 0, notification 0, aging 300

 Port 9 (Serial0/0.102) of Bridge group 1 is forwarding
   Port path cost 647, Port priority 128, Port Identifier 128.9.
   Designated root has priority 32768, address 0000.0c03.f370
   Designated bridge has priority 32768, address 0000.0c03.f370
   Designated port id is 128.4, designated path cost 0
   Timers: message age 2, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   BPDU: sent 16, received 49

 Port 10 (Serial0/0.103) of Bridge group 1 is forwarding
   Port path cost 647, Port priority 128, Port Identifier 128.10.
   Designated root has priority 32768, address 0000.0c03.f370
   Designated bridge has priority 32768, address 0000.0cdc.40b8
   Designated port id is 128.10, designated path cost 647
   Timers: message age 0, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   BPDU: sent 64, received 0
```



```
Rack1R2#show spanning-tree 1

 Bridge group 1 is executing the ieee compatible Spanning Tree protocol
  Bridge Identifier has priority 32768, address 0000.0c03.f370
  Configured hello time 2, max age 20, forward delay 15
  We are the root of the spanning tree
  Topology change flag not set, detected flag not set
  Number of topology changes 1 last change occurred 00:02:36 ago
          from Serial0/0
  Times:  hold 1, topology change 35, notification 2
          hello 2, max age 20, forward delay 15
  Timers: hello 0, topology change 0, notification 0, aging 300

 Port 4 (Serial0/0) of Bridge group 1 is forwarding
    Port path cost 647, Port priority 128, Port Identifier 128.4.
    Designated root has priority 32768, address 0000.0c03.f370
    Designated bridge has priority 32768, address 0000.0c03.f370
    Designated port id is 128.4, designated path cost 0
    Timers: message age 0, forward delay 0, hold 0
    Number of transitions to forwarding state: 1
    BPDU: sent 71, received 3

Rack1R3#show spanning-tree 1

 Bridge group 1 is executing the ieee compatible Spanning Tree protocol
  Bridge Identifier has priority 32768, address 0000.0cdc.07b8
  Configured hello time 2, max age 20, forward delay 15
  Current root has priority 32768, address 0000.0c03.f370
  Root port is 6 (Serial1/0), cost of root path is 8459
  Topology change flag not set, detected flag not set
  Number of topology changes 0 last change occurred 00:02:22 ago
  Times:  hold 1, topology change 35, notification 2
          hello 2, max age 20, forward delay 15
  Timers: hello 0, topology change 0, notification 0, aging 300

 Port 6 (Serial1/0) of Bridge group 1 is forwarding
    Port path cost 7812, Port priority 128, Port Identifier 128.6.
    Designated root has priority 32768, address 0000.0c03.f370
    Designated bridge has priority 32768, address 0000.0cdc.40b8
    Designated port id is 128.10, designated path cost 647
    Timers: message age 3, forward delay 0, hold 0
    Number of transitions to forwarding state: 1
    BPDU: sent 0, received 71
```

Test connectivity:

```
Rack1R3#ping 142.1.23.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 142.1.23.2, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 40/40/40 ms
```


Check bridging table at R1:

```
Rack1R1#show bridge 1 verbose

Total of 300 station blocks, 298 free
Codes: P - permanent, S - self

BG Hash    Address       Action   Interface      VC   Age RX count TX count
 1 0A/0 0000.0c2b.f2f8 forward  Serial0/0.102 102  0       5        4
 1 4B/0 0000.0c09.074c forward  Serial0/0.103 103  0       6        5

Flood ports (BG 1)           RX count    TX count
Serial0/0.102                       0           0
Serial0/0.103                       0           0
```

Task 2.1 Solution

R5:

```
interface FastEthernet0/1
 ip ospf network point-to-point
!
router ospf 1
router-id 150.1.5.5
network 142.1.58.5 0.0.0.0 area 0
network 142.1.5.5 0.0.0.0 area 5
```

SW2:

```
ip routing
!
router ospf 1
router-id 150.1.8.8
network 142.1.58.8 0.0.0.0 area 0
network 150.1.8.8 0.0.0.0 area 0
network 142.1.89.8 0.0.0.0 area 0
!
interface Vlan58
 ip ospf network point-to-point
```


Task 2.1 Verification

Verify OSPF neighbors and interface:

```
Rack1R5#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.8.8         0   FULL/  -        00:00:36    142.1.58.8    FastEthernet0/1
150.1.3.3         0   FULL/  -        00:01:54    142.1.0.3       Serial0/0/0
150.1.4.4         0   FULL/  -        00:01:54    142.1.0.4       Serial0/0/0

Rack1SW2#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.5.5         0   FULL/  -        00:00:32    142.1.58.5      Vlan58

Rack1SW2#show ip ospf interface vlan 58
Vlan58 is up, line protocol is up
  Internet Address 142.1.58.8/24, Area 0
  Process ID 1, Router ID 150.1.8.8, Network Type POINT_TO_POINT, Cost:
1
  Transmit Delay is 1 sec, State POINT_TO_POINT
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:04
  Supports Link-local Signaling (LLS)
  Cisco NSF helper support enabled
  IETF NSF helper support enabled
  Index 1/1, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.5.5
  Suppress hello for 0 neighbor(s)


Rack1R5#show ip route ospf
     142.1.0.0/16 is variably subnetted, 6 subnets, 2 masks
O       142.1.0.4/32 [110/64] via 142.1.0.4, 00:03:07, Serial0/0/0
O       142.1.0.3/32 [110/64] via 142.1.0.3, 00:03:07, Serial0/0/0
O       142.1.89.0/24 [110/2] via 142.1.58.8, 00:02:41, FastEthernet0/1
     150.1.0.0/16 is variably subnetted, 4 subnets, 2 masks
O       150.1.8.8/32 [110/2] via 142.1.58.8, 00:02:41, FastEthernet0/1
O       150.1.4.4/32 [110/65] via 142.1.0.4, 00:03:07, Serial0/0/0
O       150.1.3.3/32 [110/65] via 142.1.0.3, 00:03:07, Serial0/0/0
```


Task 2.2 Solution

R3:

```
interface Serial1/1
 ip ospf authentication
 ip ospf authentication-key CISCO
```

R4:

```
interface Serial0/0/0
 ip ospf authentication
 ip ospf authentication-key CISCO
```

R5:

```
interface Serial0/0/0
 ip ospf authentication
 ip ospf authentication-key CISCO
```

Task 2.2 Verification

Check OSPF neighbors again:

```
Rack1R5#show ip ospf neighbor serial 0/0/0

Neighbor ID     Pri   State           Dead Time   Address         Interface
150.1.3.3         0   FULL/  -        00:01:54    142.1.0.3       Serial0/0/0
150.1.4.4         0   FULL/  -        00:01:38    142.1.0.4       Serial0/0/0

Rack1R5#show ip ospf interface s0/0/0
Serial0/0/0 is up, line protocol is up
  Internet Address 142.1.0.5/24, Area 345
  Process ID 1, Router ID 150.1.5.5, Network Type POINT_TO_MULTIPOINT,
Cost: 64
  Transmit Delay is 1 sec, State POINT_TO_MULTIPOINT
  Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit
5
    oob-resync timeout 120
    Hello due in 00:00:19
  Supports Link-local Signaling (LLS)
  Cisco NSF helper support enabled
  IETF NSF helper support enabled
  Index 2/2, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 2, Adjacent neighbor count is 2
    Adjacent with neighbor 150.1.3.3
    Adjacent with neighbor 150.1.4.4
  Suppress hello for 0 neighbor(s)
  Simple password authentication enabled
```


Check authentication type number:

```
Rack1R5#debug ip ospf packet
OSPF packet debugging is on
Rack1R5#
OSPF: rcv. v:2 t:1 l:52 rid:150.1.4.4
      aid:0.0.1.89 chk:1CC8 aut:1 auk: from Serial0/0/0
```

Task 2.3 Solution

```
R5:
interface FastEthernet0/1
 ip ospf authentication message-digest
 ip ospf message-digest-key 1 md5 0 CISCO

SW2:
interface Vlan58
 ip ospf authentication message-digest
 ip ospf message-digest-key 1 md5 CISCO
```

Task 2.3 Verification

Verify OSPF authentication:

```
Rack1SW2#show ip ospf interface vlan 58
Vlan58 is up, line protocol is up
  Internet Address 142.1.58.8/24, Area 0
  Process ID 1, Router ID 150.1.8.8, Network Type POINT_TO_POINT, Cost:
1
  Transmit Delay is 1 sec, State POINT_TO_POINT
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:02
  Supports Link-local Signaling (LLS)
  Cisco NSF helper support enabled
  IETF NSF helper support enabled
  Index 1/1, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 150.1.5.5
  Suppress hello for 0 neighbor(s)
  Message digest authentication enabled
    Youngest key id is 1
```


Task 2.4 Solution

R3 and R4:

```
router ospf 1
 max-metric router-lsa on-startup 600
```

Task 2.4 Verification

Verify max-metric lsa generation condition. For instance at R3:

```
Rack1R3#show ip ospf
 Routing Process "ospf 1" with ID 150.1.3.3
 Supports only single TOS(TOS0) routes
 Supports opaque LSA
 Supports Link-local Signaling (LLS)
 Supports area transit capability
 Originating router-LSAs with maximum metric
    Condition: on startup for 600 seconds, State: inactive
 Initial SPF schedule delay 5000 msecs
 Minimum hold time between two consecutive SPFs 10000 msecs
 Maximum wait time between two consecutive SPFs 10000 msecs
 Incremental-SPF disabled
 Minimum LSA interval 5 secs
 Minimum LSA arrival 1000 msecs
 LSA group pacing timer 240 secs
 Interface flood pacing timer 33 msecs
 Retransmission pacing timer 66 msecs
 Number of external LSA 0. Checksum Sum 0x000000
 Number of opaque AS LSA 0. Checksum Sum 0x000000
 Number of DCbitless external and opaque AS LSA 0
 Number of DoNotAge external and opaque AS LSA 0
 Number of areas in this router is 1. 1 normal 0 stub 0 nssa
 Number of areas transit capable is 0
 External flood list length 0
```

Task 2.5 Solution

R3:

```
interface Serial1/1
 ip ospf flood-reduction
```

R4 and R5:

```
interface Serial0/0/0
 ip ospf flood-reduction
```


Task 2.5 Verification

Verify interface configuration:

```
Rack1R5#show ip ospf interface serial 0/0/0
Serial0/0/0 is up, line protocol is up
  Internet Address 142.1.0.5/24, Area 345
  Process ID 1, Router ID 150.1.5.5, Network Type POINT_TO_MULTIPOINT,
Cost: 64
  Reduce LSA flooding.
  Transmit Delay is 1 sec, State POINT_TO_MULTIPOINT
  Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit
5
    oob-resync timeout 120
    Hello due in 00:00:24
  Supports Link-local Signaling (LLS)
  Cisco NSF helper support enabled
  IETF NSF helper support enabled
  Index 2/2, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 4 msec, maximum is 4 msec
  Neighbor Count is 2, Adjacent neighbor count is 2
    Adjacent with neighbor 150.1.3.3
    Adjacent with neighbor 150.1.4.4
  Suppress hello for 0 neighbor(s)
  Simple password authentication enabled
```

Check OSPF database:

```
Rack1R5#show ip ospf database | beg Area 345
                Router Link States (Area 345)

Link ID         ADV Router      Age         Seq#       Checksum Link
count
150.1.3.3       150.1.3.3       1     (DNA) 0x8000001B 0x004DAC 4
150.1.4.4       150.1.4.4       1     (DNA) 0x8000001A 0x00C5CD 4
150.1.5.5       150.1.5.5       53          0x8000001C 0x00E0A8 4

                Summary Net Link States (Area 345)

Link ID         ADV Router      Age         Seq#       Checksum
142.1.5.0       150.1.5.5       680         0x80000001 0x007E87
142.1.58.0      150.1.5.5       680         0x80000001 0x00359B
142.1.89.0      150.1.5.5       660         0x80000001 0x00E8C7
150.1.8.8       150.1.5.5       660         0x80000001 0x00AE43
```



```
Rack1R3#show ip ospf database | beg Area 345
                Router Link States (Area 345)

Link ID         ADV Router      Age         Seq#       Checksum Link
count
150.1.3.3       150.1.3.3       89          0x8000001B 0x004DAC 4
150.1.4.4       150.1.4.4       1      (DNA) 0x8000001A 0x00C5CD 4
150.1.5.5       150.1.5.5       1      (DNA) 0x8000001C 0x00E0A8 4

                Summary Net Link States (Area 345)

Link ID         ADV Router      Age         Seq#       Checksum
142.1.5.0       150.1.5.5       717         0x80000001 0x007E87
142.1.58.0      150.1.5.5       717         0x80000001 0x00359B
142.1.89.0      150.1.5.5       697         0x80000001 0x00E8C7
150.1.8.8       150.1.5.5       697         0x80000001 0x00AE43
```

Task 2.6 Solution

R3:

```
router ospf 1
 network 142.1.34.3 0.0.0.0 area 345
```

R4:

```
router ospf 1
 network 142.1.34.4 0.0.0.0 area 345
```

R5:

```
router ospf 1
 distribute-list route-map OSPF_FILTER in
!
access-list 3 permit 142.1.0.3
access-list 34 permit 142.1.34.0
!
route-map OSPF_FILTER deny 10
 match ip address 34
 match ip next-hop 3
!
route-map OSPF_FILTER permit 20
```


Task 2.6 Verification

Check routing table before applying routing-filer:

```
Rack1R5#show ip route 142.1.34.0
Routing entry for 142.1.34.0/24
  Known via "ospf 1", distance 110, metric 65, type intra area
  Last update from 142.1.0.4 on Serial0/0/0, 00:00:14 ago
  Routing Descriptor Blocks:
    142.1.0.4, from 150.1.4.4, 00:00:14 ago, via Serial0/0/0
      Route metric is 65, traffic share count is 1
  * 142.1.0.3, from 150.1.4.4, 00:00:14 ago, via Serial0/0/0
      Route metric is 65, traffic share count is 1
```

After filtering:

```
Rack1R5#show ip route 142.1.34.0
Routing entry for 142.1.34.0/24
  Known via "ospf 1", distance 110, metric 65, type intra area
  Last update from 142.1.0.4 on Serial0/0/0, 00:00:04 ago
  Routing Descriptor Blocks:
  * 142.1.0.4, from 150.1.4.4, 00:00:04 ago, via Serial0/0/0
      Route metric is 65, traffic share count is 1
```

Confirm that other R3 prefixes are reachable:

```
Rack1R5#show ip route 150.1.3.3
Routing entry for 150.1.3.3/32
  Known via "ospf 1", distance 110, metric 65, type intra area
  Last update from 142.1.0.3 on Serial0/0/0, 00:01:36 ago
  Routing Descriptor Blocks:
  * 142.1.0.3, from 150.1.3.3, 00:01:36 ago, via Serial0/0/0
      Route metric is 65, traffic share count is 1
```


Task 2.7 Solution

R3:

```
router eigrp 100
 redistribute rip metric 10000 10 255 1 1500
 redistribute ospf 1 metric 10000 10 255 1 1500
!
router ospf 1
 distance ospf external 121
 redistribute rip subnets
 redistribute eigrp 100 subnets route-map EIGRP->OSPF
!
router rip
 redistribute eigrp 100 metric 1
 redistribute ospf 1 route-map OSPF->RIP
 offset-list 11 out 16 FastEthernet0/1
!
route-map OSPF->RIP permit 10
 set metric 10
!
route-map EIGRP->OSPF permit 10
 set tag 390
!
interface loopback 0
 ip ospf network point-to-point
!
access-list 11 permit 142.1.0.4
```

R4:

```
!
router ospf 1
 distance ospf external 121
 redistribute rip subnets
!
router rip
 redistribute ospf 1 route-map OSPF->RIP
!
route-map OSPF->RIP permit 10
 match tag 390
 set metric 10
!
route-map OSPF->RIP permit 20
 set metric 1
!
interface loopback 0
 ip ospf network point-to-point
```


Task 2.7 Verification

```
Rack1R6#show ip route rip | i FastEthernet0/0
R       142.1.13.0/24 [120/1] via 204.12.1.3, 00:00:00, FastEthernet0/0
R       142.1.13.1/32 [120/1] via 204.12.1.3, 00:00:00, FastEthernet0/0
R       142.1.0.0/24 [120/1] via 204.12.1.3, 00:00:00, FastEthernet0/0
R       142.1.27.0/24 [120/1] via 204.12.1.3, 00:00:00, FastEthernet0/0
R       142.1.23.2/31 [120/1] via 204.12.1.3, 00:00:00, FastEthernet0/0
R       142.1.34.0/24 [120/1] via 204.12.1.3, 00:00:00, FastEthernet0/0
R       150.1.7.0/24 [120/1] via 204.12.1.3, 00:00:00, FastEthernet0/0
R       150.1.2.0/24 [120/1] via 204.12.1.3, 00:00:00, FastEthernet0/0
R       150.1.1.0/24 [120/1] via 204.12.1.3, 00:00:00, FastEthernet0/0

Rack1R6#show ip route rip | i FastEthernet0/1
R       142.1.5.0/24 [120/1] via 142.1.46.4, 00:00:17, FastEthernet0/1
R       142.1.0.5/32 [120/1] via 142.1.46.4, 00:00:17, FastEthernet0/1
R       142.1.0.3/32 [120/1] via 142.1.46.4, 00:00:17, FastEthernet0/1
                     [120/1] via 142.1.46.4, 00:00:17, FastEthernet0/1
                      [120/1] via 142.1.46.4, 00:00:17, FastEthernet0/1
R       142.1.58.0/24 [120/1] via 142.1.46.4, 00:00:17, FastEthernet0/1
R       142.1.89.0/24 [120/1] via 142.1.46.4, 00:00:17, FastEthernet0/1
R       150.1.4.0/24 [120/1] via 142.1.46.4, 00:00:17, FastEthernet0/1
R       150.1.3.0/24 [120/1] via 142.1.46.4, 00:00:17, FastEthernet0/1
R       150.1.8.8/32 [120/1] via 142.1.46.4, 00:00:17, FastEthernet0/1
R       150.1.5.5/32 [120/1] via 142.1.46.4, 00:00:17, FastEthernet0/1
```


Finally, use the following Tcl script to test connectivity:

```
foreach i {
142.1.13.1
150.1.1.1
142.1.27.2
150.1.2.2
142.1.23.2
142.1.13.3
142.1.0.3
150.1.3.3
142.1.23.3
142.1.34.3
204.12.1.3
142.1.0.4
150.1.4.4
142.1.46.4
142.1.34.4
142.1.0.5
142.1.5.5
150.1.5.5
142.1.58.5
54.1.2.6
150.1.6.6
142.1.46.6
204.12.1.6
142.1.27.7
150.1.7.7
150.1.8.8
142.1.58.8
} { puts [ exec "ping $i" ] }
```

Note that VLAN12 is excluded from connectivity test.

.

Task 2.8 Solution

R3:

```
router bgp 300
neighbor 204.12.1.6 route-map EGP_ORIGIN in
neighbor 204.12.1.254 route-map INCOMPLETE_ORIGIN in
!
route-map EGP_ORIGIN permit 10
set origin egp 1
!
route-map INCOMPLETE_ORIGIN permit 10
set origin incomplete
```

R6:

```
router bgp 100
neighbor 204.12.1.3 route-map TO_R3 out
!
route-map TO_R3 permit 10
set ip next-hop peer-address
```

Task 2.8 Verification

```
Rack1R3#show ip bgp q _54$
BGP table version is 24, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*>i28.119.16.0/24   142.1.46.6               0    100      0 100 54 i
*                   204.12.1.6                            0 100 54 e
*                   204.12.1.254             0            0 100 54 ?
*>i28.119.17.0/24   142.1.46.6               0    100      0 100 54 i
*                   204.12.1.6                            0 100 54 e
*                   204.12.1.254             0            0 100 54 ?
*>i114.0.0.0        142.1.46.6               0    100      0 100 54 i
*                   204.12.1.6                            0 100 54 e
*                   204.12.1.254                          0 100 54 ?
*>i115.0.0.0        142.1.46.6               0    100      0 100 54 i
*                   204.12.1.6                            0 100 54 e
*                   204.12.1.254                          0 100 54 ?
*>i116.0.0.0        142.1.46.6               0    100      0 100 54 i
*                   204.12.1.6                            0 100 54 e
*                   204.12.1.254                          0 100 54 ?
*>i117.0.0.0        142.1.46.6               0    100      0 100 54 i
*                   204.12.1.6                            0 100 54 e
*                   204.12.1.254                          0 100 54 ?
*>i118.0.0.0        142.1.46.6               0    100      0 100 54 i
*                   204.12.1.6                            0 100 54 e
*                   204.12.1.254                          0 100 54 ?
*>i119.0.0.0        142.1.46.6               0    100      0 100 54 i
<output omitted>
```



```
Rack1R3#show ip bgp 28.119.16.0
BGP routing table entry for 28.119.16.0/24, version 57
Paths: (3 available, best #1, table Default-IP-Routing-Table)
Flag: 0x860
  Advertised to update-groups:
     1           2
  100 54
    142.1.46.6 (metric 1) from 142.1.34.4 (150.1.4.4)
      Origin IGP, metric 0, localpref 100, valid, internal, best
  100 54
    204.12.1.6 from 204.12.1.6 (150.1.6.6)
      Origin EGP, localpref 100, valid, external
  100 54
    204.12.1.254 from 204.12.1.254 (31.3.0.1)
      Origin incomplete, metric 0, localpref 100, valid, external
```

Task 2.9 Solution

R1:

```
router bgp 200
 neighbor 150.1.3.3 route-map PREPEND out
!
route-map PREPEND permit 10
 set as-path prepend 300
```

Task 2.9 Verification

Verify before applying the configuration:

```
Rack1R3#show ip bgp q _254$
BGP table version is 64, local router ID is 150.1.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 205.90.31.0      150.1.1.1                              0 200 254 ?
*> 220.20.3.0       150.1.1.1                              0 200 254 ?
*> 222.22.2.0       150.1.1.1                              0 200 254 ?
```


Check BGP updates at R3:

```
Rack1R3#debug ip bgp 150.1.1.1 updates
BGP updates debugging is on for neighbor 150.1.1.1 for address family:
IPv4 Unicast
Rack1R3#clear ip bgp 150.1.1.1 soft in

BGP(0): 150.1.1.1 rcv UPDATE w/ attr: nexthop 150.1.1.1, origin ?,
originator 0.0.0.0, path 200 300 254, community , extended community
BGP(0): 150.1.1.1 rcv UPDATE about 205.90.31.0/24 -- DENIED due to: AS-
PATH contains our own AS;
BGP(0): 150.1.1.1 rcv UPDATE about 220.20.3.0/24 -- DENIED due to: AS-
PATH contains our own AS;
BGP(0): 150.1.1.1 rcv UPDATE about 222.22.2.0/24 -- DENIED due to: AS-
PATH contains our own AS;

Rack1R3#show ip bgp q _254$

Rack1R3#
```

Task 2.10 Solution

R3:

```
router eigrp 200
redistribute bgp 300 metric 10000 10 255 1 1500 route-map BGP2IGP
!
router ospf 1
redistribute bgp 300 subnets route-map BGP2IGP
!
router bgp 300
bgp redistribute-internal
neighbor 142.1.34.4 maximum-prefix 1000
neighbor 150.1.1.1 maximum-prefix 1000
neighbor 204.12.1.6 maximum-prefix 1000
neighbor 204.12.1.254 maximum-prefix 1000
!
ip as-path access-list 1 permit ^[0-9]+$
ip as-path access-list 1 permit ^[0-9]+_[0-9]+$
ip as-path access-list 1 permit ^[0-9]+_[0-9]+_[0-9]+$
!
route-map BGP2IGP permit 10
match as-path 1
```


R4:

```
router ospf 1
redistribute bgp 300 subnets route-map BGP2IGP
!
router bgp 300
bgp redistribute-internal
neighbor 142.1.34.3 maximum-prefix 1000
neighbor 142.1.46.6 maximum-prefix 1000
!
ip as-path access-list 1 permit ^[0-9]+$
ip as-path access-list 1 permit ^[0-9]+_[0-9]+$
ip as-path access-list 1 permit ^[0-9]+_[0-9]+_[0-9]+$
!
route-map BGP2IGP permit 10
match as-path 1
```


Task 2.10 Verification

Check for redistributed routes:

```
Rack1R5#show ip route ospf | i E2
O E2 119.0.0.0/8 [110/1] via 142.1.0.4, 00:01:55, Serial0/0/0
O E2 118.0.0.0/8 [110/1] via 142.1.0.4, 00:01:55, Serial0/0/0
O E2 117.0.0.0/8 [110/1] via 142.1.0.4, 00:01:55, Serial0/0/0
O E2 116.0.0.0/8 [110/1] via 142.1.0.4, 00:01:55, Serial0/0/0
O E2 115.0.0.0/8 [110/1] via 142.1.0.4, 00:01:55, Serial0/0/0
O E2 114.0.0.0/8 [110/1] via 142.1.0.4, 00:01:55, Serial0/0/0
O E2    54.1.2.0 [110/20] via 142.1.0.4, 04:01:06, Serial0/0/0
O E2    142.1.13.0/24 [110/20] via 142.1.0.4, 04:01:06, Serial0/0/0
O E2    142.1.13.1/32 [110/20] via 142.1.0.4, 04:01:06, Serial0/0/0
O E2    142.1.27.0/24 [110/20] via 142.1.0.4, 04:01:06, Serial0/0/0
O E2    142.1.23.2/31 [110/20] via 142.1.0.4, 04:01:06, Serial0/0/0
O E2    142.1.46.0/24 [110/20] via 142.1.0.4, 04:01:11, Serial0/0/0
O E2 204.12.1.0/24 [110/20] via 142.1.0.4, 04:01:06, Serial0/0/0
O E2    28.119.17.0 [110/1] via 142.1.0.4, 00:01:55, Serial0/0/0
O E2    28.119.16.0 [110/1] via 142.1.0.4, 00:01:55, Serial0/0/0
O E2    150.1.7.0/24 [110/20] via 142.1.0.4, 04:01:06, Serial0/0/0
O E2    150.1.6.0/24 [110/20] via 142.1.0.4, 04:01:06, Serial0/0/0
O E2    150.1.2.0/24 [110/20] via 142.1.0.4, 04:01:06, Serial0/0/0
O E2    150.1.1.0/24 [110/20] via 142.1.0.4, 04:01:06, Serial0/0/0
```

Check BGP table of R4 for prefixes with long as-path (4 ASes):

```
Rack1R4#show ip bgp
BGP table version is 27, local router ID is 150.1.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop       Metric LocPrf Weight Path
*> 28.119.16.0/24   142.1.46.6                           0 100 54 i
*> 28.119.17.0/24   142.1.46.6                           0 100 54 i
*> 112.0.0.0        142.1.46.6                           0 100 54 50 60 i
*> 113.0.0.0        142.1.46.6                           0 100 54 50 60 i
*> 114.0.0.0        142.1.46.6                           0 100 54 i
*> 115.0.0.0        142.1.46.6                           0 100 54 i
*> 116.0.0.0        142.1.46.6                           0 100 54 i
*> 117.0.0.0        142.1.46.6                           0 100 54 i
*> 118.0.0.0        142.1.46.6                           0 100 54 i
*> 119.0.0.0        142.1.46.6                           0 100 54 i
```


Task 3.1 Solution

R2:

```
interface FastEthernet0/0
 ipv6 address 2002:8E01:3502:27::/64 eui-64
```

R5:

```
interface FastEthernet0/0
 ipv6 address 2002:8E01:505:5::/64 eui-64
```

Task 3.2 Solution

```
ipv6 unicast-routing
!
interface Tunnel0
 ipv6 address 2002:9601:202::2/64
 tunnel source Loopback0
 tunnel mode ipv6ip 6to4
!
ipv6 route 2002::/16 Tunnel0
ipv6 route 2002:8E01:505:5::/64 2002:9601:505::5
```

R5:

```
!
ipv6 unicast-routing
!
interface Tunnel0
 ipv6 address 2002:9601:505::5/64
 tunnel source Loopback0
 tunnel mode ipv6ip 6to4
!
ipv6 route 2002::/16 Tunnel 0

ipv6 route 2002:8E01:3502:27::/64 2002:9601:202::2
```


Tasks 3.1 – 3.2 Verification

Verify IPv6 addressing:

```
Rack1R2#show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::212:1FF:FE20:4460
    2002:8E01:3502:27:212:1FF:FE20:4460
Serial0/0                  [up/up]
Serial0/1                  [down/down]
BVI1                       [up/up]
Loopback0                  [up/up]
Tunnel0                    [up/up]
    FE80::9601:202
    2002:9601:202::2

Rack1R5#show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::21A:2FFF:FE3E:65FC
    2002:8E01:505:5:21A:2FFF:FE3E:65FC
FastEthernet0/1            [up/up]
    unassigned
Serial0/0/0                [up/up]
    unassigned
Serial0/1/0                [administratively down/down]
    unassigned
SSLVPN-VIF0                [up/up]
    unassigned
Loopback0                  [up/up]
    unassigned
Tunnel0                    [up/up]
    FE80::9601:505
    2002:9601:505::5
```


Test tunnel connectivity:

```
Rack1R5#ping 2002:9601:202::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2002:9601:202::2, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 112/113/116
ms

Rack1R5#ping 2002:8E01:3502:27:212:1FF:FE20:4460 source fastEthernet
0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2002:8E01:3502:27:212:1FF:FE20:4460,
timeout is 2 seconds:
Packet sent with a source address of 2002:8E01:505:5:21A:2FFF:FE3E:65FC
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 216/216/216
ms
```


Task 5.1 Solution

R1:

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

R3:

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

R4:

```
ip multicast-routing
!
interface FastEthernet0/0
 ip pim sparse-mode
!
interface FastEthernet0/1
 ip pim sparse-mode
!
interface Loopback0
 ip pim sparse-mode
!
ip pim bsr-candidate Loopback0 0
ip pim rp-candidate Loopback0
```


Task 5.1 Verification

Verify PIM interfaces:

```
Rack1R1#show ip pim interface

Address         Interface            Ver/   Nbr    Query  DR     DR
                                     Mode   Count  Intvl  Prior
192.10.1.1      FastEthernet0/0      v2/S   0      30     1   192.10.1.1
142.1.13.1      Serial0/1            v2/S   1      30     1   0.0.0.0

Rack1R3#show ip pim interface

Address         Interface            Ver/   Nbr    Query  DR     DR
                                     Mode   Count  Intvl  Prior
142.1.34.3      FastEthernet0/0      v2/S   1      30     1   142.1.34.4
142.1.13.3      Serial1/2            v2/S   1      30     1      0.0.0.0

Rack1R4#show ip pim interface

Address         Interface            Ver/   Nbr    Query  DR     DR
                                     Mode   Count  Intvl  Prior
142.1.46.4      FastEthernet0/0      v2/S   0      30     1   142.1.46.4
142.1.34.4      FastEthernet0/1      v2/S   1      30     1   142.1.34.4
150.1.4.4       Loopback0            v2/S   0      30     1    150.1.4.4
```

Check PIM neighbors:

```
Rack1R3#show ip pim neighbor
PIM Neighbor Table
Neighbor         Interface            Uptime/Expires    Ver   DR
Address                                                       Prio/Mode
142.1.34.4       FastEthernet0/0      00:04:40/00:01:30 v2    1 / DR S
142.1.13.1       Serial1/2            00:04:51/00:01:20 v2    1 / S
```

Verify RP mapping:

```
Rack1R3#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/4
  RP 150.1.4.4 (?), v2
    Info source: 150.1.4.4 (?), via bootstrap, priority 0, holdtime 210
         Uptime: 00:00:59, expires: 00:02:34
```



```
Rack1R1#show ip pim rp mapping
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/4
  RP 150.1.4.4 (?), v2
    Info source: 150.1.4.4 (?), via bootstrap, priority 0
         Uptime: 00:01:19, expires: 00:03:11
```

Task 5.2 Solution

R1:

```
interface FastEthernet0/0
ip igmp join-group 231.31.31.31
```

R4:

```
!
ip mroute 0.0.0.0 0.0.0.0 142.1.34.3
```


Task 5.2 Verification

Ping multicast group from R3 and R4:

```
Rack1R3#ping 231.31.31.31 repeat 5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 231.31.31.31, timeout is 2 seconds:

Reply to request 0 from 142.1.13.1, 52 ms
Reply to request 0 from 142.1.13.1, 88 ms
Reply to request 1 from 142.1.13.1, 48 ms
Reply to request 2 from 142.1.13.1, 48 ms
Reply to request 3 from 142.1.13.1, 48 ms
Reply to request 4 from 142.1.13.1, 48 ms

Rack1R4#ping 231.31.31.31 repeat 5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 231.31.31.31, timeout is 2 seconds:

Reply to request 0 from 142.1.13.1, 32 ms
Reply to request 0 from 142.1.13.1, 84 ms
Reply to request 0 from 142.1.13.1, 56 ms
Reply to request 1 from 142.1.13.1, 28 ms
Reply to request 1 from 142.1.13.1, 44 ms
Reply to request 2 from 142.1.13.1, 28 ms
Reply to request 2 from 142.1.13.1, 44 ms
Reply to request 3 from 142.1.13.1, 28 ms
Reply to request 3 from 142.1.13.1, 44 ms
Reply to request 4 from 142.1.13.1, 28 ms
Reply to request 4 from 142.1.13.1, 44 ms
```

Task 6.1 Solution

```
R6:
interface Serial0/0/0
 ip access-group 1 in
!
access-list 1 deny   51.3.0.1 0.0.0.8
access-list 1 deny   51.5.0.1 0.2.0.8
access-list 1 permit any
```

Task 6.1 Verification

```
Rack1R6#show ip interface serial 0/0/0 | section Inbound
  Inbound  access list is 1

Rack1R6#show ip access-lists 1
Standard IP access list 1
    10 deny   51.3.0.1, wildcard bits 0.0.0.8
    20 deny   51.5.0.1, wildcard bits 0.2.0.8
    30 permit any (22 matches)
```


Task 6.2 Solutiuon

R2:

```
interface BVI1
 ip access-group TASK9.2-IN in
 ip access-group TASK9.2-OUT out
!
ip access-list extended TASK9.2-IN
 permit eigrp any any
 permit icmp any host 142.1.27.7 echo
 permit icmp any host 142.1.107.7 echo
 permit icmp any host 142.1.107.10 echo
 permit tcp any host 142.1.27.7 eq telnet
 permit tcp any host 142.1.107.7 eq telnet
 permit tcp any host 142.1.107.10 eq telnet
 permit tcp any host 142.1.27.100 eq www
 permit icmp any any port-unreachable
 permit icmp any any time-exceeded
 permit 41 any any
 evaluate MY-REFLECT
!
ip access-list extended TASK9.2-OUT
 permit tcp any any reflect MY-REFLECT
 permit udp any any reflect MY-REFLECT
 permit icmp host 142.1.27.7 any echo-reply
 permit icmp host 142.1.107.7 any echo-reply
 permit icmp host 142.1.107.10 any echo-reply
```

Task 6.2 Verification

Verify basic connectivity:

```
Rack1R3#ping 142.1.27.7

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 142.1.27.7, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 128/129/132
ms

Rack1R3#ping 142.1.107.7

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 142.1.107.7, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 128/128/129
ms
```



```
Rack1R3#ping 142.1.107.10

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 142.1.107.10, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 128/129/132
ms

Rack1R3#telnet 142.1.27.7
Trying 142.1.27.7 ... Open


User Access Verification

Password:
Rack1SW1>

Rack1R3#telnet 142.1.107.10
Trying 142.1.107.10 ... Open


User Access Verification

Password:
Rack1SW4>

Rack1R3#telnet 142.1.107.7
Trying 142.1.107.7 ... Open


User Access Verification

Password:
Rack1SW1>

Rack1SW1#traceroute 142.1.0.5

Type escape sequence to abort.
Tracing the route to 142.1.0.5

  1 142.1.27.2 0 msec 0 msec 0 msec
  2 142.1.23.3 28 msec 32 msec 28 msec
  3 142.1.34.4 28 msec 28 msec 28 msec
  4 142.1.0.5 56 msec *  56 msec
```



```
Rack1R2#show ip access-lists
Reflexive IP access list MY-REFLECT
     permit udp host 142.1.0.5 eq 33445 host 142.1.27.7 eq 39280 (1
match) (time left 277)
     permit udp host 142.1.0.5 eq 33444 host 142.1.27.7 eq 37641 (1
match) (time left 274)
     permit udp host 142.1.0.5 eq 33443 host 142.1.27.7 eq 36271 (1
match) (time left 274)
     permit udp host 142.1.0.5 eq 33442 host 142.1.27.7 eq 32911 (1
match) (time left 273)
     permit udp host 142.1.0.5 eq 33441 host 142.1.27.7 eq 38458 (1
match) (time left 273)
     permit udp host 142.1.0.5 eq 33440 host 142.1.27.7 eq 36965 (1
match) (time left 273)
     permit udp host 142.1.0.5 eq 33439 host 142.1.27.7 eq 42593 (1
match) (time left 273)
     permit udp host 142.1.0.5 eq 33438 host 142.1.27.7 eq 40298 (1
match) (time left 273)
     permit udp host 142.1.0.5 eq 33437 host 142.1.27.7 eq 33485 (1
match) (time left 273)
     permit tcp host 142.1.23.3 eq 51491 host 142.1.107.7 eq telnet (33
matches) (time left 252)
     permit tcp host 142.1.23.3 eq 25721 host 142.1.107.10 eq telnet
(33 matches) (time left 235)
     permit tcp host 142.1.23.3 eq 20791 host 142.1.27.7 eq telnet (33
matches) (time left 218)
Extended IP access list TASK9.2-IN
    10 permit eigrp any any (435 matches)
    20 permit icmp any host 142.1.27.7 echo (10 matches)
    30 permit icmp any host 142.1.107.7 echo (10 matches)
    40 permit icmp any host 142.1.107.10 echo (12 matches)
    50 permit tcp any host 142.1.27.7 eq telnet (21 matches)
    60 permit tcp any host 142.1.107.7 eq telnet (21 matches)
    70 permit tcp any host 142.1.107.10 eq telnet (21 matches)
    80 permit tcp any host 142.1.27.100 eq www
    90 permit icmp any any port-unreachable (2 matches)
    100 permit icmp any any time-exceeded (6 matches)
    110 permit 41 any any (30 matches)
    120 evaluate MY-REFLECT
Extended IP access list TASK9.2-OUT
    10 permit tcp any any reflect MY-REFLECT (104 matches)
    20 permit udp any any reflect MY-REFLECT (26 matches)
    30 permit icmp host 142.1.27.7 any echo-reply (10 matches)
    40 permit icmp host 142.1.107.7 any echo-reply (10 matches)
    50 permit icmp host 142.1.107.10 any echo-reply (5 matches)
```


Task 6.3 Solution

R1:

```
interface Serial0/1
 encapsulation ppp
 ppp eap identity ROUTER1
 ppp eap password 0 CISCO
```

R3:

```
username ROUTER1 password CISCO
!
interface Serial1/2
 encapsulation ppp
 ppp authentication eap
 ppp eap identity ROUTER3
 ppp eap local
 clock rate 64000
```

Task 6.3 Verification

Test connectivity:

```
Rack1R3#ping 142.1.13.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 142.1.13.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/29/32 ms
```

Verify PPP EAP authentication:

```
Rack1R3#debug ppp authentication
PPP authentication debugging is on

Rack1R3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Rack1R3(config)#interface s1/2
Rack1R3(config-if)#shutdown
%LINK-5-CHANGED: Interface Serial1/2, changed state to administratively
down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/2, changed
state to down
Rack1R3(config-if)#no shutdown
```



```
%LINK-3-UPDOWN: Interface Serial1/2, changed state to up
 Se1/2 PPP: Using default call direction
 Se1/2 PPP: Treating connection as a dedicated line
 Se1/2 PPP: Session handle[55000003] Session id[3]
 Se1/2 PPP: Authorization required
 Se1/2 EAP: O REQUEST  IDENTITY id 3 len 5
 Se1/2 EAP: I RESPONSE IDENTITY id 3 len 12 from "ROUTER1"
 Se1/2 EAP: O REQUEST  MD5 id 4 len 29 from "ROUTER3"
 Se1/2 EAP: I RESPONSE MD5 id 4 len 29 from "ROUTER1"
 Se1/2 PPP: Sent CHAP LOGIN Request
 Se1/2 PPP: Received LOGIN Response PASS
 Se1/2 PPP: Sent LCP AUTHOR Request
 Se1/2 PPP: Sent IPCP AUTHOR Request
 Se1/2 LCP: Received AAA AUTHOR Response PASS
 Se1/2 IPCP: Received AAA AUTHOR Response PASS
 Se1/2 EAP: O SUCCESS id 4 len 4
 Se1/2 PPP: Sent CDPCP AUTHOR Request
 Se1/2 CDPCP: Received AAA AUTHOR Response PASS
 Se1/2 PPP: Sent IPCP AUTHOR Request
 %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/2, changed
state to up
```

Task 6.4 Solution

R3:

```
ip source-track 142.1.34.100
ip source-track address-limit 100
```

Task 6.4 Verification

```
Rack1R3#show ip source-track summary
Address               Bytes    Pkts     Bytes/s     Pkts/s
142.1.34.100              0       0           0          0
```

Task 7.1 Solution

R5:

```
snmp-server enable traps
snmp-server ifindex persist
snmp-server location San Jose, CA US
snmp-server contact CCIE Lab R5
snmp-server host 142.1.5.100 CISCOTRAP
snmp-server host 142.1.58.100 CISCOTRAP
snmp-server community CISCORO RO 11
snmp-server community CISCORW RW 11
!
access-list 11 permit 142.1.58.100
access-list 11 permit 142.1.5.100
access-list 11 deny any log
```


Task 7.1 Verification

Verify basic SNMP configuration:

```
Rack1R5#show snmp
Chassis: FTX1047Z144
Contact: CCIE Lab R5
Location: San Jose, CA US
0 SNMP packets input
    0 Bad SNMP version errors
    0 Unknown community name
    0 Illegal operation for community name supplied
    0 Encoding errors
    0 Number of requested variables
    0 Number of altered variables
    0 Get-request PDUs
    0 Get-next PDUs
    0 Set-request PDUs
    0 Input queue packet drops (Maximum queue size 1000)
2 SNMP packets output
    0 Too big errors (Maximum packet size 1500)
    0 No such name errors
    0 Bad values errors
    0 General errors
    0 Response PDUs
    2 Trap PDUs
SNMP Dispatcher:
   queue 0/75 (current/max), 0 dropped
SNMP Engine:
   queue 0/1000 (current/max), 0 dropped

SNMP logging: enabled
    Logging to 142.1.5.100.162, 1/10, 0 sent, 0 dropped.
    Logging to 142.1.58.100.162, 1/10, 0 sent, 0 dropped.
```



```
Rack1R5# show snmp community

Community name: ILMI
Community Index: cisco0
Community SecurityName: ILMI
storage-type: read-only  active


Community name: CISCOTRAP
Community Index: cisco1
Community SecurityName: CISCOTRAP
storage-type: nonvolatile        active


Community name: CISCORO
Community Index: cisco2
Community SecurityName: CISCORO
storage-type: nonvolatile        active access-list: 11


Community name: CISCORW
Community Index: cisco3
Community SecurityName: CISCORW
storage-type: nonvolatile        active access-list: 11
```

Task 7.2 Solution

R6:

```
snmp-server view cutoff iso included
snmp-server view cutoff ip.21 excluded
snmp-server view cutoff ip.22 excluded
!
snmp-server community CISCORO view cutoff RO 11
snmp-server community CISCORW view cutoff RW 11
```

Task 7.2 Verification

Verify restrictions:

```
Rack1R5#show snmp group
groupname: ILMI                           security model:v1
readview : *ilmi                          writeview: *ilmi
notifyview: <no notifyview specified>
row status: active

groupname: ILMI                           security model:v2c
readview : *ilmi                          writeview: *ilmi
notifyview: <no notifyview specified>
row status: active

groupname: CISCORO                        security model:v1
readview : cutoff                         writeview: <no writeview
specified>
notifyview: <no notifyview specified>
row status: active      access-list: 11

groupname: CISCORO                        security model:v2c
readview : cutoff                         writeview: <no writeview
specified>
notifyview: <no notifyview specified>
row status: active      access-list: 11

groupname: CISCORW                        security model:v1
readview : cutoff                         writeview: cutoff
notifyview: <no notifyview specified>
row status: active      access-list: 11

groupname: CISCORW                        security model:v2c
readview : cutoff                         writeview: cutoff
notifyview: <no notifyview specified>
row status: active      access-list: 11

groupname: CISCOTRAP                      security model:v1
readview : <no readview specified>        writeview: <no writeview
specified>
notifyview: *tv.FFFFFFFF.FFFFFFFF.FFFFFFFF.F
row status: active

groupname: CISCOTRAP                      security model:v2c
readview : <no readview specified>        writeview: <no writeview
specified>
notifyview: *tv.FFFFFFFF.FFFFFFFF.FFFFFFFF.F
row status: active
```


Task 7.3 Solution

R4:

```
tftp-server flash: c3640-js-mz.123-14.T4.bin alias c3600.bin
```

Task 7.4 Solution

```
R4:
username TAC privilege 0 password 0 CISCO
!
privilege exec level 0 show memory
privilege exec level 0 show processes cpu
privilege exec level 0 show processes
privilege exec level 0 show version
privilege exec level 0 show stacks
!
line vty 0 4
login local
```

Task 7.4 Verification

```
Rack1R4#enable 0
Rack1R4>show ?
  aal2              Show commands for AAL2
  call              Show call
  dial-peer         Dial Plan Mapping Table for, e.g. VoIP Peers
  flash:            display information about flash: file system
  gateway           Show status of gateway
  h323              Show H.323 VoIP information
  memory            Memory statistics
  modem             Show modem
  mrcp              MRCP information
  num-exp           Number Expansion (Speed Dial) information
  processes         Active process statistics
  rtsp              Real Time Streaming Protocol information
  settlement        Show status of settlement
  slot0:            display information about slot0: file system
  slot1:            display information about slot1: file system
  stacks            Process stack utilization
  subscription      Subscription information to show
  translation-rule  Show translation rule table
  version           System hardware and software status
```


Task 7.5 Solution

R4:

```
access-list 1 permit 41.194.41.114 20.40.132.137
!
line vty 0 4
access-class 1 in
```

Task 8.1 Solution

R1 – R6, SW1, and SW2:

```
 ip telnet tos 0
```

Task 8.1 Verification

To verify, implement simple logging solution:

R1:

```
!
access-list 100 permit tcp any any eq telnet precedence routine log
access-list 100 permit tcp any any eq telnet log

line vty 0 4
 access-class 100 in
```

Telnet to R1:

```
Rack1R2#telnet 150.1.1.1
Trying 150.1.1.1 ... Open


User Access Verification

Password:
```

Check ACL logging:

```
Rack1R1#
%SEC-6-IPACCESSLOGP: list 100 permitted tcp 142.1.23.2(11000) ->
0.0.0.0(23), 1 packet

Rack1R1#show ip access-lists 100
Extended IP access list 100
    10 permit tcp any any eq telnet precedence routine log (2 matches)
    20 permit tcp any any eq telnet log
```


Task 8.2 Solution

R2:

```
interface Serial0/0
 random-detect dscp-based
```

Task 8.3 Solution

```
R2:
ip cef
!
class-map match-all TASK8.3-CMAP
match access-group name TASK8.3-ACL
!
policy-map TASK8.3-PMAP
class TASK8.3-CMAP
set dscp cs5
!
interface FastEthernet0/0
service-policy input TASK8.3-PMAP
!
ip access-list extended TASK8.3-ACL
permit tcp host 142.1.27.100 eq www any
```

Task 8.3 Verification

To verify task, simulate traffic to the above mentioned server:

SW1:

```
!
interface vlan 27
 ip address 142.1.27.100 255.255.255.0 secondary
```

```
Rack1R5#telnet 142.1.27.100 80
Trying 142.1.27.100, 80 ... Open
GET /

HTTP/1.1 400 Bad Request
Date: Mon, 01 Mar 1993 04:10:57 GMT
Server: cisco-IOS
Connection: close
Accept-Ranges: none

400 Bad Request

[Connection to 142.1.27.100 closed by foreign host]
```



```
Rack1R2#show policy-map interface f0/0
 FastEthernet0/0

  Service-policy input: TASK8.3-PMAP

    Class-map: TASK8.3-CMAP (match-all)
      8 packets, 543 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group name TASK8.3-ACL
      QoS Set
        dscp cs5
          Packets marked 8
```