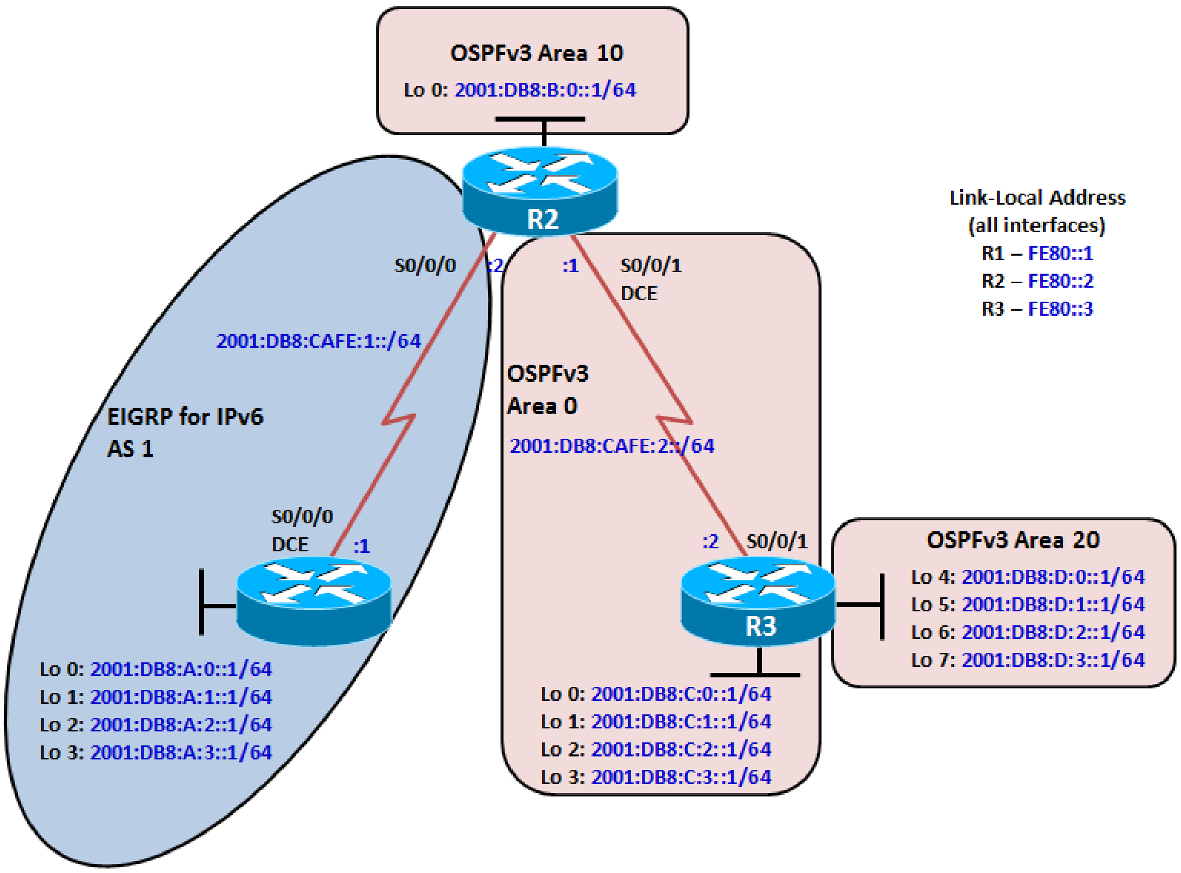
Chapter 4 Lab 4-3, Redistribution Between EIGRP for IPv6 and OSPFv3

Topology



Objectives

- Review EIGRP and OSPF configuration.
- Summarize routes in EIGRP.
- Summarize in OSPF at an ABR and an ASBR.
- Redistribute into EIGRP.
- Redistribute into OSPF.

Background

Two online booksellers, Example.com and Example.net, have merged and now need a short-term solution to inter-domain routing. Since these companies provide client services to Internet users, it is essential to have minimal downtime during the transition.

Example.com is running EIGRP while Example.net is running a multi-area OSPF. Because it is imperative that the two booksellers continuously deliver Internet services, you should bridge these two routing domains without interfering with each router's path through its own routing domain to the Internet.

The CIO determines that it is preferable to keep the two protocol domains shown in the diagram during the transition period, because the network engineers on each side need to understand the other's network before deploying a long-term solution. Redistribution will be a short-term solution.

In this scenario, R1 and R2 are running EIGRP while R2 is the OSPF autonomous system border router (ASBR) consisting of areas 0, 10, and 20. You need to configure R2 to enable these two routing protocols to interact to allow full connectivity between all networks.

In this lab, R1 is running EIGRP and R3 is running multi-area OSPF. Your task is to configure redistribution on R2 to enable these two routing protocols to interact, allowing full connectivity between all networks. In Appendix A of this lab, you explore the black hole operation.

Note: This lab uses Cisco 1941 routers with Cisco IOS Release 15.2 with IP Base. Depending on the router or switch model and Cisco IOS Software version, the commands available and output produced might vary from what is shown in this lab.

Required Resources

- 3 routers (Cisco IOS Release 15.2 or comparable)
- Serial and Ethernet cables

Step 1: Configure loopbacks and assign addresses.

a. Configure all loopback interfaces on the three routers in the diagram. Configure the serial interfaces with the IP addresses, bring them up, and set a DCE clock rate where appropriate.

```
R1(config-line)# interface Loopback0
R1(config-if)# ipv6 address 2001:db8:A:0::1/64
R1(config-if)# exit
R1(config)# interface Loopback1
R1(config-if)# ipv6 address 2001:db8:A:1::1/64
R1(config-if)# exit
R1(config)# interface Loopback2
R1(config-if)# ipv6 address 2001:db8:A:2::1/64
R1(config-if)# exit
R1(config)# interface Loopback3
R1(config-if)# ipv6 address 2001:db8:A:3::1/64
R1(config-if)# exit
R1(config)# interface serial 0/0/0
R1(config-if)# ipv6 address 2001:db8:cafe:1::1/64
R1(config-if)# ipv6 address fe80::1 link-local
R1(config-if)# clock rate 64000
R1(config-if)# no shutdown
R1(config-if)# exit
R1(config)#
*Oct 27 10:01:40.307: %LINEPROTO-5-UPDOWN: Line protocol on Interface
Loopback0, changed state to up
*Oct 27 10:01:40.711: %LINEPROTO-5-UPDOWN: Line protocol on Interface
Loopback1, changed state to up
*Oct 27 10:01:41.123: %LINEPROTO-5-UPDOWN: Line protocol on Interface
Loopback2, changed state to up
R1(config)#
*Oct 27 10:01:41.435: %LINEPROTO-5-UPDOWN: Line protocol on Interface
Loopback3, changed state to up
```

```
R1(config)#
*Oct 27 10:01:43.439: %LINK-3-UPDOWN: Interface Serial0/0/0, changed state to
down
R1(config)#
R1(config)#
*Oct 27 10:02:05.419: %LINK-3-UPDOWN: Interface Serial0/0/0, changed state to
up
*Oct 27 10:02:06.419: %LINEPROTO-5-UPDOWN: Line protocol on Interface
Serial0/0/0, changed state to up
R1(config)# exit
R1#

R2(config)# interface Loopback0
R2(config-if)# ipv6 address 2001:db8:B:0::1/64
R2(config-if)# exit
R2(config)#
R2(config)# interface serial 0/0/0
R2(config-if)# ipv6 address 2001:db8:cafe:1::2/64
R2(config-if)# ipv6 address fe80::2 link-local
R2(config-if)# no shutdown
R2(config-if)# exit
R2(config)#
R2(config)# interface serial 0/0/1
R2(config-if)# ipv6 address 2001:db8:cafe:2::1/64
R2(config-if)# ipv6 address fe80::2 link-local
R2(config-if)# clock rate 64000
R2(config-if)# no shutdown
R2(config-if)# exit
R2(config)#


R3(config)# interface Loopback0
R3(config-if)# ipv6 address 2001:db8:C:0::1/64
R3(config-if)# exit
R3(config)# interface Loopback1
R3(config-if)# ipv6 address 2001:db8:C:1::1/64
R3(config-if)# exit
R3(config)# interface Loopback2
R3(config-if)# ipv6 address 2001:db8:C:2::1/64
R3(config-if)# exit
R3(config)# interface Loopback3
R3(config-if)# ipv6 address 2001:db8:C:3::1/64
R3(config-if)# exit
R3(config)# interface Loopback4
R3(config-if)# ipv6 address 2001:db8:D:0::1/64
R3(config-if)# ipv6 address fe80::3 link-local
R3(config-if)# exit
R3(config)# interface Loopback5
R3(config-if)# ipv6 address 2001:db8:D:1::1/64
R3(config-if)# ipv6 address fe80::3 link-local
R3(config-if)# exit
R3(config)# interface Loopback6
R3(config-if)# ipv6 address 2001:db8:D:2::1/64
R3(config-if)# ipv6 address fe80::3 link-local
R3(config-if)# exit
R3(config)# interface Loopback7
R3(config-if)# ipv6 address 2001:db8:D:3::1/64
R3(config-if)# ipv6 address fe80::3 link-local
```

```
R3(config-if)#
R3(config)# interface serial 0/0/1
R3(config-if)# ipv6 address 2001:db8:cafe:2::2/64
R3(config-if)# ipv6 address fe80::3 link-local
R3(config-if)# clock rate 64000
R3(config-if)# no shutdown
R3(config-if)# exit
R3(config)#
```

b. Issue the **show ipv6 interface brief** command on each router and filter to include only the interface in an "up" status. Router R1 is shown as an example.

```
R1# show ipv6 interface brief | include up
Serial0/0/0            [up/up]
Loopback0              [up/up]
Loopback1              [up/up]
Loopback2              [up/up]
Loopback3              [up/up]
R1#
```

c. Verify that you can ping across the serial links when you are finished. Use the following Tcl script to check full and partial connectivity throughout this lab.

```
R1# tclsh

foreach address {
2001:db8:cafe:1::1
2001:db8:cafe:1::2
2001:db8:A:0::1
2001:db8:A:1::1
2001:db8:A:2::1
2001:db8:A:3::1
2001:db8:B:0::1
2001:db8:cafe:2::1
2001:db8:cafe:2::2
2001:db8:B:0::1
2001:db8:C:0::1
2001:db8:C:1::1
2001:db8:C:2::1
2001:db8:C:3::1
2001:db8:D:0::1
2001:db8:D:1::1
2001:db8:D:2::1
2001:db8:D:3::1
} { ping $address }
```

Which pings are successful and why?

____

____

____

____

Step 2: Configure EIGRP for IPv6.

a. Enable IPv6 unicast routing and EIGRP for IPv6 on each router. Since there are no active IPv4 addresses configured, EIGRP for IPv6 requires the configuration of a 32-bit router ID. Use the **router-id** command to configure the router ID in the router configuration mode.

   Note: Prior to IOS 15.2 the EIGRP IPv6 routing process is shut down by default and the no shutdown router configuration mode command is required to enable the routing process. Although not required with the IOS used in creating this lab, an example of the **no shutdown** command is shown for router R1.

   Issue the **ipv6 eigrp 1** command on the interfaces that participate in the EIGRP routing process. EIGRP for IPv6 does not use the **network** command. IPv6 prefixes are enabled on the interface. Similar to EIGRP for IPv4, the AS number must match the neighbor's configuration for the router to form an adjacency.

```
R1(config)# ipv6 unicast-routing
R1(config)# ipv6 router eigrp 1
R1(config-rtr)# eigrp router-id 1.1.1.1
R1(config-rtr)# no shutdown
R1(config-rtr)# exit
R1(config)# interface range lo 0 - 3
R1(config-if-range)# ipv6 eigrp 1
R1(config-if-range)# exit
R1(config)# interface s0/0/0
R1(config-if)# ipv6 eigrp 1
R1(config-if)#


R2(config)# ipv6 unicast-routing
R2(config)# ipv6 router eigrp 1
R2(config-rtr)# eigrp router-id 2.2.2.2
R2(config-rtr)# no shutdown
R2(config-rtr)# exit
R2(config)#
R2(config)# interface lo 0
R2(config-if)# ipv6 eigrp 1
R2(config-if)# exit
R2(config)#
R2(config)# interface s0/0/0
R2(config-if)# ipv6 eigrp 1
R2(config-if)# exit
R2(config)#
*Aug 26 09:45:14.347: %DUAL-5-NBRCHANGE: EIGRP-IPv6 1: Neighbor FE80::1
(Serial0/0/0) is up: new adjacency
R2(config)#
```

b. Verify the EIGRP configuration using the **show ipv6 eigrp neighbors** and **show ipv6 route eigrp** commands on R2.

```
R2# show ipv6 eigrp neighbors
EIGRP-IPv6 Neighbors for AS(1)
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   Link-local address:     Se0/0/0           12 00:32:18 1297   5000  0  3
    FE80::1
R2#
R2# show ipv6 route eigrp
```

```
<Output omitted>

D   2001:DB8:A::/64 [90/2297856]
     via FE80::1, Serial0/0/0
D   2001:DB8:A:1::/64 [90/2297856]
     via FE80::1, Serial0/0/0
D   2001:DB8:A:2::/64 [90/2297856]
     via FE80::1, Serial0/0/0
D   2001:DB8:A:3::/64 [90/2297856]
     via FE80::1, Serial0/0/0
R2#
```

c. Verify that R2 can reach all of the networks in the EIGRP for IPv6 routing domain using the following Tcl script.

```
R1# tclsh

foreach address {
2001:db8:cafe:1::1
2001:db8:cafe:1::2
2001:db8:A:0::1
2001:db8:A:1::1
2001:db8:A:2::1
2001:db8:A:3::1
} { ping $address }
```

All pings should be successful. Troubleshoot if necessary.

Step 3: Manually summarize with EIGRP for IPv6.

To make routing updates more efficient and ultimately reduce the size of routing tables, contiguous EIGRP routes can be summarized out an interface by using the **ipv6 summary-address eigrp** *as network mask* interface configuration command.

a. On R1, summarize the loopback interface networks.

```
R1(config)# interface s0/0/0
R1(config-if)# ipv6 summary-address eigrp 1 2001:db8:A::/62
R1(config-if)#
*Oct 27 11:05:33.019: %DUAL-5-NBRCHANGE: EIGRP-IPv6 1: Neighbor FE80::2
(Serial0/0/0) is resync: summary configured
R1(config-if)#
```

b. Verify the routing table of R2 using the **show ipv6 route eigrp** command.

```
R2# show ipv6 route eigrp

<Output omitted>

D   2001:DB8:A::/62 [90/2297856]
     via FE80::1, Serial0/0/0
R2#
```

c. Verify that R2 can still reach all of the networks in the EIGRP for IPv6 routing domain using the following Tcl script.

```
R1# tclsh
```

```
foreach address {
2001:db8:cafe:1::1
2001:db8:cafe:1::2
2001:db8:A:0::1
2001:db8:A:1::1
2001:db8:A:2::1
2001:db8:A:3::1
} { ping $address }
```

All pings should be successful. Troubleshoot if necessary.

Step 4: Configure OSPFv3 Address Family.

OSPFv3 with the addresses family (AF) unifies OSPF configuration for both IPv4 and IPv6. OSPFv3 with address families also combines neighbor tables and the LSDB under a single OSPF process. OSPFv3 messages are sent over IPv6 and therefore requires that IPv6 routing is enabled and that the interface has a link-local IPv6 address. This is the requirement even if only the IPv4 AF is configured.

a. On R2, configure OSPFv3 address family, router ID, and enable the OSPFv3 on the interface using the **ospfv3 1 ipv6 area** command.

```
R2(config)# ipv6 unicast-routing
R2(config)#
R2(config)# router ospfv3 1
R2(config-router)# address-family ipv6 unicast
*Aug 26 10:40:35.203: %OSPFv3-4-NORTRID: Process OSPFv3-1-IPv6 could not pick
a router-id, please configure manually
R2(config-router-af)# router-id 2.2.2.2
R2(config-router-af)# exit-address-family
R2(config-router)# exit
R2(config)#
R2(config)# interface Loopback0
R2(config-if)# ospfv3 1 ipv6 area 10
R2(config-if)# exit
R2(config)#
R2(config)# interface serial 0/0/1
R2(config-if)# ospfv3 1 ipv6 area 0
R2(config-if)# exit
R2(config)#
```

b. On R3, configure OSPFv3 address family, router ID, and enable the OSPFv3 on the interface using the **ospfv3 1 ipv6 area** command.

```
R3(config)# router ospfv3 1
R3(config-router)# address-family ipv6 unicast
R3(config-router-af)#
*Jul 28 03:10:48.395: %OSPFv3-4-NORTRID: Process OSPFv3-1-IPv6 could not pick
a router-id, please configure manually
R3(config-router-af)# router-id 3.3.3.3
R3(config-router-af)# exit-address-family
R3(config-router)# exit
R3(config)#
R3(config)# interface range lo 0 - 3
R3(config-if-range)# ospfv3 1 ipv6 area 0
R3(config-if-range)# exit
R3(config)#
R3(config)# interface range lo 4 - 7
```

```
R3(config-if-range)# ospfv3 1 ipv6 area 20
R3(config-if-range)# exit
R3(config)#
R3(config-if)# interface serial 0/0/1
R3(config-if)# ospfv3 1 ipv6 area 0
R3(config-if)# exit
R3(config)#
*Jul 28 03:20:29.267: %OSPFv3-5-ADJCHG: Process 1, IPv6, Nbr 2.2.2.2 on
Serial0/0/1 from LOADING to FULL, Loading Done
R3(config)#
R3(config)#
```

c. Verify that your adjacencies come up with the **show ipv6 ospf neighbor** command, and make sure that you have routes from OSPF populating the R2 routing table using the **show ipv6 route ospf** command.

```
R2# show ipv6 ospf neighbor

            OSPFv3 Router with ID (2.2.2.2) (Process ID 1)

Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
3.3.3.3           0   FULL/  -        00:00:31    6               Serial0/0/1
R2#
R2# show ipv6 route ospf

<Output omitted>

O   2001:DB8:C::1/128 [110/64]
     via FE80::3, Serial0/0/1
O   2001:DB8:C:1::1/128 [110/64]
     via FE80::3, Serial0/0/1
O   2001:DB8:C:2::1/128 [110/64]
     via FE80::3, Serial0/0/1
O   2001:DB8:C:3::1/128 [110/64]
     via FE80::3, Serial0/0/1
OI  2001:DB8:D::1/128 [110/64]
     via FE80::3, Serial0/0/1
OI  2001:DB8:D:1::1/128 [110/64]
     via FE80::3, Serial0/0/1
OI  2001:DB8:D:2::1/128 [110/64]
     via FE80::3, Serial0/0/1
OI  2001:DB8:D:3::1/128 [110/64]
     via FE80::3, Serial0/0/1
R2#
```

d. Verify the OSPF IPv6 routing table of R3.

```
R3# sho ipv6 route ospf

<Output omitted>

OI  2001:DB8:B::1/128 [110/64]
     via FE80::2, Serial0/0/1
R3#
```

e. Verify that R2 and R3 can reach all of the networks in the OSPFv3 routing domain using the following Tcl script.

```
R3# tclsh
```

```
foreach address {
2001:db8:B:0::1
2001:db8:cafe:2::1
2001:db8:cafe:2::2
2001:db8:B:0::1
2001:db8:C:0::1
2001:db8:C:1::1
2001:db8:C:2::1
2001:db8:C:3::1
2001:db8:D:0::1
2001:db8:D:1::1
2001:db8:D:2::1
2001:db8:D:3::1
} { ping $address }
```

All pings should be successful. Troubleshoot if necessary.

Step 5: Configure mutual redistribution between OSPFv3 and EIGRP for IPv6.

Notice that R2 is the only router with knowledge of all routes (EIGRP for IPv6 and OSPFv3) in the topology at this point, because it is involved with both routing protocols. Next you will redistribute the EIGRP for IPv6 routes into OSPFv3 and the OSPFv3 routes into EIGRP for IPv6.

a. To redistribute the EIGRP for IPv6 routes into OSPFv3, on R2 issue the **redistribute eigrp 1 include-connected** command.

```
R2(config)# router ospfv3 1
R2(config-router)# address-family ipv6 unicast
R2(config-router-af)# redistribute eigrp 1 include-connected
R2(config-router-af)# exit
R2(config-router)# exit
```

A default seed metric is not required for OSPFv3. Redistributed routes are assigned a metric of 20 by default.

b. To redistribute the OSPFv3 routes into EIGRP for IPv6, on R2 issue the **redistribute ospf 1 metric 10000 100 255 1 1500** command. Unlike OSPFv3, EIGRP for IPv6 must specify the metric associated to the redistributed routes. The command tells EIGRP to redistribute OSPF process 1 with these metrics: bandwidth of 10000, delay of 100, reliability of 255/255, load of 1/255, and a MTU of 1500.

```
R2(config)# ipv6 router eigrp 1
R2(config-rtr)# redistribute ospf 1 metric 1500 100 255 1 1500 include-
connected
R2(config-rtr)# exit
R2(config)#
```

c. Issue the **show ipv6 protocols** command on the redistributing router, R2. Compare your output with the following output.

```
R2# show ipv6 protocols
IPv6 Routing Protocol is "connected"
IPv6 Routing Protocol is "application"
IPv6 Routing Protocol is "ND"
IPv6 Routing Protocol is "eigrp 1"
EIGRP-IPv6 Protocol for AS(1)
  Metric weight K1=1, K2=0, K3=1, K4=0, K5=0
  NSF-aware route hold timer is 240
  Router-ID: 2.2.2.2
  Topology : 0 (base)
    Active Timer: 3 min
    Distance: internal 90 external 170
    Maximum path: 16
    Maximum hopcount 100
    Maximum metric variance 1

  Interfaces:
    Loopback0
    Serial0/0/0
  Redistribution:
    Redistributing protocol ospf 1 with metric 1500 100 255 1 1500 (internal,
external 1 & 2, nssa-external 1 & 2) include-connected
IPv6 Routing Protocol is "ospf 1"
  Router ID 2.2.2.2
  Area border and autonomous system boundary router
  Number of areas: 2 normal, 0 stub, 0 nssa
  Interfaces (Area 0):
    Serial0/0/1
  Interfaces (Area 10):
    Loopback0
  Redistribution:
    Redistributing protocol eigrp 1 include-connected
R2#
```

d. Display the routing table on R1 to verify the redistributed routes. Redistributed OSPFv3 routes display on R1 as EX, which means that they are external EIGRP for IPv6 routes.

```
R1# show ipv6 route eigrp

<Output omitted>

D   2001:DB8:A::/61 [5/128256]
     via Null0, directly connected
D   2001:DB8:B::/64 [90/2297856]
     via FE80::2, Serial0/0/0
EX  2001:DB8:C::1/128 [170/2244096]
     via FE80::2, Serial0/0/0
EX  2001:DB8:C:1::1/128 [170/2244096]
     via FE80::2, Serial0/0/0
EX  2001:DB8:C:2::1/128 [170/2244096]
     via FE80::2, Serial0/0/0
EX  2001:DB8:C:3::1/128 [170/2244096]
     via FE80::2, Serial0/0/0
EX  2001:DB8:D::1/128 [170/2244096]
     via FE80::2, Serial0/0/0
EX  2001:DB8:D:1::1/128 [170/2244096]
     via FE80::2, Serial0/0/0
EX  2001:DB8:D:2::1/128 [170/2244096]
     via FE80::2, Serial0/0/0
EX  2001:DB8:D:3::1/128 [170/2244096]
     via FE80::2, Serial0/0/0
EX  2001:DB8:CAFE:2::/64 [170/2244096]
     via FE80::2, Serial0/0/0
R1#
```

e. Display the routing table on R3 to see the redistributed routes. Redistributed EIGRP routes are tagged in the R3 routing table as O E2, which means that they are OSPF external type 2. Type 2 is the default OSPF external type.

```
R3# show ipv6 route ospf

<Output omitted>

OE2 2001:DB8:A::/61 [110/20]
     via FE80::2, Serial0/0/1
OI  2001:DB8:B::1/128 [110/64]
     via FE80::2, Serial0/0/1
OE2 2001:DB8:CAFE:1::/64 [110/20]
     via FE80::2, Serial0/0/1
R3#
```

f. Verify full connectivity with the following Tcl script:

```
R1# tclsh


foreach address {
2001:db8:cafe:1::1
2001:db8:cafe:1::2
2001:db8:A:0::1
2001:db8:A:1::1
2001:db8:A:2::1
2001:db8:A:3::1
2001:db8:B:0::1
2001:db8:cafe:2::1
2001:db8:cafe:2::2
2001:db8:B:0::1
2001:db8:C:0::1
2001:db8:C:1::1
2001:db8:C:2::1
2001:db8:C:3::1
2001:db8:D:0::1
2001:db8:D:1::1
2001:db8:D:2::1
2001:db8:D:3::1
} { ping $address }
```

All pings should now be successful. Troubleshoot as necessary.