CCNP ROUTE CASE STUDY

Rushi Patel

Yash Patel

Group 7

2017/11/26

CCNP Route Case Study

Task 1: Logical Setup

1. **Name all devices according to the topology diagram (R1-R4, SW4)**

The snippets below show the name of the corresponding router/switch. Simply write hostname alongside a name for the device to configure a name of the device.

CLI command example:

hostname R1

Router 1	Router 2	Router 3	Router 4	Switch 4
! hostname R1 !	! hostname R2 !	! hostname R3 !	! hostname R4 !	! hostname SW4 !

2. Be sure to shut down any unused ports on the routers and switches. Failure to do so will result in unexpected route selections.

CLI Commands to shut down unused interfaces. To shut down a device go on the device and input "shutdown" to administratively shutdown the interface.

R1	R2	R3	R4	S4
interface gi0/0 shutdown interface gi0/1 shutdown	interface gi0/1 shutdown interface s0/0/0 shutdown	interface gi0/1 shutdown interface s0/0/1 shutdown	interface g0 shutdown interface s0/0/0 shutdown interface s0/0/2 shutdown	int range gi1/0/1-18 shutdown exit int range gi1/0/21-28 shutdown

Below the snippets show the shutdown ports for each router/switch. They are shut down so that they do not show up in route selections.

Router 1	R1#show ip int brief | include down Embedded-Service-Engine0/0 unassigned YES unset administratively down down GigabitEthernet0/0 unassigned YES unset down down GigabitEthernet0/1 unassigned YES unset administratively down down
Router 2	R2#show ip int brief | include down Embedded-Service-Engine0/0 unassigned YES unset administratively down down GigabitEthernet0/1 unassigned YES unset down down Serial0/0/1 unassigned YES unset administratively down down

Router 3	`R3#show ip int brief | include down`
	`Embedded-Service-Engine0/0 unassigned YES unset administratively down down`
	`GigabitEthernet0/1 unassigned YES unset administratively down down`
	`Serial0/0/1 unassigned YES unset administratively down down`	

Router 4	`R4#show ip int brief | include down`
	`GigabitEthernet0/0/0 unassigned YES unset down down`
	`GigabitEthernet0/0/2 unassigned YES unset administratively down down`
	`GigabitEthernet0 unassigned YES unset administratively down down`	

Switch 4	This command will shutdown the unused interfaces in the switch and allow only the needed interfaces to remain active.	`SW4(config)#int range GigabitEthernet1/0/1-18`
		`SW4(config-if-range)#shutdown`
		`SW4(config-if-range)#exit`
		`SW4(config)#int range GigabitEthernet1/0/21-28`
		`SW4(config-if-range)#shutdown`
		`SW4(config-if-range)#exit`

3. Turn on ipv6 unicast-routing on all routers.

Router 1	`ipv6 unicast-routing`
Router 2	`ipv6 unicast-routing`
Router 3	`ipv6 unicast-routing`
Router 4	`ipv6 unicast-routing`

CLI Command in Global Config Mode for all routers:

- Ipv6 unicast-routing

- IPv6 unicast-routing is used to enable IPv6 routing on routers. This command is used for routers to have IPv6 routes in their routing table so later they can be used to route traffic between routers using IPv6.

4. Make a VRF called VRF-OSPF on R3, a VRF called VRF-EIGRP on R2, and a VRF called VRF-INET on R4. Make sure you use the vrf definition command, not the ip vrf command.

For a multiple instance to occur for Router 2, 3, and 4 we must create a VRF. Virtual Routing and Forwarding is created when this command "vrf definition [VRF Name]" is inputted in the CLI. This command enables a router to have a multiple instance.

Router 2	`R2#show ip vrf`
	`Name`
	`VRF-EIGRP`

Router 3	`R3#show ip vrf`
	`Name`
	`VRF-OSPF`

Router 4	`R4#show ip vrf VRF-INET`
	`Name`
	`VRF-INET`

CLI Commands:

R2:

- vrf definition VRF-EIGRP

R3:

- vrf definition VRF-OSPF

R4:

- vrf definition VRF-INET

5. Assign route distinguishers 650xx:y where xx is your 2-digit group number (e.g. 01, 02, 03…10, 11, etc.) and y is the router number (e.g. on R2 y = 2) to your VRFs.

A Route Distinguisher is a unique way of identifying one route from another. Each route in a single VRF is given the same Route Distinguisher so when two routers communicate, it is easy to identify which route belongs to which VRF by its Route Distinguisher.

CLI Commands to configure route distinguisher:

To configure route distinguisher, define it under the initialization of the VRF's.

R2:

- vrf definition VRF-OSPF
  rd 65007:2

R3:

- vrf definition VRF-EIGRP

rd 65007:3

R4:

- vrf definition VRF-INET
  rd 65007:4

VRF route distinguisher initialization on R2, R3 and R4

Router 2	R2#show ip vrf Name Default RD VRF-EIGRP 65007:2
Router 3	R3#show ip vrf Name Default RD VRF-OSPF 65007:3
Router 4	R4#show ip vrf VRF-INET Name Default RD VRF-INET 65007:4

6. Add both the IPv4 and IPv6 address families to each VRF.

Adding address families in VRF's on R2, R3 and R4. These address families are created so that the VRF's are capable of carrying IPv4 and IPv6 addresses for VRF routing.

Router 2	Router 3	Router 4
vrf definition VRF-EIGR rd 65007:2 ! address-family ipv4 exit-address-family ! address-family ipv6 exit-address-family	vrf definition VRF-OSPF rd 65007:3 ! address-family ipv4 exit-address-family ! address-family ipv6 exit-address-family	vrf definition VRF-INET rd 65007:4 ! address-family ipv4 exit-address-family ! address-family ipv6 exit-address-family

7. Assign interfaces to the VRFs as shown in the topology diagram.

CLI Commands to enable interfaces in each VRF.

Router 2	Router 3	Router 4
interface loopback1 vrf forwarding VRF-EIGRP interface loopback2 vrf forwarding VRF-EIGRP interface Gi0/0.20 vrf forwarding VRF-EIGRP	interface loopback0 vrf forwarding VRF-OSPF interface loopback1 vrf forwarding VRF-OSPF interface Gi0/0.10 vrf forwarding VRF- OSPF	interface Gi0/0/1.20 vrf forwarding VRF-INET interface s0/1/1 vrf forwarding VRF-INET

Initializing interfaces participating in VRFs. These interfaces need to be explicitly defined with vrf forwarding so that they are associated with the VRF. If the vrf forwarding is not used, it will not become a part of the vrf and therefore will not participate in routing protocols

Router 2	R2#show ip vrf Name Default RD Interfaces VRF-EIGRP 65007:2 Lo1 Lo2 Gi0/0.20
Router 3	R3#show ip vrf VRF-OSPF Name Default RD Interfaces VRF-OSPF 65007:3 Lo0 Lo1 Gi0/0.10
Router 4	R4#show ip vrf VRF-INET Name Default RD Interfaces VRF-INET 65007:4 Gi0/0/1.20 Se0/1/1

8. Create the VLANs on the switch as indicated in the topology diagram. Gi1/0/19 & Gi1/0/20 should both be set as static trunk links. Set VTP to Transparent mode.

CLI Commands

Switch 4 Global Config Mode:

```
vtp mode transparent

Vlan 10
Vlan 20

interface GigabitEthernet1/0/19
 switchport trunk allowed vlan 10,20
 switchport mode trunk
no shut
exit

interface GigabitEthernet1/0/20
 switchport trunk allowed vlan 10,20
 switchport mode trunk
no shut
exit
```

For the set of two routers that are connected to the switch a, "set encapsulation dot1q 10 [or] 20" command is also needed alongside the initial configuration on the switch. It is needed so that when a gigabit sub-interface sends a frame to its neighbor, the trunk port configured on the switch can look at the frame and determine its destination.

If the gigabit interface is gi0/0.10 on the router
The command is: encapsulation dot1q 10

If the gigabit interface is gi0/0.10 on the router
The command is: encapsulation dot1q 10

There is one switch but there are 2 connections between routers.

One connection is between R3 (VRF) to R2 and the other connection is between R2(VRF) to R3.

Creating "vlan 10 and vlan 20" in global config mode allows vlans to be made on the switch. Switchport mode trunk allows an interface to be part of the trunking protocol and Switchport trunk allowed 10,20 allows vlan 10 and 20 to participate on the trunking interface.

Configuring interface Gi1/0/19 and Gi1/0/20 as trunk links on the switch

```
SW4(config)#do show int trunk

Port        Mode             Encapsulation  Status        Native vlan
Gi1/0/19    on               802.1q         trunking      1
Gi1/0/20    on               802.1q         trunking      1

Port        Vlans allowed on trunk
Gi1/0/19    10,20
Gi1/0/20    10,20
```

Vlan 10 and 20 are created and they are assigned to Gi1/0/19 and Gi1/0/20

```
SW4(config)#do show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Gi1/0/1, Gi1/0/2, Gi1/0/3
                                                Gi1/0/4, Gi1/0/5, Gi1/0/6
                                                Gi1/0/7, Gi1/0/8, Gi1/0/9
                                                Gi1/0/10, Gi1/0/11, Gi1/0/12
                                                Gi1/0/13, Gi1/0/14, Gi1/0/15
                                                Gi1/0/16, Gi1/0/17, Gi1/0/18
                                                Gi1/0/21, Gi1/0/22, Gi1/0/23
                                                Gi1/0/24, Gi1/0/25, Gi1/0/26
                                                Gi1/0/27, Gi1/0/28
10   VLAN0010                         active
20   VLAN0020                         active
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup
```

Setting up VTP mode transparent configuration on Switch 4

```
hostname SW4
!
boot-start-marker
boot-end-marker
!
!
no aaa new-model
switch 1 provision ws-c2960x-24ts-l
!
!
vtp mode transparent
!
```

9. Set the clock rate of each serial link to 64,000 bps on all DCE interfaces.

Setting up clock rate is essential on routers to determine the bandwidth in which data is sent to other devices. We are setting up a clock rate of 64 0000 bps on all DCE interfaces so all routers with DCE serial interfaces will have a clock rate of 64 000 bps. To set clock rate, it needs to be defined under the interface command.

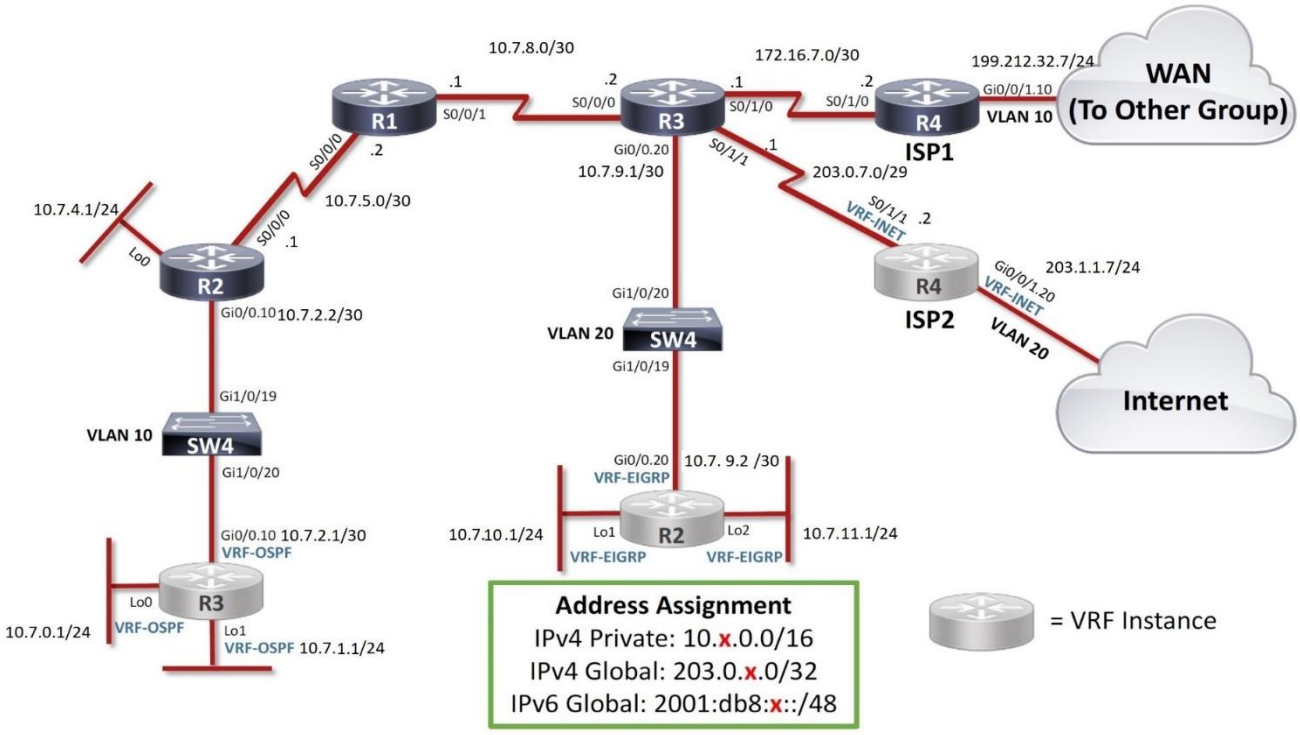
Router 1 Assigning clock rate of 64000 bps on the S0/0/0 DCE interface CLI Command • interface serial0/0/0 clock rate 64000	``` ! interface Serial0/0/0 bandwidth 64 ip address 10.7.5.2 255.255.255.252 ipv6 address FE80::1 link-local ipv6 address 2001:DB8:7:5::2/64 ospfv3 7 ipv4 area 0 ospfv3 7 ipv6 area 0 clock rate 64000 ! ```
Router 2	No DCE interfaces on Router 2

Router 3 Assigned clock rate of 64000 bps on the S0/0/0, S0/1/0, and S0/1/1 DCE interfaces on Router 3 CLI Command • interface serial0/0/0 clock rate 64000 • interface serial0/1/0 clock rate 64000 • interface serial0/1/1 clock rate 64000	! interface Serial0/0/0 ip address 10.7.8.2 255.255.255.252 ipv6 address FE80::3 link-local ipv6 address 2001:DB8:7:8::2/64 clock rate 64000 ! ! interface Serial0/1/0 ip address 172.16.7.1 255.255.255.252 clock rate 64000 ! interface Serial0/1/1 ip address 203.0.7.1 255.255.255.248 ipv6 address FE80::3 link-local ipv6 address 2001:DB8:7:ABCD::1/64 clock rate 64000 !
Router 4	No DCE interfaces on Router 4

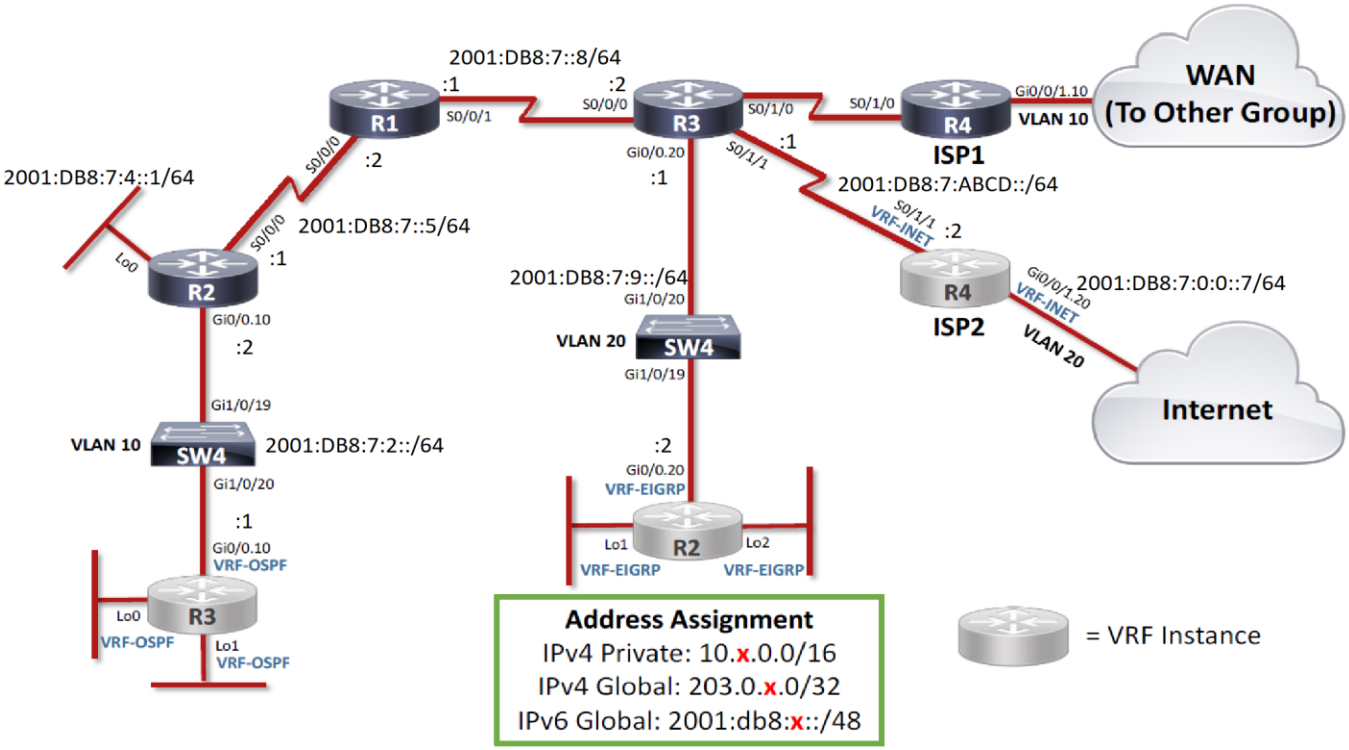
Task 2: Addressing

1) **Where possible, design your addressing scheme in a hierarchical method that allows for easy summarization. Create a diagram with your IPv4 and IPv6 addresses clearly labeled, and include it in your final report.**

IPv4 Addressing - Topology



IPv6 Addressing – Topology



Addressing Table

Router	Interface	IPv4 Address	IPv6 Address
R1	s0/0/0	10.7.5.2/30	2001:DB8:7:5::2/64, FE80::1
	s0/0/1	10.7.8.1/30	2001:DB8:7:8::1/64, FE80::1
R2	s0/0/0	10.7.5.1/30	2001:DB8:7:5::1/64, FE80::2
	g0/0.10	10.7.2.2/30	2001:DB8:7:2::2/64, FE80::2
	Lo0	10.7.4.1/24	2001:DB8:7:4::1/64 , FE80::2
R3	s0/1/1	203.0.7.1/29	2001:DB8:7:ABCD::1/64, FE80::3
	s0/0/0	10.7.8.2/30	2001:DB8:7:8::2/64, FE80::3
	s0/1/0	172.16.7.1/30	----------
	g0/0.20	10.7.9.1/30	2001:DB8:7:9::1/64, FE80::3
R4	s0/1/1	203.0.7.2/29	2001:DB8:7:ABCD::2/64, FE80::4
	s0/1/0	172.16.7.2/30	----------
(VLAN 10)	g0/0/1.10	199.212.32.7/24	----------
R2-VRF	g0/0.20	10.7.9.2/30	2001:DB8:7:9::2/64, FE80::2

	Lo1	**10.7.3.1/24**	**2001:DB8:7:3::1/64 , FE80::2**
	Lo2	**10.7.4.1/24**	**2001:DB8:7:4::1/64, FE80::2**
R3-VRF	**g0/0.10**	**10.7.2.1/30**	**2001:DB8:7:2::1/64, FE80::3**
	Lo0	**10.7.0.1/24**	**2001:DB8:7:0::1/64, FE80::3**
	Lo1	**10.7.1.1/24**	**2001:DB8:7:1::1/64, FE80::3**
R4-VRF	**g0/0/1.20 (vlan 20)**	**203.1.1.7/24**	**2001:DB8:0:0::7/64, FE80::4**
	s0/1/1	**203.0.7.2/29**	**2001:DB8:7:ABCD::2/64, FE80::4**

2) Assign R3 s0/1/1 the IPv4 address 203.0.x.1/29 and R4 s0/1/1 the IPv4 address 203.0.x.2/29.

Setting up a connection between S0/1/1 in R3 to S0/1/1 in R4-VRF by assigning IPv4 addresses.

Assigning IPv4 to s0/1/1 on R3 and R4		CLI Command Upon entering these commands, the interfaces will come up and form connections with the neighbouring router. R3 and R4 will be able to communicate with each other.

Router 3	`R3#show ip int s0/1/1`
	`Serial0/1/1 is up, line protocol is up`
	`  Internet address is 203.0.7.1/29`
		interface s0/1/1
		ip address 203.0.7.1 255.255.255.248
		no shutdown

Router 4	`R4#show ip int s0/1/1`
	`Serial0/1/1 is up, line protocol is up`
	`  Internet address is 203.0.7.2/29`
		interface s0/1/1
		ip address 203.0.7.2 255.255.255.248
		no shutdown

3) Assign R3 s0/1/0 the IPv4 address 172.16.x.1/30 and R4 s0/1/0 the IPv4 address 172.16.x.2/30.

Setting up a connection between S0/1/0 in R3 to S0/1/0 in R4 by assigning IPv4 addresses.

Assigning IPv4 to s0/1/0 on R3 and R4		CLI Command Upon entering these commands, the interfaces will come up and form connections with the neighbouring router. R3 and R4 will be able to communicate with each other.

Router 3	`R3#show ip int s0/1/0`
	`Serial0/1/0 is up, line protocol is up`
	`  Internet address is 172.16.7.1/30`
		interface s0/1/0
		ip address 172.16.7.1 255.255.255.248
		no shutdown

Router 4	R4#show ip int s0/1/0 Serial0/1/0 is up, line protocol is up Internet address is 172.16.7.2/30	interface s0/1/0 ip address 172.16.7.2 255.255.255.248 no shutdown

4) Assign R3 s0/1/1 the IPv6 address 2001:DB8:x:ABCD::1/64 and R4 s0/1/1 the IPv6 address

Setting up a connection between S0/1/1 in R3 to S0/1/1 in R4 by assigning IPv6 addresses.

Assigning IPv6 and link local address to s0/1/1 on R3 and R4		CLI Command Upon entering these commands, the interfaces will come up and form connections with the neighbouring router. R3 and R4 will be able to communicate with each other.
Router 3	R3#show ipv6 int brief Serial0/1/1 [up/up] FE80::3 2001:DB8:7:ABCD::1	interface s0/1/1 ipv6 address 2001:DB8:7:ABCD::1/64 no shutdown
Router 4	R4#show ipv6 int brief Serial0/1/1 [up/up] FE80::4 2001:DB8:7:ABCD::2	interface s0/1/1 ipv6 address 2001:DB8:7:ABCD::2/64 no shutdown

5) Assign R4 Gi0/0/1.10 (VLAN 10) the IPv4 address 199.212.32.x/24.

Assigning IPv4 addresses to sub-interface (Gi0/0/1.10) on R4 for a connection over the WAN link:

	CLI Command • interface gi0/0/1.10 ip address 199.212.32.7 255.255.255.0 no shutdown
Router 4	R4#show ip int brief Interface IP-Address OK? Method Status Protocol GigabitEthernet0/0/0 unassigned YES unset down down GigabitEthernet0/0/1 unassigned YES unset administratively down down Gi0/0/1.10 199.212.32.7 YES manual administratively down down Gi0/0/1.20 203.1.1.7 YES manual administratively down down

6) Assign R4 Gi0/0/1.20 (VLAN 20) the IPv4 address 203.1.1.x/24 and the IPv6 address

Assigning IPv4 addresses to sub-interface (Gi0/0/1.20) on R4-VRF for a IPv4 connection over the internet:

	CLI Command • interface gi0/0/1.10 ip address 203.1.1.7 255.255.255.0 no shutdown

Router 4

```
R4#show ip int brief
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0/0   unassigned      YES unset  down                  down
GigabitEthernet0/0/1   unassigned      YES unset  administratively down down
Gi0/0/1.10             199.212.32.7    YES manual administratively down down
Gi0/0/1.20             203.1.1.7       YES manual administratively down down
```

Assigning IPv6 addresses to sub-interface (Gi0/0/1.20) on R4-VRF for a IPv6 connection over the internet:

Router 4

```
R4#show ipv6 int brief
GigabitEthernet0/0/0    [down/down]
    unassigned
GigabitEthernet0/0/1    [administratively down/down]
    unassigned
Gi0/0/1.10              [administratively down/down]
    unassigned
Gi0/0/1.20              [administratively down/down]
    FE80::4
    2001:DB8::7
```

CLI Command

- interface gi0/0/1.20
  2001:DB8:0:0::7/64
  no shutdown

7) Assign a /24 IPv4 subnet and a /64 IPv6 subnet to each Loopback interface.

In this topology, we used a hierarchical addressing scheme to allocate addresses to each device. For the loopbacks we used /24 IPv4's and /64 IPv6 addresses as per the requirements.

This figure shows the hierarchical addressing scheme applied

P2P Connections	IPv4 Addresses	IPv6 Addresses
R3-VRF Loopbacks	Lo0 - 10.7.**0**.1/24 Lo1 - 10.7.**1**.1/24	2001:DB8:7:**0**::1/64 2001:DB8:7:**1**::1/64
R3-VRF – R2	10.7.**2**.1 - 10.7.2.2/30	2001:DB8:7:**2**::1/64 - 2001:DB8:7:2::2/64
R2 Loopback	Lo0 - 10.7.**4**.1/24	2001:DB8:7:**4**::1/64 - 2001:DB8:7:4::2/64
R2 – R1	10.7.**5**.1 - 10.7.5.2/30	2001:DB8:7:**5**::1/64 - 2001:DB8:7:5::2/64
R1 – R3	10.7.**8**.1 - 10.7.8.2/30	2001:DB8:7:**8**::1/64 - 2001:DB8:7:8::2/64
R3 – R2-VRF	10.7.**9**.1 - 10.7.9.2/30	2001:DB8:7:**9**::1/64 - 2001:DB8:7:9::2/64
R2-VRF Loopback	Lo1 - 10.7.**10**.1/24 Lo2 - 10.7.**11**.1/24	2001:DB8:7:**10**::1/64 2001:DB8:7:**11**::1/64

R2: CLI Command	R2 (VRF): CLI Command	R3 (VRF): CLI Command
interface loopback0 ip address 10.7.4.1 255.255.255.0 ipv6 address 2001:DB8:7:4::1/64	interface loopback1 ip address 10.7.10.1 255.255.255.0 ipv6 address 2001:DB8:7:10::1/64 interface loopback2 ip address 10.7.11.1 255.255.255.0 ipv6 address 2001:DB8:7:11::1/64	interface loopback0 ip address 10.7.0.1 255.255.255.0 ipv6 address 2001:DB8:7:0::1/64 interface loopback1 ip address 10.7.1.1 255.255.255.0 ipv6 address 2001:DB8:7:1::1/64

	/24 IPv4 Loopback	**/64 IPv6 Loopback**

Router 2	R2#show int loopback 0 Loopback0 is up, line protocol is up Hardware is Loopback Internet address is 10.7.4.1/24	R2#show ipv6 int loopback0 Loopback0 is up, line protocol is up IPv6 is enabled, link-local address is FE80::2 No Virtual link-local address(es): Global unicast address(es): 2001:DB8:7:4::1, subnet is 2001:DB8:7:4::/64
Router 2 (VRF)	R2#show int loopback 1 Loopback1 is up, line protocol is up Hardware is Loopback Internet address is 10.7.10.1/24 R2#show int loopback 2 Loopback2 is up, line protocol is up Hardware is Loopback Internet address is 10.7.11.1/24	R2#show ipv6 int loopback 1 Loopback1 is up, line protocol is up IPv6 is enabled, link-local address is FE80::2 No Virtual link-local address(es): Global unicast address(es): 2001:DB8:7:10::1, subnet is 2001:DB8:7:10::/64 R2#show ipv6 int loopback 2 Loopback2 is up, line protocol is up IPv6 is enabled, link-local address is FE80::2 No Virtual link-local address(es): Global unicast address(es): 2001:DB8:7:11::1, subnet is 2001:DB8:7:11::/64
Router 3 (VRF)	R3#show int loopback 0 Loopback0 is up, line protocol is up Hardware is Loopback Internet address is 10.7.0.1/24 R3#show int loopback 1 Loopback1 is up, line protocol is up Hardware is Loopback Internet address is 10.7.1.1/24	R3#show ipv6 int loopback 0 Loopback0 is up, line protocol is up IPv6 is enabled, link-local address is FE80::3 No Virtual link-local address(es): Global unicast address(es): 2001:DB8:7::1, subnet is 2001:DB8:7::/64 R3#show ipv6 int loopback 1 Loopback1 is up, line protocol is up IPv6 is enabled, link-local address is FE80::3 No Virtual link-local address(es): Global unicast address(es): 2001:DB8:7:1::1, subnet is 2001:DB8:7:1::/64

8) Assign a /30 IPv4 subnet and a /64 IPv6 subnet to each point-to-point link between routers. Use the pools shown in the diagram. Give the lower numbered router the first address in each range, and the other router the second address.

Routers are assigned a /30 IPv4 subnet because they are point-to-point links and therefore do not need more than 4 IPv4s (2 addresses for hosts and 2 more for allocating network and broadcast address). For this topology we have assigned in a hierarchical fashion so addresses are summarized as efficiently as possible in OSPF and EIGRP redistribution process. For the addressing scheme we started allocating /30 and /64 hierarchically from R3-VRF to R2-VRF.

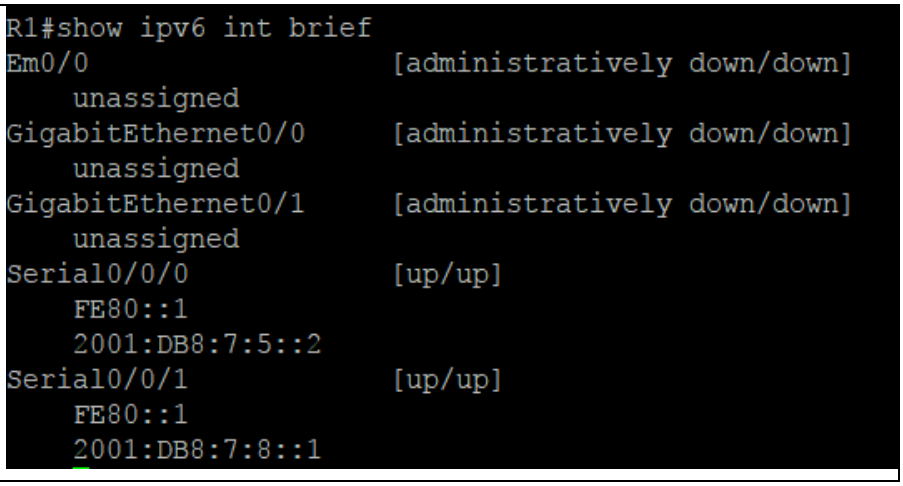
P2P Connections	IPv4 Addresses	IPv6 Addresses
R3-VRF Loopbacks	Lo0 - 10.7.**0**.1/24 Lo1 - 10.7.**1**.1/24	2001:DB8:7:**0**::1/64 2001:DB8:7:**1**::1/64
R3-VRF – R2	10.7.**2**.1 - 10.7.2.2/30	2001:DB8:7:**2**::1/64 - 2001:DB8:7:2::2/64
R2 Loopback	Lo0 - 10.7.**4**.1/24	2001:DB8:7:**4**::1/64 - 2001:DB8:7:4::2/64
R2 – R1	10.7.**5**.1 - 10.7.5.2/30	2001:DB8:7:**5**::1/64 - 2001:DB8:7:5::2/64
R1 – R3	10.7.**8**.1 - 10.7.8.2/30	2001:DB8:7:**8**::1/64 - 2001:DB8:7:8::2/64
R3 – R2-VRF	10.7.**9**.1 - 10.7.9.2/30	2001:DB8:7:**9**::1/64 - 2001:DB8:7:9::2/64
R2-VRF Loopback	Lo1 - 10.7.**10**.1/24 Lo2 - 10.7.**11**.1/24	2001:DB8:7:**10**::1/64 2001:DB8:7:**11**::1/64

The figure above shows the hierarchy that took place in allocating /30 and /64 in the topology. We matched the subnets of IPv6 to the subnets of IPv4 for simplicity and also to easily map of which subnet belongs to which router.

9) Statically configure link-local addresses on each router interface to be FE80::y, where y is the router number (e.g. R3 would have FE80::3 on all of its interfaces).

Since IPv6 is enabled across routers, link-local addresses are needed. For IPv6 features to work, most of the time link-local addresses are needed. One main reason why it is needed is because of neighbor discovery between IPv6 addresses.

R1: CLI Command	R2: CLI Command	R3: CLI Command	R4: CLI Command
Interfaces s0/0/0 ipv6 address FE80::1 interface s0/0/1 ipv6 address FE80::1	Interface gi0/0.10 ipv6 address FE80::2 Interface gi0/0.20 ipv6 address FE80::2 Interfaces s0/0/0 ipv6 address FE80::2 interface loopback0 ipv6 address FE80::2 interface loopback1 ipv6 address FE80::2 interface loopback2 ipv6 address FE80::2	Interface gi0/0.10 ipv6 address FE80::3 Interface gi0/0.20 ipv6 address FE80::3 interface loopback0 ipv6 address FE80::3 interface loopback1 ipv6 address FE80::3 Interfaces s0/0/0 ipv6 address FE80::3 Interfaces s0/1/0 ipv6 address FE80::3 Interfaces s0/1/1 ipv6 address FE80::3	Interface gi0/0/1.10 ipv6 address FE80::4 Interface gi0/0/1.20 ipv6 address FE80::4 Interfaces s0/1/0 ipv6 address FE80::4 Interfaces s0/1/1 ipv6 address FE80::4

Router 1 Assigning Link-local addresses in all interfaces participating in Router 1	

Router 2 Assigning Link-local addresses in all interfaces participating in Router 2	R2#show ipv6 int brief Em0/0 [administratively down/down] unassigned GigabitEthernet0/0 [up/up] unassigned GigabitEthernet0/0.10 [up/up] FE80::2 2001:DB8:7:2::2 GigabitEthernet0/0.20 [up/up] FE80::2 2001:DB8:7:9::2 GigabitEthernet0/1 [administratively down/down] unassigned Serial0/0/0 [up/up] FE80::2 2001:DB8:7:5::1 Serial0/0/1 [administratively down/down] unassigned Loopback0 [up/up] FE80::2 2001:DB8:7:4::1 Loopback1 [up/up] FE80::2 2001:DB8:7:10::1 Loopback2 [up/up] FE80::2 2001:DB8:7:11::1
Router 3 Assigning Link-local addresses in all interfaces participating in Router 3	R3#show ipv6 int brief Em0/0 [administratively down/down] unassigned GigabitEthernet0/0 [up/up] unassigned GigabitEthernet0/0.10 [up/up] FE80::1A8B:9DFF:FEAF:7C68 2001:DB8:7:2::1 GigabitEthernet0/0.20 [up/up] FE80::3 2001:DB8:7:9::1 GigabitEthernet0/1 [administratively down/down] unassigned Serial0/0/0 [up/up] FE80::3 2001:DB8:7:8::2 Serial0/0/1 [administratively down/down] unassigned Serial0/1/0 [up/up] unassigned Serial0/1/1 [up/up] FE80::3 2001:DB8:7:ABCD::1 Loopback0 [up/up] FE80::3 2001:DB8:7::1 Loopback1 [up/up] FE80::3 2001:DB8:7:1::1

Router 4 Assigning Link-local addresses in all interfaces participating in Router 4	R4#show ipv6 int brief GigabitEthernet0/0/0 [down/down] unassigned GigabitEthernet0/0/1 [up/up] unassigned Gi0/0/1.10 [up/up] unassigned Gi0/0/1.20 [up/up] FE80::4 2001:DB8::7 GigabitEthernet0/0/2 [administratively down/down] unassigned Serial0/1/0 [up/up] unassigned Serial0/1/1 [up/up] FE80::4 2001:DB8:7:ABCD::2 GigabitEthernet0 [administratively down/down] unassigned

Task 3: Configure OSPF

1. **Use a process number equal to your group number.**

Configuring OSPFv3 process ID as 7 for each router that participate in opsfv3 process

We are configuring process ID as 7 to specify which interfaces will belong to ospfv3 7 process, since we can have different ospfv3 processes running on each interface

To enable ospfv3 process 7 you just enter global configuration mode and over there just type [router ospfv3 (process ID)] and it will enable ospfv3 process

R1: Process id (PID) is 7	R1#show ospfv3 int brief Interface PID Area AF Cost State Nbrs F/C Se0/0/0 7 0 ipv4 65535 P2P 1/1 Se0/0/0 7 0 ipv6 65535 P2P 1/1
R2: Process id (PID) is 7	R2#show ospfv3 int brief Interface PID Area AF Cost State Nbrs F/C Lo0 7 0 ipv4 1 P2P 0/0 Se0/0/0 7 0 ipv4 65535 P2P 1/1 Gi0/0.10 7 7 ipv4 65535 DR 1/1 Lo0 7 0 ipv6 1 P2P 0/0 Se0/0/0 7 0 ipv6 65535 P2P 1/1 Gi0/0.10 7 7 ipv6 65535 DR 1/1
R3 (VRF): Process id (PID) is 7	R3#show ospfv3 vrf VRF-OSPF int brief Interface PID Area AF Cost State Nbrs F/C Gi0/0.10 7 7 ipv4 65535 DR 0/0 Gi0/0.10 7 7 ipv6 65535 DR 0/0

CLI Commands:

router ospfv3 7

2. Set the bandwidth of all interfaces appropriately.

Configuring each interface's bandwidth to 64,000 bps for those who are participating in ospfv3 process

In this step, we configured our bandwidth on each interface to 64,000 bps to make sure all the routers select accurate path to the destination since bandwidth is use in metric calculation, which is use to identify best path to the destination

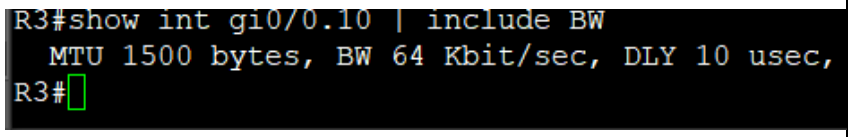
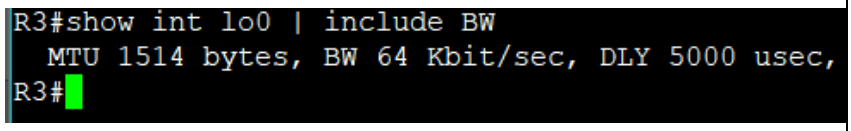
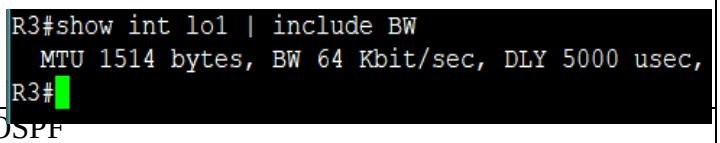
To configure bandwidth enter global mode, type your interface and then just type in [bandwidth 64] command, and it will increase or decrease bandwidth to assigned value

R1:Bandwidth set as 64 on s0/0/0	`R1#show int s0/0/0 | include BW`
	`  MTU 1500 bytes, BW 64 Kbit/sec, DLY 20000 usec,`
	`R1#`

CLI Commands:	R1: interface Serial0/0/0 bandwidth 64

R2:
 	
Bandwidth set as 64 on s0/0/0
 	
Bandwidth set as 64 on Gi0/0.10
 	
Bandwidth set as 64 on loopback0
	`R2#show int s0/0/0 | include BW`
	`  MTU 1500 bytes, BW 64 Kbit/sec, DLY 20000 usec,`
	`R2#`
	
	`R2#show int gi0/0.10 | include BW`
	`  MTU 1500 bytes, BW 64 Kbit/sec, DLY 10 usec,`
	`R2#`
	
	`R2#show int lo0 | include BW`
	`  MTU 1514 bytes, BW 64 Kbit/sec, DLY 5000 usec,`

CLI Commands:	R2: interface Serial0/0/0 bandwidth 64 interface Gi0/0.20

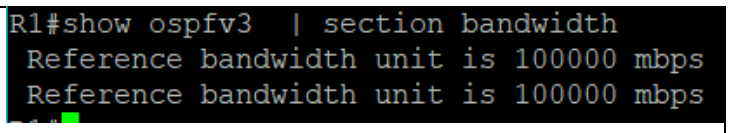
	bandwidth 64 interface Loopback0 bandwidth 64
R3 (VRF): Bandwidth set as 64 on Gi0/0.10 Bandwidth set as 64 on loopback0 Bandwidth set as 64 on loopback1	  
CLI Commands	R3: VRF-OSPF interface Loopback0 bandwidth 64 interface Loopback1 bandwidth 64 interface Gi0/0.10 bandwidth 64

3. Change the OSPF reference bandwidth to 100 Gbps.

Configuring Reference bandwidth on OSPFv3 for each router

We set our reference bandwidth to 100000 mbps on each OSPFv3 router so we can have an increased cost of route to ospf neighbors

To configure reference bandwidth we first enter into ospfv3 process and under that, we type [auto-cost reference-bandwidth 100000]

R1: Reference bandwidth	
CLI Commands:	R1: router ospfv3 7

	auto-cost reference-bandwidth 100000
R2: Reference Bandwidth	R2#show ospfv3 | section bandwidth Reference bandwidth unit is 100000 mbps Reference bandwidth unit is 100000 mbps
CLI Commands:	R2: router ospfv3 7 auto-cost reference-bandwidth 100000
R3 (VRF): Reference Bandwidth	R3#show ospfv3 | section bandwidth Reference bandwidth unit is 100000 mbps Reference bandwidth unit is 100000 mbps
CLI Commands:	R3: VRF-OSPF router ospfv3 7 auto-cost reference-bandwidth 100000

4. Enable OSPFv3 on R1, R2, and R3 for both IPv4 and IPv6 address families, on the interfaces indicated in the diagram. (Note that the commands all start with "ospfv3", not the older "ipv6 router ospf" or "ip ospf" commands).

Enabling OSPFv3 for ipv4 and ipv6 address-families

We are using address-families to make sure we have separate ipv4 and ipv6 address families. All ipv4 related configuration goes under ipv4 address family and all ipv6 related configuration goes under ipv6 address family this way we have more room to keep ipv4 and ipv6 separate, and reduce the complexity when there are two or more ospfv3 processes are running on an interface

To Configure, we first enter into ospfv3 process and then add the [address-family ipv4/ipv6 unicast] to enable them under ospfv3 process. Also to enter vrf address families you must enter [address-family ipv4/ipv6 unicast vrf (vrf-name)]

R1: IPv4 address Family IPv6 address family	R1#show ospfv3 ipv4 int brief Interface PID Area AF Cost State Nbrs F/C Se0/0/0 7 0 ipv4 65535 P2P 1/1 R1#show ospfv3 ipv6 int brief Interface PID Area AF Cost State Nbrs F/C Se0/0/0 7 0 ipv6 65535 P2P 1/1
CLI Commands:	R1: router ospfv3 7 address-family ipv4 unicast exit-address-family address-family ipv6 unicast exit-address-family

<table>
<tr><td>R2:

IPv4 address Family

IPv6 address family</td><td>

```
R2#show ospfv3 ipv4 int brief
Interface    PID   Area            AF         Cost  State Nbrs F/C
Lo0          7     0               ipv4       1     LOOP  0/0
Se0/0/0      7     0               ipv4       65535 P2P   1/1
Gi0/0.10     7     7               ipv4       65535 DR    1/1
```

```
R2#show ospfv3 ipv6 int brief
Interface    PID   Area            AF         Cost  State Nbrs F/C
Lo0          7     0               ipv6       1     LOOP  0/0
Se0/0/0      7     0               ipv6       65535 P2P   1/1
Gi0/0.10     7     7               ipv6       65535 DR    1/1
```

</td></tr>
<tr><td>CLI Commands:</td><td>R2:
router ospfv3 7
address-family ipv4 unicast
exit-address-family
address-family ipv6 unicast
exit-address-family</td></tr>
<tr><td>R3 (VRF):

IPv4 address Family

IPv6 address family</td><td>

```
R3#show ospfv3 ipv4 int brief
R3#show ospfv3 vrf VRF-OSPF ipv4 int brief
Interface    PID   Area            AF         Cost  State Nbrs F/C
Lo0          7     7               ipv4       1     P2P   0/0
Lo1          7     7               ipv4       1     P2P   0/0
Gi0/0.10     7     7               ipv4       65535 BDR   1/1
```

```
R3#show ospfv3 vrf VRF-OSPF ipv6 int brief
Interface    PID   Area            AF         Cost  State Nbrs F/C
Lo0          7     7               ipv6       1     P2P   0/0
Lo1          7     7               ipv6       1     P2P   0/0
Gi0/0.10     7     7               ipv6       65535 BDR   1/1
```

</td></tr>
<tr><td>CLI Commands:</td><td>R3: VRF-OSPF
router ospfv3 7
address-family ipv4 unicast vrf VRF-OSPF
exit-address-family
address-family ipv6 unicast vrf VRF-OSPF
exit-address-family</td></tr>
</table>

5. Use the router number as the router ID (e.g., on R1 use 1.1.1.1). Use this router ID for IPv4, IPv6, and the VRF address families as applicable.

Configuring router-id on each router under ipv4, ipv6 and vrf ipv4/ipv6

We decided to use router-id here in the ospf process to make sure all router are uniquely identified inside the Autonomous System

To configure router-id you must enter ospfv3 process and then enter into ipv4/ipv6 address family or ipv4/ipv6 vrf (vrf-name) address family and type [router-id (32-bit number: 2.2.2.2)] to specify that particular router uniquely in the whole Autonomous System

R1: Router-id (1.1.1.1) **IPv4 address family** **IPv6 address family**	R1#show ospfv3 summary OSPFv3 7 address-family ipv4 (router-id 1.1.1.1) OSPFv3 7 address-family ipv6 (router-id 1.1.1.1)
CLI Commands:	R1: router ospfv3 7 address-family ipv4 unicast router-id 1.1.1.1 exit-address-family address-family ipv6 unicast router-id 1.1.1.1 exit-address-family
R2: Router-id (2.2.2.2) **IPv4 address family** **IPv6 address family**	R2#show ospfv3 summary-prefix OSPFv3 7 address-family ipv4 (router-id 2.2.2.2) OSPFv3 7 address-family ipv6 (router-id 2.2.2.2)
CLI Command:	R2: router ospfv3 7 address-family ipv6 unicast router-id 2.2.2.2 exit-address-family address-family ipv6 unicast router-id 2.2.2.2 exit-address-family
R3 (VRF): Router-id (3.3.3.3) **IPv4 address family** **IPv6 address family**	R3#show ospfv3 vrf VRF-OSPF summary-prefix OSPFv3 7 address-family ipv4 vrf VRF-OSPF (router-id 3.3.3.3) OSPFv3 7 address-family ipv6 vrf VRF-OSPF (router-id 3.3.3.3)
CLI Commands:	R3: router ospfv3 7 address-family ipv4 unicast vrf VRF-OSPF router-id 3.3.3.3 exit-address-family address-family ipv6 unicast vrf VRF-OSPF router-id 3.3.3.3 exit-address-family

6. Change the network type on the loopback interfaces so that the routes are advertised with the correct subnet mask.

7.

Configure network type on all loopbacks under ospfv3 process to point-to-point

Purpose of point-to-point is to make sure that address has advertised with right subnet instead of /32

To configure network type to point-to-point you can just go under a loopback interface you want as point-to-point and type in (ospfv3 network point-to-point) command this will advertised addresses with right subnet instead of /32

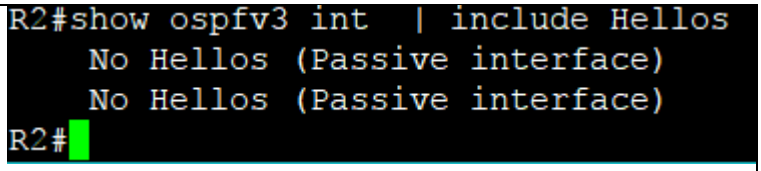
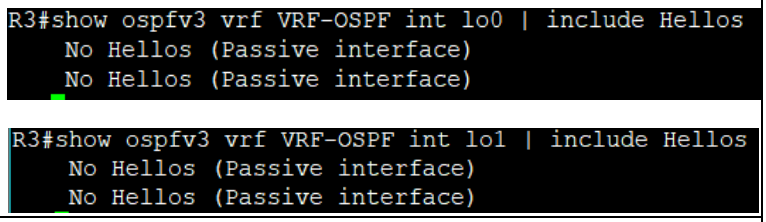
R2: Loopback0 **Proof of network type to point-to-point**	R2#show ospfv3 int lo0 | include Type Network Type POINT_TO_POINT, Cost: 1 Network Type POINT_TO_POINT, Cost: 1 R2#
CLI Commands:	R2: interface Loopback0 ospfv3 network point-to-point exit
R3 (VRF): Loopback0 and Loopback1 **Proof of network type to point-to-point**	R3#show ospfv3 vrf VRF-OSPF int lo0 | include Type Network Type POINT_TO_POINT, Cost: 1 Network Type POINT_TO_POINT, Cost: 1 R3#show ospfv3 vrf VRF-OSPF int lo1 | include Type Network Type POINT_TO_POINT, Cost: 1 Network Type POINT_TO_POINT, Cost: 1
CLI Commands:	R3: VRF-OSPF interface Loopback0 ospfv3 network point-to-point exit interface Loopback1 ospfv3 network point-to-point exit

7. Configure all Loopback interfaces as passive.

Configuring each loopback interface as a passive interface in ospfv3

We are configuring passive-interface on each loopback since we are not forming neighbor adjacency between/using loopbacks. We set loopback interfaces as passive to prevent routing updates from being sent out by loopbacks to all of its neighbors and also, to reduce consumption of bandwidth by sending out those updates

To configure an interface to passive you should enter ospfv3 process then, enter into ipv4/ipv6 address families, and type in passive-interface (interface)

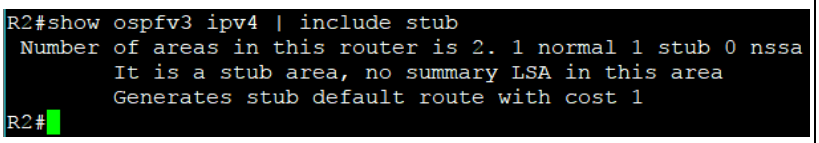
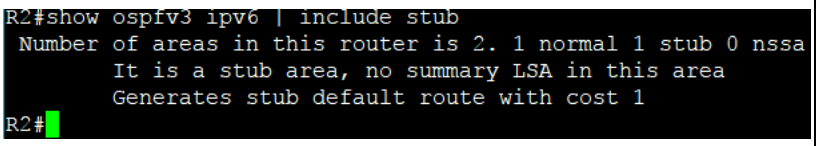
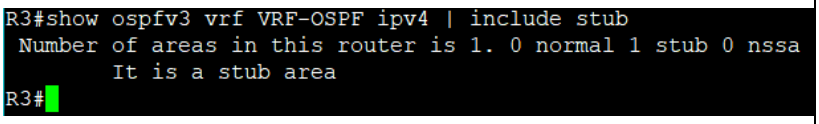
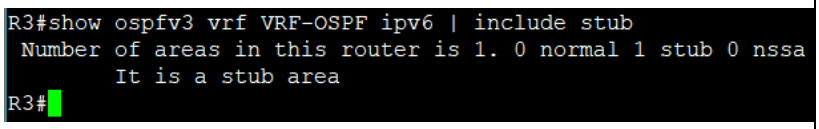
R2:Proof of all loopbacks to passive **Loopback0**	
CLI Commands:	R2: router ospfv3 7 address-family ipv4 unicast vrf VRF-OSPF passive-interface Loopback0 exit-address-family address-family ipv6 unicast vrf VRF-OSPF passive-interface Loopback0 exit-address-family exit
R3 (VRF): Proof of all loopbacks to passive **Loopback0** **Loopback1**	
CLI Commands:	R3: router ospfv3 7 address-family ipv4 unicast vrf VRF-OSPF passive-interface Loopback0 passive-interface Loopback1 exit-address-family address-family ipv6 unicast vrf VRF-OSPF passive-interface Loopback0 passive-interface Loopback1 exit-address-family

8. Configure area x as a totally stubby area for both IPv4 and IPv6.

Configuring area 7 as totally stubby area in both ipv4 and ipv6 address-families

In this step, we are configuring a totally stubby area on R3-vrf and R2 to only allow intra-area LSAs to be exchanged and, to block any external type 5 LSAs, summary type 3 LSAs, and type 4 LSAs coming into the area. Using this method, we can reduce the amount of routes we have in our routing table and only have one default route on ABR (R2), which is a path to any route unknown in totally stub area. This way we can increase stability of the area since, totally stubby area will limit the flow of traffic coming into the area.

To configure totally stubby area we first enter ospfv3 process, in our next step we enter a specific address family (ipv4/ipv6), then we type area 7 stub no-summary command which will activate the specified area as totally stubby

R2: Area 7 **Configuring Gi0/0.10 as totally stubby in area 7** **IPv4 address family** **IPv6 address family**	 
CLI Commands:	R2: router ospfv3 7 address-family ipv4 unicast area 7 stub no-summary exit-address-family address-family ipv6 unicast area 7 stub no-summary exit-address-family
R3 (VRF): Area 7 **Configuring area 7 as totally stubby area** **IPv4 address family** **IPv6 address family**	 
CLI Commands:	R3: VRF-OSPF router ospfv3 7 address-family ipv4 unicast vrf VRF-OSPF area 7 stub exit-address-family address-family ipv6 unicast vrf VRF-OSPF area 7 stub exit-address-family

9. Note that on R3 in the VRF address family (IPv4 and IPv6) you must include the following command for your routes to show up in the routing table: capability vrf-lite

- Purpose of configuring capability vrf-lite is to make sure that packets are sent out of the area using correct LSAs instead of having type 3 LSAs which will only work inter area
- [capability vrf-lite] applies multi-VRF capability to the ospf process

R3 (VRF): Configure capability vrf-lite **IPv4 address family**	
CLI Command:	router ospfv3 7 address-family ipv4 unicast vrf VRF-OSPF

	capability vrf-lite
R3 (VRF): Configure capability vrf-lite **IPv6 address family**	
CLI Command: This command is configured under ospfv3 process in each ipv4 and ipv6 address-family	router ospfv3 7 address-family ipv6 unicast vrf VRF-OSPF Capability vrf-lite

Task 4: Configure EIGRP

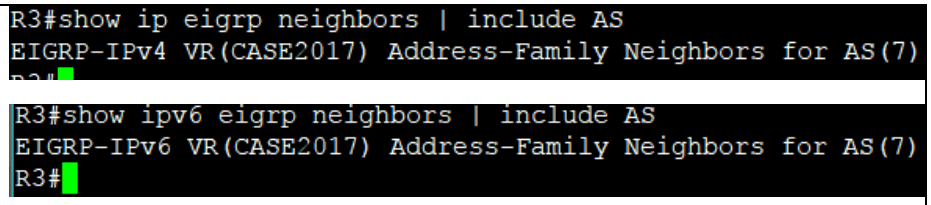
1. **Use an AS number equal to your group number.**

Configuring Autonomous number for EIGRP router protocol

The purpose of assigning Autonomous system (AS) number to EIGRP is to identify that this EIGRP is an internal EIGRP protocol running inside the Autonomous system defined

To configure Autonomous system enter EIGRP Named mode, then type address-family ipv4/ipv6 unicast autonomous-system (AS number) this will assign an Autonomous system (AS) number to EIGRP

R1: Configuring AS for IPv4 and IPv6	R1#show ip eigrp neighbors | include AS EIGRP-IPv4 VR(CASE2017) Address-Family Neighbors for AS(7) R1#show ipv6 eigrp neighbors | include AS EIGRP-IPv6 VR(CASE2017) Address-Family Neighbors for AS(7)
CLI Commands:	R1: router eigrp CASE2017 address-family ipv4 unicast autonomous-system 7 exit-address-family address-family ipv6 unicast autonomous-system 7 exit-address-family
R2 (VRF): Configuring AS for IPv4 and IPv6	R2#show ip eigrp vrf VRF-EIGRP neighbors | include AS EIGRP-IPv4 VR(CASE2017) Address-Family Neighbors for AS(7) R2# R2#show eigrp address-family ipv6 vrf VRF-EIGRP neighbors | include AS EIGRP-IPv6 VR(CASE2017) Address-Family Neighbors for AS(7) R2#
CLI Commands:	R2: VRF-EIGRP router eigrp CASE2017 address-family ipv4 unicast autonomous-system 7 vrf VRF-EIGRP exit-address-family address-family ipv6 unicast autonomous-system 7 vrf VRF-EIGRP exit-address-family

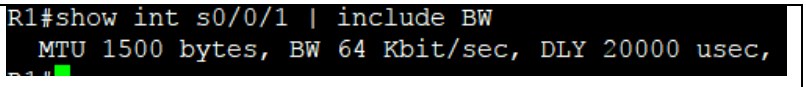
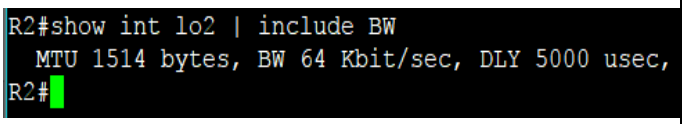
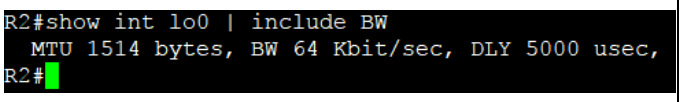
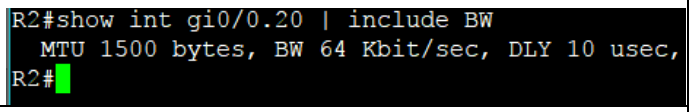
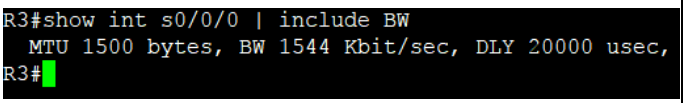
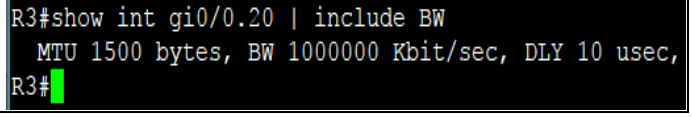
R3: Configuring AS for IPv4 and IPv6	
CLI Commands:	R3: router eigrp CASE2017 address-family ipv4 unicast autonomous-system 7 exit-address-family address-family ipv6 unicast autonomous-system 7 exit-address-family

2. Set the bandwidth of all interfaces appropriately.

Configure bandwidth as 64,000 bps on each interface that participate in EIGRP

Purpose of configuring bandwidth is to increase or decrease the flooding of traffic into EIGRP through an interface so by setting bandwidth to 64,000 bps it will reduce the amount of traffic flow into EIGRP

To configure bandwidth you must enter in an interface that belongs to EIGRP and then under the interface type bandwidth (bandwidth-number) command and it will set bandwidth to whatever number you entered.

R1: Configuring bandwidth to 64 on s0/0/1	
R2 (VRF): Configuring bandwidth to 64 on: **loopback1** **loopback2** **gi0/0.20**	  
R3: Configuring bandwidth to 64 on **s0/0/0** **gi0/0.20**	 

CLI Commandss:

R3:

```
interface GigabitEthernet0/0.20
    bandwidth 64
 exit
 interface Serial0/0/0
   bandwidth-percent 64
 exit
```

R1:

```
interface Serial0/0/1
 bandwidth 64
exit
```

R2: VRF-EIGRP

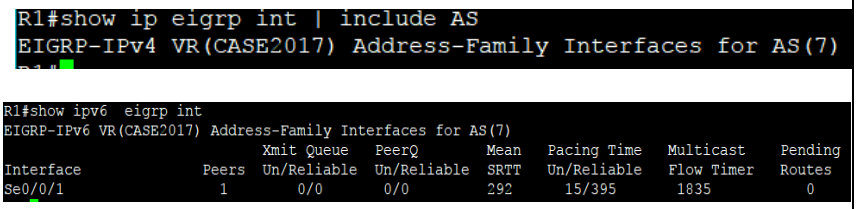
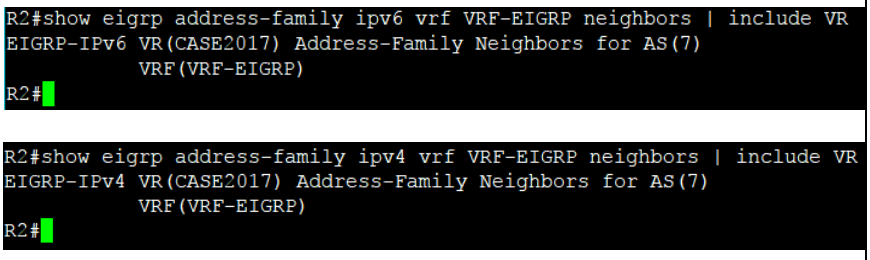
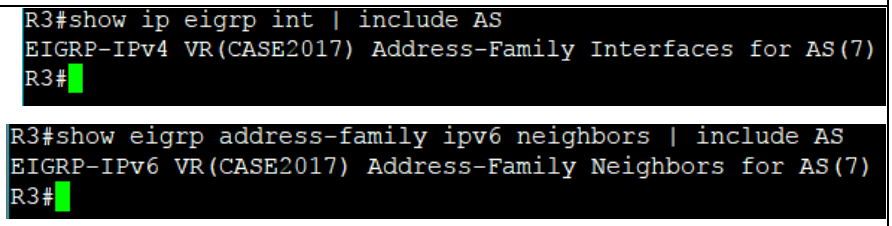
```
interface Loopback1
 bandwidth 64
exit
interface Loopback2
 bandwidth 64
exit

interface GigabitEthernet0/0.20
 bandwidth 64
exit
```

3. **Enable EIGRP Named Mode on R1, R2, and R3, for both IPv4 and IPv6, as indicated in the diagram. Name your EIGRP process CASE2017**

Configure EIGRP named mode on each router for both ipv4/ipv6 address family

We are using EIGRP named mode to reduced complexity of configuring both ipv4 and ipv6 since it takes two different router configuration mode to configure ipv4 and ipv6. Hence, to eliminate this problem we will use named mode so we can keep ipv4 and ipv6 separately under one EIGRP process instead of two.

To configure EIGRP named mode we type router eigrp [(name of you eigrp)] now under that we will create our ipv4/ipv6 address-families with specified AS which we in our first step

R1: Enabling EIGRP Named mode	
R2 (VRF): Enabling EIGRP Named mode	
R3: Enabling EIGRP Named mode	

CLI Commands:

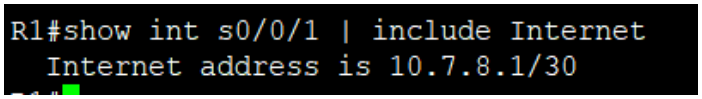
R1, R2: VRF-EIGRP , R3: router eigrp CASE2017

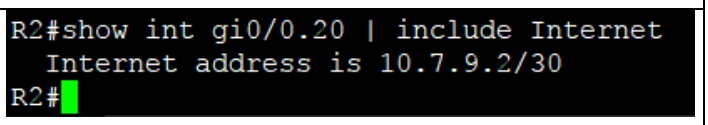
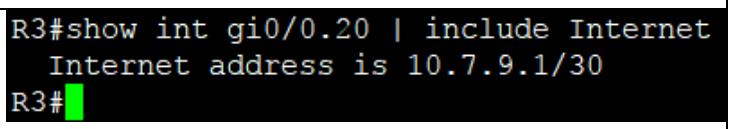
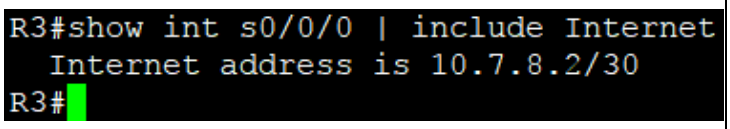
4. **Use /32 wlidcard masks for each interface in your network commands.**

Configuring 0.0.0.0 wildcard mask for each network command

(why)

To configure /32 wildcard mask we enter into EIGRP named mode and then we enter into address-family for ipv4 under that we use network command (network-to-advertise wildcard-mask)

R1: /32 wildcard mask for interface s0/0/1	
CLI Commands:	R1: router eigrp CASE2017 address-family ipv4 unicast autonomous-system 7 network 10.7.8.1 0.0.0.0

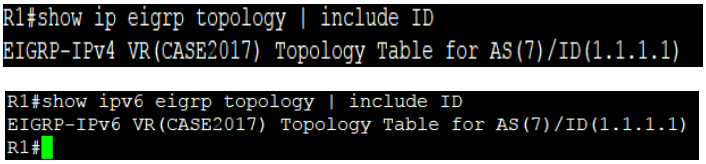
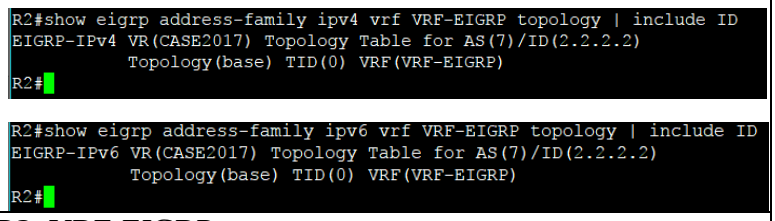
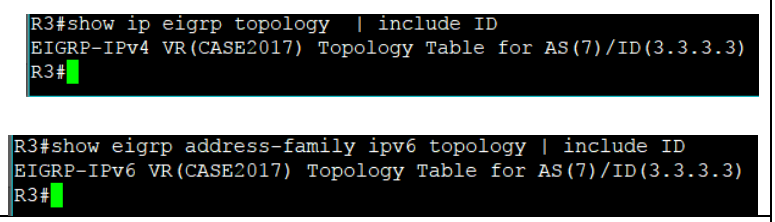
	exit-address-family
R2 (VRF): /32 wildcard mask for interface gi0/0.20	
CLI Commands:	R2: VRF-EIGRP router eigrp CASE2017 address-family ipv4 unicast autonomous-system 7 network 10.7.9.2 0.0.0.0 network 10.7.10.1 0.0.0.0 network 10.7.11.1 0.0.0.0 exit-address-family
R3: /32 wildcard mask for interfaces **gi0/0.20** **s0/0/0**	 
CLI Commands:	R3: router eigrp CASE2017 address-family ipv4 unicast autonomous-system 7 network 10.7.8.2 0.0.0.0 network 10.7.9.1 0.0.0.0 exit-address-family

5. **Use the router number as the router ID (e.g., on R1 use 1.1.1.1). Use this router ID for IPv4, IPv6, and the VRF address families as applicable**

Configuring router-id for each router

In this step we will configure router Id to make sure all routers are identified uniquely inside the network by all of the other router

To configure router ID we will enter EIGRP named mode first, secondly we will enter into each ipv4/ipv6 address-families and just type [eigrp router-id (32-bit number)] command which will assign a router-id to that router

R1: Router-id (1.1.1.1) **IPv4 address family** **IPv6 address family**	
CLI Commands:	R1: router eigrp CASE2017 address-family ipv4 unicast autonomous-system 7 eigrp router-id 1.1.1.1 exit-address-family address-family ipv6 unicast autonomous-system 7 eigrp router-id 1.1.1.1 exit-address-family
R2 (VRF): Router-id (2.2.2.2) **IPv4 address family** **IPv6 address family**	
CLI Commands:	R2: VRF-EIGRP router eigrp CASE2017 address-family ipv4 unicast vrf VRF-EIGRP autonomous-system 7 eigrp router-id 2.2.2.2 exit-address-family address-family ipv6 unicast vrf VRF-EIGRP autonomous-system 7 eigrp router-id 2.2.2.2 exit-address-family
R3: Router-id (3.3.3.3) **IPv4 address family** **IPv6 address family**	
CLI Commands:	R3: router EIGRP CASE2017

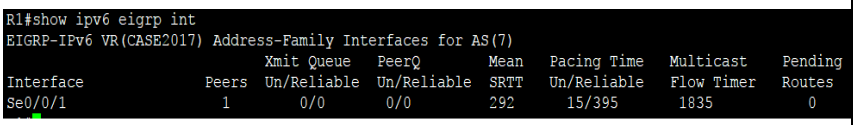
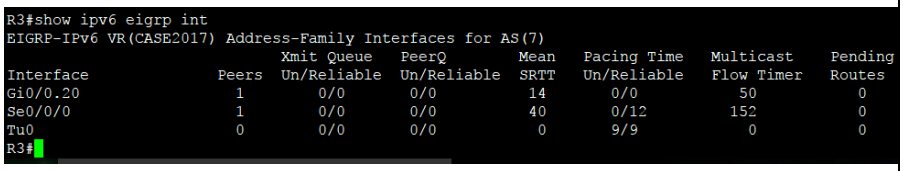
	address-family ipv4 unicast autonomous-system 7 EIGRP router-id 3.3.3.3 exit-address-family address-family ipv6 unicast autonomous-system 7 EIGRP router-id 3.3.3.3 exit-address-family

6. **By default, all IPv6 interfaces participate in EIGRP Named Mode. Remove EIGRP from interfaces where it is not required (check show ipv6 EIGRP interface).**

Shutdown all unwanted ipv6 interfaces that are participating in EIGRP named mode

Purpose of this is to make sure only necessary interfaces are participating in ipv6 EIGRP named mode

To shutdown all unwanted interfaces for ipv6 we first enter into EIGRP named mode and then we enter specifically into ipv6 address-family, now we will type [af-interface (interface port number)], under that type [(shutdown)] to eliminate that interface from ipv6 EIGRP named mode

R1: Removing EIGRP NAMED Mode from ipv6 where it is not required **s0/0/0**	
CLI Commands:	R1: router EIGRP CASE2017 address-family ipv6 unicast autonomous-system 7 af-interface Serial0/0/0 shutdown exit-af-interface
R3: Removing EIGRP NAMED Mode from ipv6 where it is not required **s0/1/0** **s0/1/1**	

CLI Commands:	R3: router EIGRP CASE2017 address-family ipv6 unicast autonomous-system 7 af-interface Serial0/1/0 shutdown exit-af-interface af-interface Serial0/1/1 shutdown exit-af-interface

7. **Configure all Loopback interfaces as passive.**

We are configuring all loopbacks to passive-interface

Purpose of passive-interface is to stop sending hello packets to the neighbors which its not directly connected, also block any incoming routing updates, which reduces the flow of traffic in EIGRP

To configure passive-interface we can enter router EIGRP named mode or we can configure it directly on interface by just typing [passive-interface] once you enter global-if mode

R2 (VRF): Configuring all loopback to passive **Loopback1** **Loopback2**	R2#show eigrp address-family ipv4 vrf VRF-EIGRP int lo1 EIGRP-IPv4 VR(CASE2017) Address-Family Interfaces for AS(7) VRF(VRF-EIGRP) Xmit Queue PeerQ Mean Pacing Time Multicast Pending Interface Peers Un/Reliable Un/Reliable SRTT Un/Reliable Flow Timer Routes R2# R2#show eigrp address-family ipv4 vrf VRF-EIGRP int lo2 EIGRP-IPv4 VR(CASE2017) Address-Family Interfaces for AS(7) VRF(VRF-EIGRP) Xmit Queue PeerQ Mean Pacing Time Multicast Pending Interface Peers Un/Reliable Un/Reliable SRTT Un/Reliable Flow Timer Routes R2#

CLI Commands:

R2: VRF-EIGRP router EIGRP CASE2017 address-family ipv4 unicast vrf VRF-EIGRP autonomous-system 7 af-interface Loopback1 passive-interface exit-af-interface af-interface Loopback2 passive-interface exit-af-interface

```
address-family ipv4 unicast vrf VRF-EIGRP autonomous-system 7
  af-interface Loopback1
    passive-interface
  exit-af-interface
 af-interface Loopback2
    passive-interface
 exit-af-interface
```

8. **Configure R2 VRF-EIGRP as a stub router in both IPv4 and IPv6, advertising only connected routes.**

In this step we have to configure R2-vrf as a stub router which only advertises connected routes

Stub router helps save bandwidth and increase the speed of convergence. Stub router do not forward any routing updates to its neighbors unless otherwise such as in this case we are told to advertise only connected routes. Therefore, it will advertise only connected subnets to its neighbors. They also do not accept any updates from it neighbors which is a difference between OSPF and EIGRP stubbiness.

To configure a stub connected route you must enter EIGRP named mode then enter address-family for ipv4/ipv6 [address-family ipv4/ipv6 unicast vrf (vrf-name) (AS)] under that type [EIGRP stub connected] and after that router will be a stub router, but it will only advertising connected routes to its neighbors

R2 (VRF): Proof of stub router, but advertise only connected routes **IPv4 address family** **IPv6 address family**	R3#show ip eigrp neighbors detail EIGRP-IPv4 VR(CASE2017) Address-Family Neighbors for AS(7) H Address Interface Hold Uptime SRTT RTO Q Seq (sec) (ms) Cnt Num 1 10.7.9.2 Gi0/0.20 10 00:16:06 1597 5000 0 2 Version 16.0/2.0, Retrans: 1, Retries: 0, Prefixes: 2 Topology-ids from peer - 0 Stub Peer Advertising (CONNECTED) Routes Suppressing queries R3#show ipv6 eigrp neighbors detail EIGRP-IPv6 VR(CASE2017) Address-Family Neighbors for AS(7) H Address Interface Hold Uptime SRTT RTO Q Seq (sec) (ms) Cnt Num 1 Link-local address: Gi0/0.20 14 00:18:38 1598 5000 0 2 FE80::2 Version 16.0/2.0, Retrans: 1, Retries: 0, Prefixes: 3 Topology-ids from peer - 0 Stub Peer Advertising (CONNECTED) Routes Suppressing queries

CLI Commands:

```
R2: VRF-EIGRP
router EIGRP CASE2017
 address-family ipv4 unicast vrf VRF-EIGRP autonomous-system 7
   EIGRP stub connected
exit-address-family
address-family ipv6 unicast vrf VRF-EIGRP autonomous-system 7
   EIGRP stub connected
exit-address-family
```

Task 5: Configure Redistribution and Summarization

1. **Perform mutual redistribution between EIGRP and OSPF on R1 for both IPv4 and IPv6. For EIGRP metrics use the following values:** Bandwidth: 1 Gbps. Delay: 200 μsec, Reliability: 255/255, Load: 1/255, MTU: 1500

- Firstly, to redistribute OSPF and EIGRP, we must first summarize OSPF and EIGRP individually. In OSPF, there are two areas, area 0 and area 7. The ABR between area 0 and area 7 needs a summarized route, advertising to area 0. In the topology there are 3 routers that participate in OSPF, R1, R2, and R3-VRF. R2 is a ABR while, R1 is part of area 0 and R3-VRF is part of area 7. We first summarized all the routes in area 7 and come up with a summary address that will be advertised in area 0. The summary route will then be redistributed into EIGRP from R1 along side area 0 OSPF routes. Secondly, in the EIGRP section there are 3 routers that participate in it, R1, R2-VRF, and R3. However, they do not need summarization like OSPF, they simply all belong to 1 area, which makes redistribution easier. Redistribution for EIGRP into OSPF can be done by having redistribution commands in the OSPF address family of R1. Below in the snippets of "Show IP route" in R1, R2 and R3 which show either OSPF or EIGRP External routes depending on where the device is located. If all routes are present in the routing table, then redistribution and summarization was successful. To simplify, the redistribute commands in EIGRP address family below redistribute OSPF routes into EIGRP and the commands in the OSPF address family redistribute EIGRP routes into OSPF.

```
router eigrp CASE2017
 address-family ipv4 unicast autonomous-system 7
  topology base
   redistribute ospfv3 7 metric 10000 200 255 1 1500
  exit-af-topology
 exit-address-family

 address-family ipv6 unicast autonomous-system 7
  af-interface Serial0/0/1
  exit-af-interface

  topology base
   redistribute ospf 7 metric 10000 200 255 1 1500 include-connected
  exit-af-topology
 exit-address-family

router ospfv3 7
 address-family ipv4 unicast
  redistribute eigrp 7
exit-address-family

 address-family ipv6 unicast
  redistribute eigrp 7 include-connected
 exit-address-family
```

Here is an example of router 1's routing table showing OSPF and Inter Area OSPF routes.

Router 1 IPv4	R1#show ip route | include O 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 o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP O IA 10.7.0.0/22 [110/131071] via 10.7.5.1, 00:15:19, Serial0/0/0 O 10.7.4.0/24 [110/65536] via 10.7.5.1, 00:16:35, Serial0/0/0 Here is an example of R1 showing EIGRP External routes in its routing table. R1#show ip route | include D D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP D*EX 0.0.0.0/0 [170/91264000] via 10.7.8.2, 00:15:29, Serial0/0/1 D 10.7.9.0/30 [90/90245120] via 10.7.8.2, 00:16:19, Serial0/0/1 D 10.7.10.0/24 [90/92805120] via 10.7.8.2, 00:16:01, Serial0/0/1 D 10.7.11.0/24 [90/92805120] via 10.7.8.2, 00:16:01, Serial0/0/1

	Here is an example of R2 showing OSPF External routes in its routing table.
Router 2 IPv4	R2#show ip route | include O E2 O E2 10.7.8.0/30 [110/20] via 10.7.5.2, 00:13:28, Serial0/0/0 O E2 10.7.9.0/30 [110/20] via 10.7.5.2, 00:13:09, Serial0/0/0 O E2 10.7.10.0/24 [110/20] via 10.7.5.2, 00:12:51, Serial0/0/0 O E2 10.7.11.0/24 [110/20] via 10.7.5.2, 00:12:51, Serial0/0/0

	Here is an example of router 2's routing table showing external EIGRP routes.
Router 2 -VRF IPv4	R2#show ip route vrf VRF-EIGRP | include D*EX D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area D*EX 0.0.0.0/0 [170/81029120] via 10.7.9.1, 00:07:44, GigabitEthernet0/0.20 D EX 10.7.0.0/22 D EX 10.7.4.0/24 D EX 10.7.5.0/30

	Here is an example of router 3's routing table showing external OSPF routes.
Router 3 IPv4	R3#show ip route | include D EX D EX 10.7.0.0/22 [170/14580062] via 10.7.8.1, 00:17:43, Serial0/0/0 D EX 10.7.4.0/24 [170/14580062] via 10.7.8.1, 00:17:43, Serial0/0/0 D EX 10.7.5.0/30 [170/14580062] via 10.7.8.1, 00:17:43, Serial0/0/0

2. **Create a static default route on R3 pointing to the IPv4 address of ISP2 (R4). Do the same for IPv6.**

Since, 203. network(ISP2) is not included in the routing table of R1, R2, or R4, they need a static default route to get to ISP2. The only router that knows about ISP2 is R3, since it is directly connected to it. Therefore, a static default route on R3 pointing to R4-VRF is created so when a router wants to reach ISP2, it is able to gain reachability to it.

CLI Command • ip route 0.0.0.0 0.0.0.0 203.0.7.2
R3: IPv4 static route pointing to ISP2 (R4)

```
R3#show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is 203.0.7.2 to network 0.0.0.0

S*    0.0.0.0/0 [1/0] via 203.0.7.2
```

R3 IPv6 static route pointing to ISP2 (R4)

```
R3#show ipv6 route
IPv6 Routing Table - default - 11 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, HA - Home Agent, MR - Mobile Router, R - RIP
       H - NHRP, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       IS - ISIS summary, D - EIGRP, EX - EIGRP external, NM - NEMO
       ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2, la - LISP alt
       lr - LISP site-registrations, ld - LISP dyn-eid, a - Application
S   ::/0 [1/0]
     via 2001:DB8:7:ABCD::2
```

3. **Create a static default route on R4 to 203.1.1.254 (a gateway on the Internet).**

   Creating a default route to get to the gateway on the internet on R4. This is done so an outside router has reachability to the internet.

CLI Command

- ip route 0.0.0.0 0.0.0.0 203.1.1.254

R4: default static route to gateway of internet (203.1.1.254)

```
R4#show ip route vrf VRF-INET

Routing Table: VRF-INET
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is 203.1.1.254 to network 0.0.0.0

S*    0.0.0.0/0 [1/0] via 203.1.1.254
      203.0.7.0/24 is variably subnetted, 2 subnets, 2 masks
C        203.0.7.0/29 is directly connected, Serial0/1/1
L        203.0.7.2/32 is directly connected, Serial0/1/1
      203.1.1.0/24 is variably subnetted, 2 subnets, 2 masks
C        203.1.1.0/24 is directly connected, GigabitEthernet0/0/1.20
L        203.1.1.7/32 is directly connected, GigabitEthernet0/0/1.20
```

4. **Distribute the default route for IPv4 and IPv6 via redistribution into EIGRP, using the metrics given previously for R1.**

Redistributing all default routes for IPv4 and IPv6 into EIGRP so when the initial redistribution between OSPF and EIGRP process is complete, the default routes are shown in OSPF router's routing table. This is essential if an OSPF router wants reachability to the destination of the default route.

R1: Redistributing IPv4 default route into EIGRP via 10.7.8.2 network

```
R1#show ip route eigrp
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static rout
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is 10.7.8.2 to network 0.0.0.0

D*EX  0.0.0.0/0 [170/91264000] via 10.7.8.2, 00:27:19, Serial0/0/1
      10.0.0.0/8 is variably subnetted, 9 subnets, 4 masks
D        10.7.9.0/30 [90/90245120] via 10.7.8.2, 00:23:41, Serial0/0/1
D        10.7.10.0/24 [90/90245760] via 10.7.8.2, 00:23:06, Serial0/0/1
D        10.7.11.0/24 [90/90245760] via 10.7.8.2, 00:23:06, Serial0/0/1
```

R1: Redistributing IPv6 default route into EIGRP via R3's link-local address

```
R1#show ipv6 route eigrp
IPv6 Routing Table - default - 14 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, HA - Home Agent, MR - Mobile Router, R - RIP
       H - NHRP, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       IS - ISIS summary, D - EIGRP, EX - EIGRP external, NM - NEMO
       ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2, la - LISP alt
       lr - LISP site-registrations, ld - LISP dyn-eid, a - Application
EX  ::/0 [170/91264000]
     via FE80::3, Serial0/0/1
```

Redistributing IPv4 default route into EIGRP via 10.7.9.1 network

```
R2#show ip route vrf VRF-EIGRP

Routing Table: VRF-EIGRP
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is 10.7.9.1 to network 0.0.0.0

D*EX  0.0.0.0/0 [170/1541120] via 10.7.9.1, 00:28:49, GigabitEthernet0/0.20
```

Redistributing IPv6 default route into EIGRP via FE80::3 link-local address

```
R2#show ipv6 route vrf VRF-EIGRP
IPv6 Routing Table - VRF-EIGRP - 10 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, HA - Home Agent, MR - Mobile Router, R - RIP
       H - NHRP, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       IS - ISIS summary, D - EIGRP, EX - EIGRP external, NM - NEMO
       ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2, la - LISP alt
       lr - LISP site-registrations, ld - LISP dyn-eid, a - Application
EX  ::/0 [170/1541120]
     via FE80::3, GigabitEthernet0/0.20
```

5. **On R1, originate a default route into OSPFv3, only as long as there is a default route already in R1's routing table.**

Since OSPF does not redistribute default routes we must apply a specific command that injects default routes into OSPF. The command is "default-information originate", this command will inject the default route in the routing domain of OSPF. We can only use this command if there is a default route in R1's routing table and as per the snippet below, a default route is present, therefore this originate command is viable and needed.

	CLI command router ospfv3 7 address-family ipv4 unicast default-information originate exit-address-family address-family ipv6 unicast default-information originate exit-address-family
Router 1	R1#show ip route | include 0.0.0.0 Gateway of last resort is 10.7.8.2 to network 0.0.0.0 D*EX 0.0.0.0/0 [170/91264000] via 10.7.8.2, 00:18:40, Serial0/0/1 10.0.0.0/8 is variably subnetted, 9 subnets, 4 masks

6. **Create a static route on R4 to the 2001:db8:x::/48 subnet. Be sure this route is created in the VRF-INET VRF.**

This static route is created so that ISP2 has reachability to R3 via IPv6. R4-VRF has no knowledge of the IPv6 routes that R3 has and therefore needs a static route to R3 so R4-VRF can get reachability outside of R3.

CLI Command • ipv6 route vrf VRF-INET 2001:DB8:7::/48 2001:DB8:7:ABCD::1
Static route is showed in R4's VRF routing table.

```
R4#show ipv6 route vrf VRF-INET
IPv6 Routing Table - VRF-INET - 6 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, R - RIP, I1 - ISIS L1, I2 - ISIS L2
       IA - ISIS interarea, IS - ISIS summary, D - EIGRP, EX - EIGRP external
       ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2, a - Application
C   2001:DB8::/64 [0/0]
     via GigabitEthernet0/0/1.20, directly connected
L   2001:DB8::7/128 [0/0]
     via GigabitEthernet0/0/1.20, receive
S   2001:DB8:7::/48 [1/0]
     via 2001:DB8:7:ABCD::1
C   2001:DB8:7:ABCD::/64 [0/0]
     via Serial0/1/1, directly connected
L   2001:DB8:7:ABCD::2/128 [0/0]
     via Serial0/1/1, receive
L   FF00::/8 [0/0]
     via Null0, receive
```

7. **Summarize the IPv4 routes in OSPF Area x to the most efficient summary address and advertise it into Area 0.**

In Area 7 there are 3 OSPF subnets. 10.7.0.1/24, 10.7.1.1/24, and 10.7.2.0/30. Below, a summary route calculation is done to obtain a summary address, summarizing the entire OSPFv3 area 7. This is done so when EIGRP devices wants to gain reachability to OSPF area 7, area 0 will have a route to area 7. Instead of advertising single area 7 addresses to area 0, this is much more efficient.

10.7.0.1 - **0.0.0.0.1.0.1.0**.0.0.0.0.0.1.1.1.**0.0.0.0.0.0.0.0**.0.0.0.0.0.0.0.1
10.7.1.1 - **0.0.0.0.1.0.1.0**.0.0.0.0.0.1.1.1.**0.0.0.0.0.0.0.1**.0.0.0.0.0.0.0.1
10.7.2.1 - **0.0.0.0.1.0.1.0**.0.0.0.0.0.1.1.1.**0.0.0.0.0.0.1.0**.0.0.0.0.0.0.0.1
10.7.2.1 - **0.0.0.0.1.0.1.0**.0.0.0.0.0.1.1.1.**0.0.0.0.0.0.1.0**.0.0.0.0.0.0.0.1

Summarized Address: 10.7.0.0/22

The Show IP Route of R2 shows the 10.7.0.0/22 summary, as it was advertised from R3 VRF.

```
R2#show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is 10.7.5.2 to network 0.0.0.0

O*E2  0.0.0.0/0 [110/1] via 10.7.5.2, 02:04:41, Serial0/0/0
      10.0.0.0/8 is variably subnetted, 13 subnets, 4 masks
O        10.7.0.0/22 is a summary, 02:08:02, Null0
O        10.7.0.0/24 [110/65536] via 10.7.2.1, 02:08:02, GigabitEthernet0/0.10
O        10.7.1.0/24 [110/65536] via 10.7.2.1, 02:08:02, GigabitEthernet0/0.10
C        10.7.2.0/30 is directly connected, GigabitEthernet0/0.10
L        10.7.2.2/32 is directly connected, GigabitEthernet0/0.10
C        10.7.4.0/24 is directly connected, Loopback0
L        10.7.4.1/32 is directly connected, Loopback0
C        10.7.5.0/30 is directly connected, Serial0/0/0
L        10.7.5.1/32 is directly connected, Serial0/0/0
O E2     10.7.8.0/30 [110/20] via 10.7.5.2, 02:04:43, Serial0/0/0
O E2     10.7.9.0/30 [110/20] via 10.7.5.2, 02:04:41, Serial0/0/0
O E2     10.7.10.0/24 [110/20] via 10.7.5.2, 02:04:41, Serial0/0/0
O E2     10.7.11.0/24 [110/20] via 10.7.5.2, 02:04:41, Serial0/0/0
```

8. **Create a single EIGRP summary route on the R1 interface to R3, summarizing all of the IPv6 routes in the OSPF network as efficiently as possible.**

For IPv6 reachability across OSPF and EIGRP, a summary route similar to the one created for IPv4 is needed for summarizing OSPF areas. We need to advertise IPv6 routes from OSPF by making a summary address that encompasses all subnets used in to EIGRP so the devices participating in EIGRP will have a IPV6 route to the OSPF areas. The command in R1 address family of EIGRP summarizes the IPv6 routes from OSPF to R3.

CLI Command

R1:

```
router eigrp CASE2017
 address-family ipv6 unicast autonomous-system 7
  af-interface Serial0/0/1
   summary-address 2001:DB8:7::/61
  exit-af-interface
 exit-address-family
```

R1: Show IPv6 route shows a /61 summary route that was created in R1

```
R1#
R1#show ipv6 route eigrp
IPv6 Routing Table - default - 14 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, HA - Home Agent, MR - Mobile Router, R - RIP
       H - NHRP, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       IS - ISIS summary, D - EIGRP, EX - EIGRP external, NM - NEMO
       ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2, la - LISP alt
       lr - LISP site-registrations, ld - LISP dyn-eid, a - Application
EX  ::/0 [170/91264000]
     via FE80::3, Serial0/0/1
D   2001:DB8:7::/61 [5/1536000]
     via Null0, directly connected
D   2001:DB8:7:9::/64 [90/90245120]
     via FE80::3, Serial0/0/1
D   2001:DB8:7:10::/64 [90/90245760]
     via FE80::3, Serial0/0/1
D   2001:DB8:7:11::/64 [90/90245760]
     via FE80::3, Serial0/0/1
```

R3: Show IPv6 route shows a /61 summary route that was created in R3

```
R3#show ipv6 route eigrp
IPv6 Routing Table - default - 11 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, HA - Home Agent, MR - Mobile Router, R - RIP
       H - NHRP, I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       IS - ISIS summary, D - EIGRP, EX - EIGRP external, NM - NEMO
       ND - ND Default, NDp - ND Prefix, DCE - Destination, NDr - Redirect
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2, la - LISP alt
       lr - LISP site-registrations, ld - LISP dyn-eid, a - Application
D   2001:DB8:7::/61 [90/14580062]
     via FE80::1, Serial0/0/0
D   2001:DB8:7:10::/64 [90/10880]
     via FE80::2, GigabitEthernet0/0.20
D   2001:DB8:7:11::/64 [90/10880]
     via FE80::2, GigabitEthernet0/0.20
```

Task 6: Configure MP-BGP

1. **The BGP AS number is 650xx, where xx is your 2-digit group number (e.g. 01, 02, 03…10, 11, etc.).**

BGP AS number is configured on R3 and R4 to initialize the process of activating BGP on both routers. BGP process needs to know if the autonomous system number to determine if the router is participating in either iBGP or eBGP process.

CLI Command for R3 and R4 to initialize BGP AS number

router bgp 65007

Router 3

```
R3# show ip bgp summary
BGP router identifier 7.3.3.3, local AS number 65007
```

Router 4

```
R4#show ip bgp summary
BGP router identifier 7.4.4.4, local AS number 65007
```

2. **Use router ID x.y.y.y, where x is your group number and y is the router number (e.g. Group 5 would use 5.3.3.3 on R3)**

Router ID in BGP is used to identify the router so when a packet from router is sent to another device, you are able to identify the source of the packet.

CLI Command for R3 and R4 to initialize BGP router-id number

R3:	R4:
router bgp 65007	router bgp 65007
bgp router-id 7.3.3.3	bgp router-id 7.4.4.4

Router 3

```
R3# show ip bgp summary
BGP router identifier 7.3.3.3, local AS number 65007
```

Router 4

```
R4#show ip bgp summary
BGP router identifier 7.4.4.4, local AS number 65007
```

3. Configure iBGP neighbor relationships between R3 and R4 as shown in the topology diagram.

Neighbor relationship is configured on both R3 and R4 to enable an internal BGP session between the two routers. On R3 and R4 an identical autonomous system number is used to issue a iBGP session. Then using a remote-as command using the neighbors IP address is issued to identify the neighbor. Finally, a neighbor activate command is used to activate the session between the two router so both routers can exchange routing information.

CLI Command for R3 and R4 to initialize iBGP neighbor relationships

R3:

```
router bgp 65007
bgp router-id 7.3.3.3
neighbor 172.16.7.2 remote-as 65007

address-family ipv4
 neighbor 172.16.7.2 activate
 neighbor 172.16.7.2 next-hop-self
exit-address-family

address-family ipv6
 neighbor 172.16.7.2 activate
 exit-address-family
```

R4:

```
router bgp 65007
bgp router-id 7.4.4.4
neighbor 172.16.7.1 remote-as 65007

address-family ipv4
 neighbor 172.16.7.1 activate
 neighbor 172.16.7.1 next-hop-self
exit-address-family

address-family ipv6
 neighbor 172.16.7.1 activate
 exit-address-family
```

Router 3 is neighbors with R4's 172.16.7.2 network

```
R3# show ip bgp summary
BGP router identifier 7.3.3.3, local AS number 65007
BGP table version is 11, main routing table version 11
8 network entries using 1152 bytes of memory
9 path entries using 720 bytes of memory
5/3 BGP path/bestpath attribute entries using 800 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 2672 total bytes of memory
BGP activity 9/1 prefixes, 11/2 paths, scan interval 60 secs

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
172.16.7.2      4        65007      55      57       11    0    0 00:44:32        1
```

Router 4 is neighbors with R3's 172.16.7.1 network

```
R4#show ip bgp su
R4#show ip bgp summary
BGP router identifier 7.4.4.4, local AS number 65007
BGP table version is 10, main routing table version 10
8 network entries using 1984 bytes of memory
9 path entries using 1080 bytes of memory
4/4 BGP path/bestpath attribute entries using 992 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 4056 total bytes of memory
BGP activity 8/0 prefixes, 9/0 paths, scan interval 60 secs

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
172.16.7.1      4        65007      55      53       10    0    0 00:43:20        8
```

4. **Configure R3 and R4 to advertise themselves as the next hop for all IPv4 routes they exchange with each other.**

The "neighbor [neighbor IP address] next-hop-self" command is used to determine the next hop for the routers in the iBGP session. This command forces the routers in the iBGP session to choose the neighbor as the next-hop. Therefore, R3's next-hop is 172.16.7.2 and R4's next-hop-self is 172.16.7.1.

CLI Command for R3 and R4 to advertise themselves as the next hop for all IPv4 routes they exchange with each other.

R3:

```
router bgp 65007
bgp router-id 7.3.3.3
neighbor 172.16.7.2 remote-as 65007

address-family ipv4
 neighbor 172.16.7.2 next-hop-self
exit-address-family
```

R4:

```
router bgp 65007
bgp router-id 7.4.4.4
neighbor 172.16.7.1 remote-as 65007

address-family ipv4
 neighbor 172.16.7.1 next-hop-self
exit-address-family
```

Router 3 is neighbors with R4's 172.16.7.2 network

```
R3# show ip bgp summary
BGP router identifier 7.3.3.3, local AS number 65007
BGP table version is 11, main routing table version 11
8 network entries using 1152 bytes of memory
9 path entries using 720 bytes of memory
5/3 BGP path/bestpath attribute entries using 800 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 2672 total bytes of memory
BGP activity 9/1 prefixes, 11/2 paths, scan interval 60 secs

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
172.16.7.2      4        65007      55      57       11    0    0 00:44:32        1
```

Router 4 is neighbors with R3's 172.16.7.1 network

```
R4#show ip bgp su
R4#show ip bgp summary
BGP router identifier 7.4.4.4, local AS number 65007
BGP table version is 10, main routing table version 10
8 network entries using 1984 bytes of memory
9 path entries using 1080 bytes of memory
4/4 BGP path/bestpath attribute entries using 992 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 4056 total bytes of memory
BGP activity 8/0 prefixes, 9/0 paths, scan interval 60 secs

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
172.16.7.1      4        65007      55      53       10    0    0 00:43:20        8
```

5. **The configuration should use MP-BGP to carry both IPv4 and IPv6 routes (IPv6 will be configured in Task 8).**

The MP-BGP allows the BGP process to create address families in which routers can carry IPv4 and IPv6 routes.

CLI Command for R3 and R4 showing MP-BGP

R3:

```
router bgp 65007

address-family ipv4
exit-address-family

address-family ipv6
exit-address-family
```

R4:

```
router bgp 65007

address-family ipv4
exit-address-family

address-family ipv6
exit-address-family
```

Router 3 – example when IPv4 routes are added in address families.

```
R3#show bgp ipv4 unicast
BGP table version is 36, local router ID is 7.3.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  10.7.0.0/22     10.7.8.1          14580062         32768 i
 *>  10.7.4.0/24     10.7.8.1          14580062         32768 i
 *>  10.7.5.0/30     10.7.8.1          14580062         32768 i
 *>  10.7.8.0/30     0.0.0.0                 0         32768 i
 *>  10.7.9.0/30     0.0.0.0                 0         32768 i
 *>  10.7.10.0/24    10.7.9.2          82565120         32768 i
 *>  10.7.11.0/24    10.7.9.2          82565120         32768 i
```

Router 3 – example when IPv4 routes are added in address families.

```
R4#show bgp ipv4 unicast | include 172
 *>i 10.7.0.0/22      172.16.7.1        14580062    100      0 i
 *>i 10.7.4.0/24      172.16.7.1        14580062    100      0 i
 *>i 10.7.5.0/30      172.16.7.1        14580062    100      0 i
 *>i 10.7.8.0/30      172.16.7.1               0    100      0 i
 *>i 10.7.9.0/30      172.16.7.1               0    100      0 i
 *>i 10.7.10.0/24     172.16.7.1        82565120    100      0 i
 *>i 10.7.11.0/24     172.16.7.1        82565120    100      0 i
 * i 172.16.7.0/30    172.16.7.1               0    100      0 i
```

6. **Advertise all subnets of the 10.x.0.0/16 networks, except any /32 routes, from R3 to R4. Do not add any static or summary routes to accomplish this.**

This step advertises the entire network comprising of OSPF and EIGRP networks into R4. This is done so R4 has routes to the OSPF and EIGRP networks. To accomplish the advertising of routes into R4, network commands are issued that were learned from R3's routing table and then the same routes from the routing table are issued in the IPv4 address family of BGP in R3. R4 will now have all of R3's routes.

CLI Command to advertise all of the 10.7.0.0/16 network from R3's routing table to R4

R3:

```
router bgp 65007
 bgp router-id 7.3.3.3
 neighbor 172.16.7.2 remote-as 65007

 address-family ipv4
  network 10.7.0.0 mask 255.255.252.0
  network 10.7.4.0 mask 255.255.255.0
  network 10.7.5.0 mask 255.255.255.252
  network 10.7.8.0 mask 255.255.255.252
  network 10.7.9.0 mask 255.255.255.252
  network 10.7.10.0 mask 255.255.255.0
  network 10.7.11.0 mask 255.255.255.0
  neighbor 172.16.7.2 activate
  neighbor 172.16.7.2 next-hop-self
exit-address-family
```

Router 3

```
R3#show ip bgp
BGP table version is 11, local router ID is 7.3.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  10.7.0.0/22      10.7.8.1          14580062         32768 i
 *>  10.7.4.1/32      10.7.8.1          14580062         32768 i
 *>  10.7.5.0/30      10.7.8.1          14580062         32768 i
 *>  10.7.8.0/30      0.0.0.0                 0         32768 i
 *>  10.7.9.0/30      0.0.0.0                 0         32768 i
 *>  10.7.10.0/24     10.7.9.2            10880         32768 i
 *>  10.7.11.0/24     10.7.9.2            10880         32768 i
 * i 172.16.7.0/30    172.16.7.2              0    500      0 i
 *>                   0.0.0.0                 0         32768 i
```

Router 4

```
R4#show ip bgp
BGP table version is 10, local router ID is 7.4.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>i 10.7.0.0/22      172.16.7.1        14580062    100      0 i
 *>i 10.7.4.1/32      172.16.7.1        14580062    100      0 i
 *>i 10.7.5.0/30      172.16.7.1        14580062    100      0 i
 *>i 10.7.8.0/30      172.16.7.1              0     100      0 i
 *>i 10.7.9.0/30      172.16.7.1              0     100      0 i
 *>i 10.7.10.0/24     172.16.7.1          10880     100      0 i
 *>i 10.7.11.0/24     172.16.7.1          10880     100      0 i
 * i 172.16.7.0/30    172.16.7.1              0     100      0 i
 *>                   0.0.0.0                 0          32768 i
```

7. Also advertise the 172.16.x.0/30 subnet.

As said in the previous step, all routes available in the routing table of R3 are advertised by IPv4 Address family of BGP. 172.16.7.0/30 was present in R3's routing table therefore, it must also be advertised to R4. Also this network is also the network used to implement an iBGP session with R4, therefore, this network command is essential to have in the address family of BGP.

CLI Command to advertise the 172.16.7.0/30 network from R3's routing table to R4

```
router bgp 65007
 bgp router-id 7.3.3.3
 neighbor 172.16.7.2 remote-as 65007
 address-family ipv4
  network 172.16.7.0 mask 255.255.255.252
  neighbor 172.16.7.2 activate
  neighbor 172.16.7.2 next-hop-self
exit-address-family
```

Router 3

```
R3#show ip bgp
BGP table version is 11, local router ID is 7.3.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  10.7.0.0/22      10.7.8.1          14580062         32768 i
 *>  10.7.4.1/32      10.7.8.1          14580062         32768 i
 *>  10.7.5.0/30      10.7.8.1          14580062         32768 i
 *>  10.7.8.0/30      0.0.0.0                 0         32768 i
 *>  10.7.9.0/30      0.0.0.0                 0         32768 i
 *>  10.7.10.0/24     10.7.9.2            10880         32768 i
 *>  10.7.11.0/24     10.7.9.2            10880         32768 i
 * i 172.16.7.0/30    172.16.7.2              0    500      0 i
 *>                   0.0.0.0                 0         32768 i
```

Router 4

```
R4#show ip bgp
BGP table version is 10, local router ID is 7.4.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>i 10.7.0.0/22      172.16.7.1        14580062    100      0 i
 *>i 10.7.4.1/32      172.16.7.1        14580062    100      0 i
 *>i 10.7.5.0/30      172.16.7.1        14580062    100      0 i
 *>i 10.7.8.0/30      172.16.7.1              0     100      0 i
 *>i 10.7.9.0/30      172.16.7.1              0     100      0 i
 *>i 10.7.10.0/24     172.16.7.1          10880     100      0 i
 *>i 10.7.11.0/24     172.16.7.1          10880     100      0 i
 * i 172.16.7.0/30    172.16.7.1              0     100      0 i
 *>                   0.0.0.0                 0          32768 i
```

8. Configure R3 to set a Local Preference of 500 on all routes received from R4

Local preference indicates to routers which path is the best path to exit the autonomous system. In this case we are setting a local preference of 500 on all routes received from R4. Therefore, all routes that are received R4 to R3 will have a local preference of 500. On R3, there is only one route that is received from R4 so its local preference will be 500. Alongside creating a route-map, an inbound attribute on the neighbor is configured to determine that the next hop in R3 will have a local preference of 500.

```
router bgp 65007
 bgp router-id 7.3.3.3
 address-family ipv4
 neighbor 172.16.7.2 route-map LOCALPREF in
```

exit-address-family

route-map LOCALPREF permit 10
set local-preference 500

Router 3

```
R3#show ip bgp
BGP table version is 11, local router ID is 7.3.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found
R3#show ip bgp | include (.*500)
 * i 172.16.7.0/30    172.16.7.2               0    500      0 i
```

Task 7: Configure NAT

1. Configure NAT on R3 for all IPv4 connections to the Internet. Specifically, use NAT Overload (PAT) so that all outbound connections from 10.**x**.0.0/16 will be translated to the IP address assigned to the s0/1/1 interface of R3.

Configuring NAT overload so that when an outbound connection from 10.7.0.0/16 occurs, it is translated to the s0/1/1 interface of R3. To achieve this, we must first put an access list permitting the 10.7.0.0/16 address on R3. Then we need to define ip nat inside or outside depending on where the interfaces of R3 point. Finally, an overload command appending the access list is needed were it activates NAT Overload on the s0/1/1 interface.

CLI Command to configure NAT Overload

```
access-list 1 permit 10.7.0.0 0.0.255.255

interface s0/0/0
ip nat inside

interface GigabitEthernet0/0.20
ip nat inside

interface s0/1/1
ip nat outside

ip nat inside source list 1 interface s0/1/1 overload
```

Ping from R1	Output in R3. Below the translation results of inside local to inside global, the output is of nat debug to show translations being done. This snippet is an example of a ping from R1 to R3, showing a translation. When R1 (source) pinged to R3 (destination), its address got translated to 203.0.7.1 and in the second line, the destination address (203.0.7.1) is translated back to the source address of R1. Router 3 is tranalation packets in both directions.

```
R3#show ip nat translation
Pro Inside global      Inside local       Outside local      Outside global
--- 203.0.7.3          10.7.4.1           ---                ---
icmp 203.0.7.1:33      10.7.5.1:33        203.0.7.3:33       203.0.7.3:33
R3#
*Nov  1 03:00:03.338: NAT: s=10.7.5.1->203.0.7.1, d=203.0.7.3 [165]
*Nov  1 03:00:03.338: NAT: s=203.0.7.3, d=203.0.7.1->10.7.5.1 [165]
*Nov  1 03:00:03.394: NAT: s=10.7.5.1->203.0.7.1, d=203.0.7.3 [166]
*Nov  1 03:00:03.394: NAT: s=203.0.7.3, d=203.0.7.1->10.7.5.1 [166]
*Nov  1 03:00:03.450: NAT: s=10.7.5.1->203.0.7.1, d=203.0.7.3 [167]
*Nov  1 03:00:03.450: NAT: s=203.0.7.3, d=203.0.7.1->10.7.5.1 [167]
*Nov  1 03:00:03.506: NAT: s=10.7.5.1->203.0.7.1, d=203.0.7.3 [168]
*Nov  1 03:00:03.506: NAT: s=203.0.7.3, d=203.0.7.1->10.7.5.1 [168]
*Nov  1 03:00:03.562: NAT: s=10.7.5.1->203.0.7.1, d=203.0.7.3 [169]
*Nov  1 03:00:03.562: NAT: s=203.0.7.3, d=203.0.7.1->10.7.5.1 [169]
```

2. Create a static NAT mapping for the IPv4 address of R2's Loopback interface to the global address 203.0.**x**.3

For any device trying connect to 203.0.7.3, their ping will reach the 203.0.7.3 but will be translated to the loopback address of R2 (10.7.4.1). This is an example of one to one mapping where 203.0.7.3 is statically mapped to 10.7.4.1. To accomplish the mapping, we need to define which interfaces on R3 are part of nat inside or out side, then write a command like so, “ip nat inside source static 10.7.4.1 203.0.7.3” to issue a static translation (mapping).

CLI Command

```
interface s0/0/0
ip nat inside

interface GigabitEthernet0/0.20
ip nat inside

interface s0/1/1
ip nat outside

ip nat inside source static 10.7.4.1 203.0.7.3
```

Inside local is R2’s loopback and it is translated to 203.0.7.3

```
R3#show ip nat translation
Pro Inside global      Inside local       Outside local      Outside global
--- 203.0.7.3          10.7.4.1           ---                ---
```

Task 8: Connecting Pods:

1. **Create a tunnel interface on R3 running GRE over IPv4. The tunnel source should be s0/1/0 and the destination should be the address of the other pods R3 s0/1/0 interface. Give the tunnel interface the IPv6 address FEC0:1::x/64. The tunnel should not have any IPv4 address.**

Configuring a GRE tunnel over ipv4 with tunnel source and destination address and also configure ipv6 address on the tunnel interface

Purpose of GRE tunnel is to exchange packets using direct connection from one network to other network rather than passing all those packets through public network, which adds the risk of eyes-dropping, DDOS attacks. For tunnel to reach from one network to other network we need to have source and destination for tunnel to work. Also we can assign ipv4 or ipv6 addresses to the tunnel interface

To configure tunnel we enter global configuration mode now, we enter [interface tunnel0] command this will activate the tunnel0 now we will need to configure tunnel source and destination so that our network knows where to reach, right under tunnel0 interface we will add [tunnel source (interface)] command this will create a source through which all packets will go through. To create a Destination for our tunnel we use [tunnel destination (ip address of destination interface)] command to let packets know where to go. No shutdown command is necessary for the tunnel0 so that its status is up and it can be configure under tunnel0 interface using [no shutdown] command. To assign an ipv6 address we use [ipv6 (ipv6-address)] command under tunnel0 interface.

R3: Proof of tunnel over ipv4 without any ipv4 address configured	`R3#show ip int brief tunnel0`
	`Interface IP-Address OK? Method Status Prot ocol`
	`Tunnel0 unassigned YES unset up up`
	`R3#`

CLI Commands:	R3: interface Tunnel0 tunnel source Serial0/1/0 tunnel destination 172.16.50.1 no shut exit

R3: Proof of giving tunnel an ipv6 address	`R3#show ipv6 int brief tunnel0`
	`Tunnel0 [up/up]`
	`FE80::82E0:1DFF:FEB6:C8C8`
	`FEC0:1::7`
	`R3#`

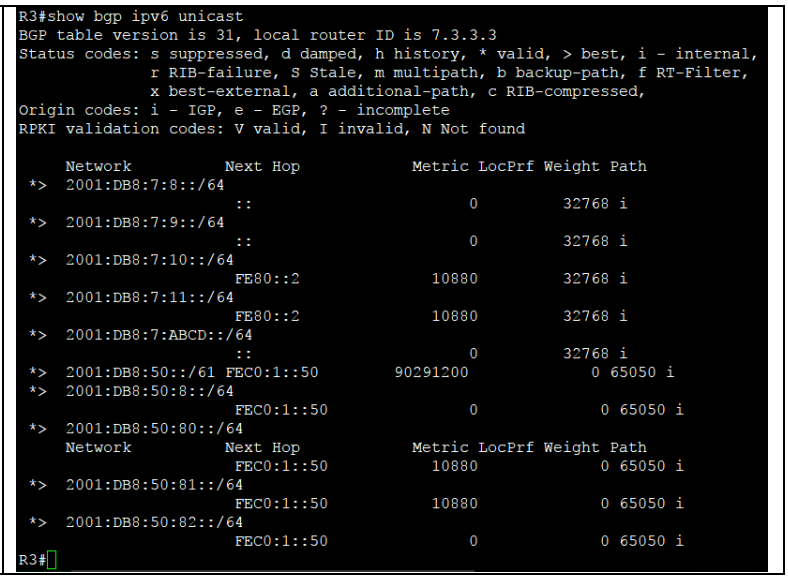
CLI Commands:	R3: interface Tunnel0 ipv6 address FEC0:1::7/64 no shut exit

2. **Configure MP-BGP on both R3 routers and form eBGP neighbor relationships between them using their FEC0:1::/64 addresses. These BGP routers should exchange only IPv6 routes.**

We are forming ebgp neighbor relationships between two-network using ipv6 address

Purpose of forming ebgp relationship is to be able to exchange packets from one network to other since they both are in two different Autonomous system

To configure this we entered bgp mode and then configure neighbor in this situation our neighbor is FEC0:1::50 in the Autonomous System 65050 so the command for this would be [neighbor FECO:1::50] remote-as 65050] this command will let our router R3 know that this is the neighbor that we are connected to in our ebgp neighbor relationship. Since, we only want to exchange ipv6 routes we will go under address family of ipv4 and type in [no neighbor FEC0:1::50 activate] this command will remove ipv4 as a carrier of ipv6 routes of that specific neighbor, and then we will activate it under ipv6 address family using [neighbor FEC0:1::50 activate] command which will make ipv6 passenger and carrier of only ipv6 routes.

R3: Proof of neighbor relationship between eBGP neighbors	

```
R3#show bgp ipv6 unicast
BGP table version is 31, local router ID is 7.3.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  2001:DB8:7:8::/64
                      ::                       0         32768 i
 *>  2001:DB8:7:9::/64
                      ::                       0         32768 i
 *>  2001:DB8:7:10::/64
                      FE80::2                10880         32768 i
 *>  2001:DB8:7:11::/64
                      FE80::2                10880         32768 i
 *>  2001:DB8:7:ABCD::/64
                      ::                       0         32768 i
 *>  2001:DB8:50::/61 FEC0:1::50          90291200            0 65050 i
 *>  2001:DB8:50:8::/64
                      FEC0:1::50               0             0 65050 i
 *>  2001:DB8:50:80::/64
     Network          Next Hop            Metric LocPrf Weight Path
                      FEC0:1::50             10880             0 65050 i
 *>  2001:DB8:50:81::/64
                      FEC0:1::50             10880             0 65050 i
 *>  2001:DB8:50:82::/64
                      FEC0:1::50               0             0 65050 i
R3#
```

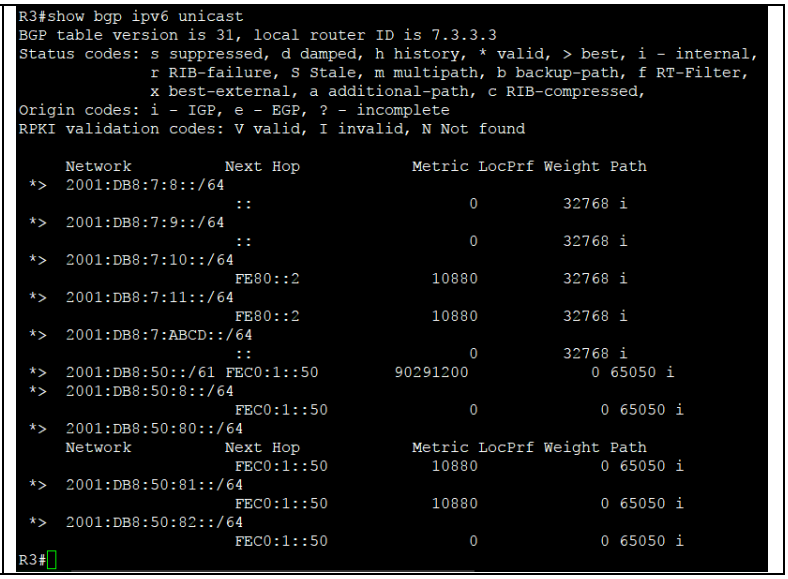
CLI Commands:

```
R3:
router bgp 65007
 neighbor FEC0:1::50 remote-as 65050
 address-family ipv4
no neighbor FEC0:1::50 activate
exit-address-family
 address-family ipv6
 neighbor FEC0:1::50 activate
 exit-address-family
```

3. **Advertise all subnets of the 2001:db8:x::/48 from R3 to the other pod, except any /128 routes you may have in your routing table.**

In this step we are advertising our ipv6 routes from R3 to other network's R3

To advertise you need to enter bgp mode, after that you need to enter into ipv6 address-family, here we will advertise our ipv6 routes using [network (ipv6-address)] command this will advertise all its ipv6 addresses to other network's R3 and it will show up in its routing table

R3: Proof of advertising ipv6 routes	

```
R3#show bgp ipv6 unicast
BGP table version is 31, local router ID is 7.3.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  2001:DB8:7:8::/64
                      ::                       0         32768 i
 *>  2001:DB8:7:9::/64
                      ::                       0         32768 i
 *>  2001:DB8:7:10::/64
                      FE80::2                10880         32768 i
 *>  2001:DB8:7:11::/64
                      FE80::2                10880         32768 i
 *>  2001:DB8:7:ABCD::/64
                      ::                       0         32768 i
 *>  2001:DB8:50::/61 FEC0:1::50          90291200             0 65050 i
 *>  2001:DB8:50:8::/64
                      FEC0:1::50               0             0 65050 i
 *>  2001:DB8:50:80::/64
     Network          Next Hop            Metric LocPrf Weight Path
                      FEC0:1::50             10880             0 65050 i
 *>  2001:DB8:50:81::/64
                      FEC0:1::50             10880             0 65050 i
 *>  2001:DB8:50:82::/64
                      FEC0:1::50               0             0 65050 i
R3#
```

CLI Commands:

```
R3:
router bgp 65007
address-family ipv6
  network 2001:DB8:7::/64
  network 2001:DB8:7:1::/64
  network 2001:DB8:7:2::/64
  network 2001:DB8:7:4::/64
  network 2001:DB8:7:5::/64
  network 2001:DB8:7:8::/64
  network 2001:DB8:7:9::/64
  network 2001:DB8:7:10::/64
  network 2001:DB8:7:11::/64
  network 2001:DB8:7:ABCD::/64
exit-address-family
```

4. **Form an eBGP neighbor relationship between R4 on your pod an R4 on the other pod. This relationship should be made using the IPv4 addresses on your Gi0/0/1.10 interfaces.**

To accomplish this step we will need to form ebgp relationship between R4 and other network's R4 using Ipv4 address

To configure this task we first enter bgp mode then, use [neighbor 199.212.32.50 remote-as 65050] command to advertise our bgp neighbor who is in a different Autonomous System. Once we have advertised our neighbor we

will have to enter ipv4 address-family and type [neighbor 199.212.32.50 next-hop-self] command to accept all ebgp routes and advertise them into ibgp.

CLI Commands:

```
R4:
router bgp 65007
 neighbor 199.212.32.50 remote-as 65050
 address-family ipv4
  neighbor 199.212.32.50 activate
  neighbor 199.212.32.50 next-hop-self
 exit-address-family
```

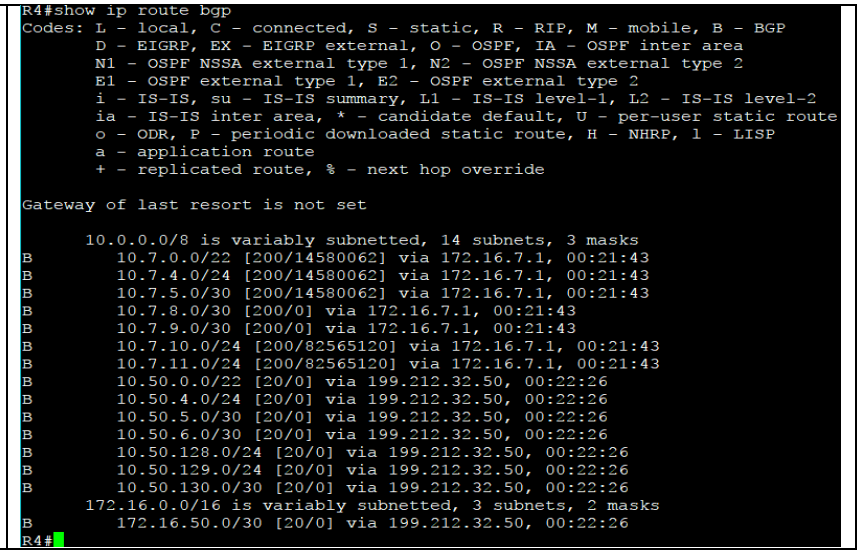
R3: Proof of advertising ipv6 routes

```
R3#show bgp ipv6 unicast
BGP table version is 31, local router ID is 7.3.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  2001:DB8:7:8::/64
                      ::                       0         32768 i
 *>  2001:DB8:7:9::/64
                      ::                       0         32768 i
 *>  2001:DB8:7:10::/64
                      FE80::2              10880         32768 i
 *>  2001:DB8:7:11::/64
                      FE80::2              10880         32768 i
 *>  2001:DB8:7:ABCD::/64
                      ::                       0         32768 i
 *>  2001:DB8:50::/61 FEC0:1::50        90291200             0 65050 i
 *>  2001:DB8:50:8::/64
                      FEC0:1::50               0             0 65050 i
 *>  2001:DB8:50:80::/64
     Network          Next Hop            Metric LocPrf Weight Path
                      FEC0:1::50           10880             0 65050 i
 *>  2001:DB8:50:81::/64
                      FEC0:1::50           10880             0 65050 i
 *>  2001:DB8:50:82::/64
                      FEC0:1::50               0             0 65050 i
R3#
```

5. **Advertise all IPv4 routes available on R4 to the other group via MP-BGP.**

In this step we are advertising our ipv4 routes from R4 to other network's R4

To advertise you need to enter bgp mode, after that you will need to enter into ipv4 address-family, here we will advertise our ipv4 routes using [network (ipv4-address)] command this will advertise all its ipv4 addresses to other network's R4 and it will show up in its routing table meaning that it has learned all the routes we have advertised

R4: Proof of advertising ipv4 routes	

CLI Commands:

```
R4:
router bgp 65007
  address-family ipv4
    network 10.7.0.0 mask 255.255.252.0
    network 10.7.4.0 mask 255.255.255.0
    network 10.7.5.0 mask 255.255.255.252
    network 10.7.8.0 mask 255.255.255.252
    network 10.7.9.0 mask 255.255.255.252
    network 10.7.10.0 mask 255.255.255.0
    network 10.7.11.0 mask 255.255.255.0
exit-address-family
```

6. **Configure R4 to set the MED to 50 on all IPv4 routes sent to the other pod.**

In this step we will configure MED to 50 for all ipv4 routes sent to the other network, this will notify other network about which enter point or path it should prefer when enter into my network.

To configure MED metric to 50 we first create a route map using [route-map MED permit 10] command to specify from what place you can send your routes to, under that command you will need to set med metric to 50 [set metric 50]. In the next, we enter bgp mode and then enter into our ipv4 address-family since we are only configuring MED 50 to all ipv4 routes. In our last step we will specify where are we using our route-map which is on our neighbors ip address so it will know that it have to use that path to come back to my network, command for that would be [neighbor 199.212.32.50 route-map MED out]

CLI Commands:

```
R4:

route-map MED permit 10
 set metric 50
 exit
```

```
router bgp 65007
  address-family ipv4
  neighbor 199.212.32.50 route-map MED out
  exit-address-family
exit
```

Task 9: Testing

R2-VRF

foreach address {

203.0.7.3

203.0.7.2

203.0.7.1

10.7.0.1

10.7.1.1

10.7.2.1

10.7.2.2

10.7.4.1

10.7.5.1

10.7.5.2

10.7.8.1

10.7.8.2

10.7.9.1

10.7.9.2

10.7.10.1

10.7.11.1

172.16.7.1

172.16.7.2

2001:db8:7:0::1

2001:db8:7:1::1

2001:db8:7:2::1

2001:db8:7:2::2

2001:db8:7:4::1

2001:db8:7:5::1

2001:db8:7:5::2

2001:db8:7:8::1

2001:db8:7:8::2

2001:db8:7:9::1

2001:db8:7:9::2

2001:db8:7:10::1

2001:db8:7:11::1

2001:db8:7:abcd::1

2001:db8:7:abcd::2

} { ping vrf VRF-EIGRP $address }

R2-VRF: IPv4

```
R2(tcl)#foreach address {
+>(tcl)#10.7.0.1
+>(tcl)#10.7.1.1
+>(tcl)#10.7.2.1
+>(tcl)#10.7.2.2
+>(tcl)#10.7.4.1
+>(tcl)#10.7.5.1
+>(tcl)#10.7.5.2
+>(tcl)#10.7.8.1
+>(tcl)#10.7.8.2
+>(tcl)#10.7.9.2
+>(tcl)#10.7.10.1
+>(tcl)#10.7.11.1
+>(tcl)#172.16.7.1
+>(tcl)#172.16.7.2
+>(tcl)#203.0.7.3
+>(tcl)#203.0.7.2
+>(tcl)#203.0.7.1
+>(tcl)#} { ping vrf VRF-EIGRP $address }
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.2.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.4.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.5.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.5.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 10.7.8.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.8.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.9.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.10.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.11.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.7.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.7.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.7.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.7.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.7.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
R2(tcl)#
```

R2-VRF: IPv6

```
R2(tcl)#foreach address {
+>(tcl)#2001:db8:7:0::1
+>(tcl)#2001:db8:7:1::1
+>(tcl)#2001:db8:7:2::1
+>(tcl)#2001:db8:7:2::2
+>(tcl)#2001:db8:7:4::1
+>(tcl)#2001:db8:7:5::1
+>(tcl)#2001:db8:7:5::2
+>(tcl)#2001:db8:7:8::1
+>(tcl)#2001:db8:7:8::2
+>(tcl)#2001:db8:7:9::2
+>(tcl)#2001:db8:7:10::1
+>(tcl)#2001:db8:7:11::1
+>(tcl)#2001:db8:7:abcd::1
+>(tcl)#2001:db8:7:abcd::2
+>(tcl)#} { ping vrf VRF-EIGRP $address }
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:1::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:2::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:2::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:4::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 52/54/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:5::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:5::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/32/48 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:8::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:8::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:9::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:10::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:11::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:ABCD::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:ABCD::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
R2(tcl)#
```

R1,R2, R3

foreach address {

10.7.0.1

10.7.1.1

10.7.2.1

10.7.2.2

10.7.4.1

10.7.5.1

10.7.5.2

10.7.8.1

10.7.8.2

10.7.9.1

10.7.9.2

10.7.10.1

10.7.11.1

172.16.7.1

172.16.7.2

203.0.7.1

203.0.7.2

203.0.7.3

2001:db8:7:0::1

2001:db8:7:1::1

2001:db8:7:2::1

2001:db8:7:2::2

2001:db8:7:4::1

2001:db8:7:5::1

2001:db8:7:5::2

2001:db8:7:8::1

2001:db8:7:8::2

2001:db8:7:9::1

2001:db8:7:9::2

2001:db8:7:10::1

2001:db8:7:11::1

2001:db8:7:abcd::1

2001:db8:7:abcd::2

} { ping $address }

R1: IPv4

```
R1(tcl)#foreach address {
+>(tcl)#10.7.0.1
+>(tcl)#10.7.1.1
+>(tcl)#10.7.2.1
+>(tcl)#10.7.2.2
+>(tcl)#10.7.4.1
+>(tcl)#10.7.5.2
+>(tcl)#10.7.8.2
+>(tcl)#10.7.9.1
+>(tcl)#10.7.9.2
+>(tcl)#10.7.10.1
+>(tcl)#10.7.11.1
+>(tcl)#172.16.7.1
+>(tcl)#172.16.7.2
+>(tcl)#203.0.7.3
+>(tcl)#203.0.7.2
+>(tcl)#203.0.7.1
+>(tcl)#} { ping $address }
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.2.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.4.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.5.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 52/55/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.8.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 10.7.9.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 24/27/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.9.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.10.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.11.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.7.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.7.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.7.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 24/27/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.7.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.7.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
R1(tcl)#
```

R1: IPv6

```
R1(tcl)#foreach address {
+>(tcl)#
+>(tcl)#
+>(tcl)#
+>(tcl)#2001:db8:7:0::1
+>(tcl)#2001:db8:7:1::1
+>(tcl)#2001:db8:7:2::1
+>(tcl)#2001:db8:7:2::2
+>(tcl)#2001:db8:7:4::1
+>(tcl)#2001:db8:7:5::2
+>(tcl)#2001:db8:7:8::2
+>(tcl)#2001:db8:7:9::1
+>(tcl)#2001:db8:7:9::2
+>(tcl)#2001:db8:7:10::1
+>(tcl)#2001:db8:7:11::1
+>(tcl)#2001:db8:7:abcd::1
+>(tcl)#2001:db8:7:abcd::2
+>(tcl)#2001:db8:7:abcd::
+>(tcl)#
+>(tcl)#
+>(tcl)#} { ping $address }
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:1::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:2::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:2::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:4::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:5::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:8::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:9::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:9::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:10::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:11::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:ABCD::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/33/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:ABCD::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 52/54/56 ms
Type escape sequence to abort.
```

R2: IPv4

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.2.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.4.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.5.1, timeout is 2 seconds:
```

```
Sending 5, 100-byte ICMP Echos to 10.7.8.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.8.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.9.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.10.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.11.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.7.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.7.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.7.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.7.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.7.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
R2(tcl)#
```

R2: IPv6

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:1::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:2::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:2::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:4::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 52/54/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:5::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:8::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:9::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:10::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:11::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:ABCD::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:ABCD::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
R2(tcl)#
```

R4

foreach address {

10.7.0.1

10.7.1.1

10.7.2.1

10.7.2.2

10.7.4.1

10.7.5.1

10.7.5.2

10.7.8.1

10.7.8.2

10.7.9.1

10.7.9.2

10.7.10.1

10.7.11.1

172.16.7.1

172.16.7.2

} { ping $address }

R4 IPv4

```
R4#tclsh
R4(tcl)#
R4(tcl)#
R4(tcl)#
R4(tcl)#foreach address {
+>(tcl)#10.7.0.1
+>(tcl)#10.7.1.1
+>(tcl)#10.7.2.1
+>(tcl)#10.7.2.2
+>(tcl)#10.7.4.1
+>(tcl)#10.7.5.1
+>(tcl)#10.7.5.2
+>(tcl)#10.7.8.1
+>(tcl)#10.7.8.2
+>(tcl)#10.7.9.1
+>(tcl)#10.7.9.2
+>(tcl)#10.7.10.1
+>(tcl)#10.7.11.1
+>(tcl)#172.16.7.1
+>(tcl)#172.16.7.2
+>(tcl)#} { ping $address }
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.2.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.4.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.5.1, timeout is 2 seconds:
```

```
Sending 5, 100-byte ICMP Echos to 10.7.8.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.8.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.9.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.10.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.11.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.7.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.7.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

R4-VRF

foreach address {

203.0.7.3

2001:db8:7:0::1

2001:db8:7:1::1

2001:db8:7:2::1

2001:db8:7:2::2

2001:db8:7:4::1

2001:db8:7:5::1

2001:db8:7:5::2

2001:db8:7:8::1

2001:db8:7:8::2

2001:db8:7:9::1

2001:db8:7:9::2

2001:db8:7:10::1

2001:db8:7:11::1

2001:db8:7:abcd::1

2001:db8:7:abcd::2

} { ping vrf VRF-INET $address }

R4 IPV6 VRF-INET

```
R4#tclsh
R4(tcl)#
R4(tcl)#
R4(tcl)#
R4(tcl)#foreach address {
+>(tcl)#2001:db8:7:0::1
+>(tcl)#2001:db8:7:1::1
+>(tcl)#2001:db8:7:2::1
+>(tcl)#2001:db8:7:2::2
+>(tcl)#2001:db8:7:4::1
+>(tcl)#2001:db8:7:5::1
+>(tcl)#2001:db8:7:5::2
+>(tcl)#2001:db8:7:8::1
+>(tcl)#2001:db8:7:8::2
+>(tcl)#2001:db8:7:9::1
+>(tcl)#2001:db8:7:9::2
+>(tcl)#2001:db8:7:10::1
+>(tcl)#2001:db8:7:11::1
+>(tcl)#2001:db8:7:abcd::1
+>(tcl)#2001:db8:7:abcd::2
+>(tcl)#} { ping vrf VRF-INET $address }
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:1::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:2::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:2::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:4::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 52/54/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:5::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:8::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:9::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:10::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:11::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:ABCD::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:ABCD::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
R2(tcl)#
```

Josh's Network:

foreach address {

10.50.0.1

10.50.1.1

10.50.2.1

10.50.4.1

10.50.5.1

10.50.6.1

10.50.128.1

10.50.129.1

10.50.130.1

172.16.50.1

2001:db8:50:0::1

2001:db8:50:1::1

2001:db8:50:2::1

2001:db8:50:4::1

2001:db8:50:8::1

2001:db8:50:80::1

2001:db8:50:81::1

2001:db8:50:82::1

fec0:1::50

} { ping $address }

R3: To Josh’s network

```
R3#tclsh
R3(tcl)#foreach address {
+>(tcl)#
+>(tcl)#
+>(tcl)#10.50.0.1
+>(tcl)#10.50.1.1
+>(tcl)#10.50.2.1
+>(tcl)#10.50.4.1
+>(tcl)#10.50.5.1
+>(tcl)#10.50.6.1
+>(tcl)#10.50.128.1
+>(tcl)#10.50.129.1
+>(tcl)#10.50.130.1
+>(tcl)#172.16.50.1
+>(tcl)#
+>(tcl)#
+>(tcl)#2001:db8:50:0::1
+>(tcl)#2001:db8:50:1::1
+>(tcl)#2001:db8:50:2::1
+>(tcl)#2001:db8:50:4::1
+>(tcl)#2001:db8:50:8::1
+>(tcl)#2001:db8:50:80::1
+>(tcl)#2001:db8:50:81::1
+>(tcl)#2001:db8:50:82::1
+>(tcl)#
+>(tcl)#
+>(tcl)#
+>(tcl)#} { ping $address }
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.50.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 84/84/88 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.50.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 84/88/104 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.50.2.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 84/86/92 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.50.4.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 84/84/88 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.50.5.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms
```

```
Sending 5, 100-byte ICMP Echos to 10.50.6.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/60 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.50.128.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/31/36 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.50.129.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/29/32 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.50.130.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.50.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/30/36 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:50::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 88/91/92 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:50:1::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 88/91/92 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:50:2::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 88/91/92 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:50:4::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 60/63/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:50:8::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 64/64/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:50:80::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/40/60 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:50:81::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/36/36 ms
```

```
Sending 5, 100-byte ICMP Echos to 2001:DB8:50:82::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/36/36 ms
R3(tcl)#
```

```
R3#ping FEC0:1::50
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FEC0:1::50, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/36/36 ms
R3#ping 199.212.32.50
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 199.212.32.50, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
R3#
```

R3-VRF

foreach address {

10.7.0.1

10.7.1.1

10.7.2.1

10.7.2.2

10.7.4.1

10.7.5.1

10.7.5.2

10.7.8.1

10.7.8.2

10.7.9.1

10.7.9.2

10.7.10.1

10.7.11.1

172.16.7.1

172.16.7.2

203.0.7.1

203.0.7.2

203.0.7.3

2001:db8:7:0::1

2001:db8:7:1::1

2001:db8:7:2::1

2001:db8:7:2::2

2001:db8:7:4::1

2001:db8:7:5::1

2001:db8:7:5::2

2001:db8:7:8::1

2001:db8:7:8::2

2001:db8:7:9::1

2001:db8:7:9::2

2001:db8:7:10::1

2001:db8:7:11::1

2001:db8:7:abcd::1

2001:db8:7:abcd::2

} { ping vrf VRF-OSPF $address }

R3-VRF

```
R3#tclsh
R3(tcl)#
R3(tcl)#
R3(tcl)#
R3(tcl)#foreach address {
+>(tcl)#10.7.0.1
+>(tcl)#10.7.1.1
+>(tcl)#10.7.2.1
+>(tcl)#10.7.2.2
+>(tcl)#10.7.4.1
+>(tcl)#10.7.5.1
+>(tcl)#10.7.5.2
+>(tcl)#10.7.8.1
+>(tcl)#10.7.8.2
+>(tcl)#10.7.9.1
+>(tcl)#10.7.9.2
+>(tcl)#10.7.10.1
+>(tcl)#10.7.11.1
+>(tcl)#172.16.7.1
+>(tcl)#172.16.7.2
+>(tcl)#203.0.7.1
+>(tcl)#203.0.7.2
+>(tcl)#203.0.7.3
+>(tcl)#} { ping vrf VRF-OSPF $address }
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.2.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.4.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.5.1, timeout is 2 seconds:
```

```
Sending 5, 100-byte ICMP Echos to 10.7.8.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.8.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.9.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.10.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.7.11.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.7.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.7.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.7.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.7.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.7.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
R2(tcl)#
```

R3 VRF- IPv6

```
R3(tcl)#foreach address {
+>(tcl)#2001:db8:7:0::1
+>(tcl)#2001:db8:7:1::1
+>(tcl)#2001:db8:7:2::1
+>(tcl)#2001:db8:7:2::2
+>(tcl)#2001:db8:7:4::1
+>(tcl)#2001:db8:7:5::1
+>(tcl)#2001:db8:7:5::2
+>(tcl)#2001:db8:7:8::1
+>(tcl)#2001:db8:7:8::2
+>(tcl)#2001:db8:7:9::1
+>(tcl)#2001:db8:7:9::2
+>(tcl)#2001:db8:7:10::1
+>(tcl)#2001:db8:7:11::1
+>(tcl)#2001:db8:7:abcd::1
+>(tcl)#2001:db8:7:abcd::2
+>(tcl)#} { ping vrf VRF-OSPF $address }
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/57/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:1::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:2::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:2::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:4::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 52/54/56 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:5::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:8::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:9::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:10::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:11::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:ABCD::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:7:ABCD::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
```

- **A web server has been configured to respond to HTTP requests on 198.51.100.1 port 80. From R3, in the VRF-OSPF VRF, telnet to this address on port 80, type GET, and verify that you receive a web response.**

CLI Command: First you need to enter exec mode Telnet 199.51.100.1 80

Security Features for Routing Protocols

EIGRP MD5 Authentication

1) What the chosen security feature does
    - All routers participating in EIGRP will have message authentication when they communicate. They will only become neighbors if both routers have identical message authentication. This is done so it prevents malicious routing information being introduced in the routing domain of the EIGRP router.

2) Why you chose this specific feature
    - This prevents a security vulnerability where an attacker may try to form neighbor relationship between a router participating in EIGRP. EIGRP will not accept neighbor relationships if the authentication does not match on either ends of the routers.

3) Where you implemented the feature in your network
    - The interfaces in EIGRP named mode had implementation of authentication. It is configured for IPv4 and IPv6.

4) The configurations you used to implement this feature

The first step in authenticating the messages in EIGRP is by creating the keychain and key and then finally the configuration of EIGRP authentication using the keychain and the key.

R1

```
key chain EIGRP1
 key 1
  key-string 0987654321
exit

router eigrp CASE2017

address-family ipv4 unicast
autonomous-system 7

af-interface Serial0/0/1
 authentication mode md5
 authentication key-chain EIGRP1
exit-af-interface


address-family ipv6 unicast
autonomous-system 7

af-interface Serial0/0/1
 authentication mode md5
 authentication key-chain EIGRP1

exit-address-family
```

R2-VRF

```
key chain EIGRP1
 key 1
  key-string 0987654321
exit

router eigrp CASE2017

address-family ipv4 unicast vrf VRF-EIGRP
autonomous-system 7


af-interface GigabitEthernet0/0.20
 authentication mode md5
 authentication key-chain EIGRP2
exit-af-interface
 exit-address-family

address-family ipv6 unicast vrf VRF-EIGRP
autonomous-system 7

af-interface GigabitEthernet0/0.20
 authentication mode md5
 authentication key-chain EIGRP2
exit-af-interface
```

R3

```
key chain EIGRP3
 key 1
  key-string 0987654321
exit

router eigrp CASE2017
 address-family ipv6 unicast autonomous-system 7
  af-interface GigabitEthernet0/0.20
   authentication mode md5
   authentication key-chain EIGRP3
   bandwidth-percent 64
  exit-af-interface

af-interface Serial0/0/0
   authentication mode md5
   authentication key-chain EIGRP3
   bandwidth-percent 64
  exit-af-interface
exit-address-family

 address-family ipv6 unicast autonomous-system 7
  af-interface GigabitEthernet0/0.20
   authentication mode md5
   authentication key-chain EIGRP3
```

```
   bandwidth-percent 64
  exit-af-interface

af-interface Serial0/0/0
   authentication mode md5
   authentication key-chain EIGRP3
   bandwidth-percent 64
  exit-af-interface
 exit-address-family
```

Link State Database Overload Protection

1) What the chosen security feature does
    - This features limits the routers participating in OSPF to generate non-self LSA's. Routers generate this type of LSA when routers are misconfigured and because of that they may generate high level of LSAs. This may "overload" the CPU and memory and potentially cause shortages. When this protection is enabled, routers keep a count of non-self generated LSA's received. When the count of the LSA's exceed the limit set for received non-self LSA's the router may shut down the OSPF process and clear the OSPF database. This can be configured in many ways for example a router does not have to shut down its OSPF process, it may just display a warning message.

2) Why you chose this specific feature
    - Routers can potentially in a big network may sometimes be misconfigured and therefore may create a high level of non-self generated LSA's and may potentially take down the network because of high CPU and memory usage. Therefore, it is best to keep such an action that handles this overload. Hence, why the name of the security feature is overload protection.

3) Where you implemented the feature in your network
    - This is configured after defining OSPFv3 process on the router.

4) The configurations you used to implement this feature

R1
`router ospfv3 7` `max-lsa 100 ignore-time 5 ignore-count 3 reset-time 5`
R2
`router ospfv3 7` `max-lsa 100 ignore-time 5 ignore-count 3 reset-time 5`
R3
`router ospfv3 7` `max-lsa 100 ignore-time 5 ignore-count 3 reset-time 5`

This configuration will allow 100 non-self generated under their ospfv3 process. After exceeding the max-lsa count of 100, the OSPF process will ignore all neighbors for 5 minutes, then after the neighbors are for 5 minutes the OSPF process will be in placed in ignored state for consequent count of 3 times and finally when the ignore count is 0, the reset time of OSPF process is set to 5 minutes. After reset time is over OSPF can perform its process.

BGP TTL Security Check

1) What the chosen security feature does
    - BGP (Time-To-Live) Security Check takes hop count into account for protection an eBGP session. eBGP will only allow neighbor relationships to form if the set hop count matches in the neighbor statement configured under the BGP process. BGP will maintain its session with a external BGP network only if the TTL value in the IP packet header is more than the TTL value configured for the current peering session. If the value is more than the configured value, BGP will not create a ICMP message and also the packet will be discarded silently.

2) Why you chose this specific feature
    - This feature protects against attackers who want to gain connection to a BGP network and perform malicious activity.

3) Where you implemented the feature in your network
    - This feature is implemented under the declaration of BGP process.

4) The configurations you used to implement this feature

Router 3
`router bgp 65007` `neighbor 199.212.32.50 ttl-security hops 2`

This is configured such that the hop count is set to 2 and only allow BGP to accept the packet header with a TTL count of equal or greater than 253. If the neighbor 199.212.32.50 is more than 2 hops away the eBGP peering session will not commence.

Security Features for the Network

Unicast Reverse Path Forwarding (uRPF)

1) What the chosen security feature does
    - This security tries to maintain legit traffic on the network. Upon configuration of this feature on per interface basis on each router, the router will verify that the source of any packets received is actually in the Cisco Express Forwarding table and also reachable via the routing table.

2) Why you chose this specific feature
    - This feature is chosen simply for that fact that it prevents malicious traffic on the network.

3) Where you implemented the feature in your network
    - This feature is implemented on a per interface basis on each router.

4) The configurations you used to implement this feature

The configuration is done so that the source of each packet is verified in the CEF and the routing table. Also the configuration is done in loose mode meaning the source IP of the received packet must be in the routing table of the router. There is an option to configure in strict mode but it might drop legitimate packets if asymmetric paths are present in the network. Therefore, to be on the safe side if the network were to have a asymmetric path in the network it is best to use loose mode.

CLI commands to configure uRPF on a per-interface-basis

R1
Interfaces s0/0/0 ip verify unicast source reachable-via any interface s0/0/1 ip verify unicast source reachable-via any
R2
Interface gi0/0.10 ip verify unicast source reachable-via any Interface gi0/0.20 ip verify unicast source reachable-via any Interfaces s0/0/0 ip verify unicast source reachable-via any
R3
Interface gi0/0.10 ip verify unicast source reachable-via any Interface gi0/0.20 ip verify unicast source reachable-via any Interfaces s0/0/0 ip verify unicast source reachable-via any Interfaces s0/1/0 ip verify unicast source reachable-via any Interfaces s0/1/1 ip verify unicast source reachable-via any
R4
Interface gi0/0/1.10 ip verify unicast source reachable-via any Interface gi0/0/1.20 ip verify unicast source reachable-via any Interfaces s0/1/0 ip verify unicast source reachable-via any Interfaces s0/1/1 ip verify unicast source reachable-via any

Logging Implementation

1) What the chosen security feature does
    - One of the most basic and important security feature is Logging. This feature logs any type of traffic on the network. May it be unusual or legitimate. It helps determine network problems that may occur on the network.

2) Why you chose this specific feature
    - This feature is chosen to monitor traffic on the network. At this point in time, social engineering is a big threat to networks and to overcome that our first line of defence should be logging to in future we can prevent something malicious. The purpose of this feature is to track legitimate traffic also so network administrators can identify improper functionality on devices.

3) Where you implemented the feature in your network
    - This feature is implemented on each router in global config mode.

4) The configurations you used to implement this feature

R1
enable configure terminal service timestamps log datetime
R2
enable configure terminal service timestamps log datetime
R3
enable configure terminal service timestamps log datetime
R4
enable configure terminal service timestamps log datetime

Each log created on devices will come with a date and time to specify at what point in date and time the log occurred on the network.

SSHv2:

1) What the chosen security feature does
    - SSHv2 provides strong security using Diffie-Hellman key. It generates 3DES and AES keys to transport packets from one router to other, it also provide Massage Authentication Code (MAC) for integrity check every time a unauthorized packets are being sent towards that specific network, and it records all failed unauthorized attempts made. We can also create ACLs to strengthen the security of the network; you can explicitly deny or permit any address you want. Once, the packet reaches the

point of security firstly, keys comparison will take place to see if it is an authenticated packet has been sent and, then it will be matched with ACL list. Therefore, even if someone tries to spoof the network and that spoofed address is not in the ACL list then, that packet will not pass that checkpoint. This increase overall security of our network.

2) Why you chose this specific feature
   - It provides better security then Telnet, and SSHv1, which are introduce to vulnerabilities, and keys are easily crackable but SSHv2 uses 3DES and AES to generate keys, which are two of the most secure key generator in present time.
3) Where you implemented the feature in your network
   - SSHv2 will be implemented on R3 and R4 because, on R3 we have a tunnel connection with some other network and on R4 we are connected to other network using BGP protocol
4) The configurations you used to implement this feature

R3
Enable configure terminal ip ssh version 2 ip access-list standard PERMIT-SSH remark ACL permitting SSH to hosts on the Management LAN permit 10.0.0.0 0.0.0.255 deny any log exit line vty 0 4 transport input ssh access-class PERMIT-SSH in end
R4
Enable configure terminal ip ssh version 2 ip access-list standard PERMIT-SSH remark ACL permitting SSH to hosts on the Management LAN permit 10.0.0.0 0.0.0.255 deny any log exit

Additional Changes in the Network

One of the two things that can be changed is not having both OSPF and EIGRP on the network at this point in time when the network is already small. Depending on the type of network or how far the Agency wants to scale their network its best to stick to one routing protocol for internal routing. At the current point in time, the company's network is small and therefore should only one IGP protocol to maintain simplicity. EIGRP is the best shout at this point to have over the network replacing OSPF because EIGRP is flat and only uses one area. However, when the company begins to scale and becomes more complex, they can think to add OSPF and use areas to segment the network and make it simplistic. Therefore, it depends on the company on what type of

network they decide to keep. This does not mean BGP will not be used in the network. BGP is an integral part of the network as it has the ability connects to external routing domains.

Second thing that we can implement in this topology is multiple direct connections between two routers to increase redundancy