Steven Kim

Mr. Mason

Advanced Cisco CCNP, P1,2

4/25/14

Lab 10: BGP Part 2

Purpose

The purpose of this lab was to implement IBGP that simultaneously runs IPv4 and IPv6 (although EBGP was used in the core router). Most of basic BGP was introduced in part 1; part 2 was an application of we learned in part 1.

Background Information on Lab Concepts

The first part of this lab (BGP Part 1) has already compared IBGP to EBGP. The main difference is their use of autonomous system numbers: while EBGP runs different autonomous systems, IBGP runs a single autonomous system.

To review, the following are the differences between IBGP and EBGP:

- IBGP (Internal BGP): A BGP that processes a single autonomous system.
    - The main requirement for this BGP is that peers must be in full mesh mode, meaning that a direct connection needs to be established between routes.
    - The administrative distance of IBGP is 200
- EBGP (Exterior BGP): A BGP that runs different autonomous systems.
    - The administrative distance of EBGP is 20.

The main difference between IBGP and EBGP is the way that a route is forwarded to another route. If a route is learned from EBGP, it will be redistributed to all IBGP and EBGP peers. On the other hand, if a route is learned from IBGP, it will be redistributed to only EBGP peers. Since IBGP establishes a full mesh topology, the speed of IBGP will be much faster than that of EBGP. In addition, when EBPG and IBGP are both configured on the same network, IBGP, whose administrative distance of the two BGP is lower, will have the priority.

In this situation where multiple companies are trying to communicate with each other, EBGP is needed, for it can manage different autonomous systems that different companies will have. However, when a company is trying to communicate with another enterprise of itself, IBGP establishes a full mesh that allows a more efficient transfer of information.

When implementing IBGP for both IPv4 and IPv6, however, several different concepts were introduced:

- IPv6 (Internet Protocol version 6) - The most recent version of IPs that has replaced Internet Protocol version 4 (IPv4). This implementation was required for the growing need for new addresses, for IPv6 could provide more than people needed.

- Address Families - A feature of BGP that congregates a certain group of devices under the name "address family." Due to this grouping feature, address families can separate IPv4 and IPv6 routes. If, by any chance, a wrong IP address is put under a family, routes would fail. For example, if a router is physically adjacent to a router but is not included in the address family, that router will not have a proper route propagated to it.
- Multiprotocol BGP: A type of BGP that employs address families to run both IPv4 and IPv6. Since by default BGP classifies unspecified routes and addresses as a part of an IPv4 family, this protocol is essential for efficiently managing both IPv4 and IPv6.
- RIPng: Also known as Routing Information Protocol New Generation, a modified version of RIPv2 that uses IPv6. Unlike RIPv2 that requires a manual route command, RIPng automatically figures out routes and finds next hop addresses. The UDP port number for RIPng is 521, and the multicast address for it is FF02::9. For security, RIPng uses IPsec.
- OSPFv3: Also known as Open Shortest Path First version 3, a protocol that makes use of IPv6. OSPFv3 uses 32-bit identifiers to find areas and IPsec for security. Just like OSPF, OSPFv3 governs routes based on areas. Using Dijkstra's algorithm, it comes up with the shortest path tree, using a shortest first path algorithm.
- IPv6 EIGRP: Enhanced Interior Gateway Routing Protocol that runs EIGRP. As a distance vector protocol, EIGRP uses the DUAL algorithm to manage routes within the same network, under a single autonomous system number. In the Neighbor Table of this routing protocol, there are routes from neighbors that maintain direct connections with the router running EIGRP. The Topology Table stores metrics for determining successors and feasible successors that propagate routes.

Lab Summary

Similar to the first part of BGP, this lab required various commands to be issued.

1. Connect devices with the appropriate cables, as shown in the topology below (in Network Diagram with IP's)
2. Assign IPv4 and IPv6 addresses to each interface. When an IPv6 address cannot be assigned, issue the command *Router (config)# ipv6 unicast-routing*. On switches, make sure to issue the command *Switch (config)# no switchport and Switch (config)# ip routing*.
3. Configure RIP, RIPng, EIGRP, OSPF, and OSPFv3 on the interfaces shown below.
4. For RIPng, the commands *Router (config)# ipv6 enable* and *Router (config)# ipv6 rip [name of RIPng] enable* must be issued. If the latter isn't assigned and thus a RIPng does not have a name, RIPng cannot be configured. Remember that RIPng CANNOT be assigned in the global configuration mode but only in the interface configuration mode.
5. For OSPFv3, enter the configuration mode using the command *Router (config)# ipv6 router ospf [ospf number].* Assign a network address and a subnet mask using the command
6. For IPv6 EIGRP, issue the command *Router (config)# ipv6 router eigrp [autonomous system number].* Then, enter the command Router (config-router)# eigrp router-id [router id of the router] to configure a router ID.
7. After the three routing protocols have been configured, set up BGP. To do so, first enter the configuration mode by issuing the command *Router (config)# router bgp [autonomous system number].* Enter the command *Router (config-router)# no bgp default ipv4-unicast* to enable IPv6.

8. Next, set up the neighbor addresses using the command *Router (config-router)# neighbor [ipv4 or ipv6 address] remote-as [autonomous system number]*.
9. Enter the address family mode by using the command *Router (config-router)# address-family [ipv4 or ipv6].* As mentioned above, address families help BGP distinguish IPv4 and IPv6 addresses.
10. Redistribute all the routing protocols using the command *Router (config-router)# redistribute [routing protocol] [autonomous system number or OSPF number].* This is exactly like part 1.
11. Ping from one loopback to another to verify connectivity within the network.

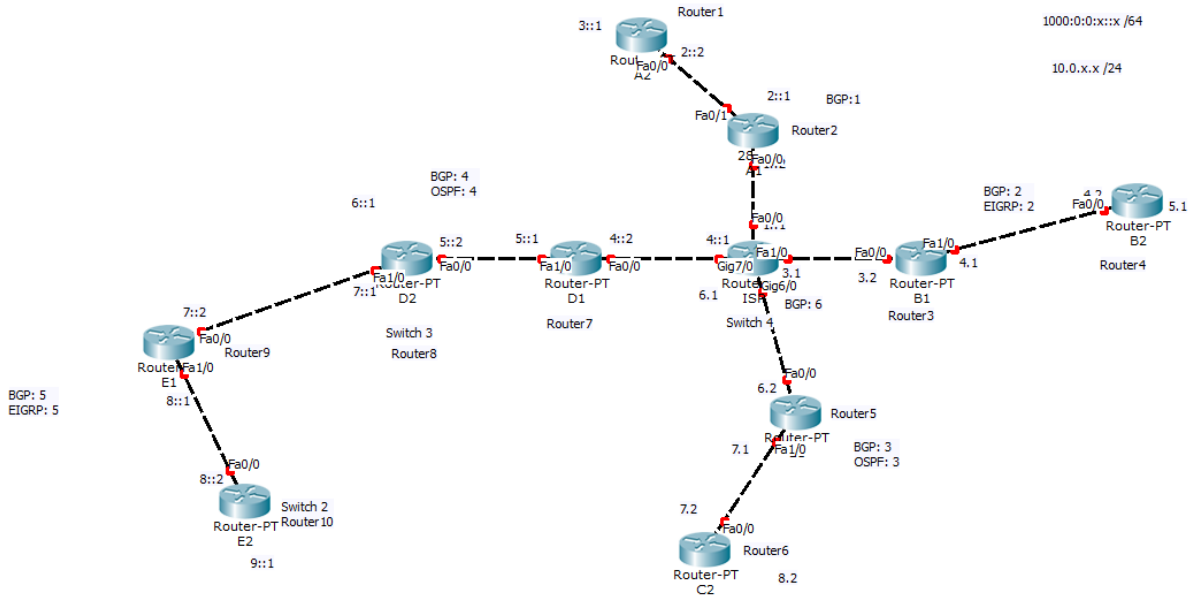
Lab Commands

Router (config-if)# ipv6 enable	Enables the configuration of IPv6 addresses and routes
Router (config-if)# ipv6 address [ipv6 address of the interface]	Configures an IPv6 address on the interface.
Router (config-if)# ipv6 rip [name of RIPng] enable	Enables RIPng with the given name.
Router (config-if)# ipv6 router rip [name of RIPng]	Configures RIPng on the interface. No network commands are needed because RIPng will automatically figure out adjacent networks.
Router (config)# router bgp [autonomous system number]	Lets the user enter the BGP configuration mode.
Router (config-router)# no bgp default ipv4-unicast	As mentioned in Background Information on Lab Concepts, BGP puts all unspecified addresses under IPv4 families. To enable IPv6 with BGP, this command is needed.
Router (config-router)# neighbor [ipv4 or ipv6 address] remote-as [autonomous system number]	Configures the neighbor address with the given autonomous system numbers and IPv4 or IPv6 address.
Router (config-router)# address-family ipv4	Lets the user enters the IPv4 address family configuration mode.
Router (config-router-af)# neighbor [ipv4 address of the neighbor interface] activate	Confirms the neighbor IPv4 address on the given interface.
Router (config-router-af)# network [network address of the neighbor interface] mask [subnet mask]	Configures the network IPv4 address and subnet mask of the given interface.
Router (config-router)# address-family ipv6	Lets the user enters the IPv6 address family configuration mode.
Router (config-router-af)# neighbor [ipv6 address of the neighbor interface] activate	Confirms the neighbor IPv6address on the given interface.
Router (config-router-af)# network [network address of the neighbor interface/subnet mask]	Configures the network IPv6 address and subnet mask of the given interface.
Router (config-router-af)# redistribute [routing protocol]	Redistributes the given routing protocol.
Router (config)# ipv6 router ospf [ospf number]	Enters the configuration mode for OSPFv3
Router (config-router)# network [network address of ipv6 ospf]	Configures the network IPv6 address on OSPFv3.
Router (config-router)# redistribute [routing	Redistributes the given routing protocol.

protocol]	
Router (config)# ipv6 router eigrp [autonomous system number]	Allows the user to enter the eigrp configuration mode.
Router (config-router)# [routing protocol] router-id [router id of the router]	For all protocols, this command configures the router ID.

Network Diagram with IP's

*NOTE: company A uses RIPng



Configurations

A1

```
hostname A1
!
ip routing
ipv6 unicast-routing
!
interface fa0/0
 ipv6 enable
 ipv6 address 1000:0:0:1::2/64
 ipv6 rip A_RIPNG enable
 no shutdown
 exit
!
interface fa0/1
 ipv6 enable
 ipv6 address 1000:0:0:2::1/64
 ipv6 rip A_RIPNG enable
 no shutdown
 exit
!
router bgp 1
 no bgp default ipv4-unicast
 bgp router-id 2.0.0.0
 neighbor 1000:0:0:1::1 remote-as 6
 !
 address-family ipv6
  neighbor 1000:0:0:1::1 activate
  network 1000:0:0:2::/64
  redistribute rip A_RIPNG
  exit
 !
!
ipv6 router rip A_RIPNG
 redistribute bgp 1 metric 10
 exit
!
line con 0
 logging synchronous
 exit
line vty 0 4
 logging synchronous
```

```
 exit
!
end
```

A2

```
hostname A2
!
ipv6 unicast-routing
!
interface fa0/0
 ipv6 enable
 ipv6 address 1000:0:0:2::2/64
 ipv6 rip A_RIPNG enable
 no shutdown
 exit
!
interface loopback 0
 ipv6 enable
 ipv6 address 1000:0:0:3::1/64
 ipv6 rip A_RIPNG enable
 no shutdown
 exit
!
ipv6 router rip A_RIPNG
 exit
!
line con 0
 logging synchronous
 exit
line vty 0 4
 logging synchronous
 exit
!
end
```

B1

```
hostname B1
!
interface g0/0
 ip address 10.0.3.2 255.255.255.0
 no shutdown
 exit
!
interface g0/1
 ip address 10.0.4.1 255.255.255.0
 no shutdown
 exit
!
router bgp 2
 no bgp default ipv4-unicast
 bgp router-id 3.0.0.0
 neighbor 10.0.3.1 remote-as 6
 !
 address-family ipv4
  neighbor 10.0.3.1 activate
  network 10.0.3.0 mask 255.255.255.0
  redistribute eigrp 2
  exit
 !
 exit
!
router eigrp 2
 network 10.0.4.0 0.0.0.255
 redistribute bgp 2 metric 10 10 255
255 1000
 no auto-summary
 exit
!
line con 0
 logging synchronous
 exit
line vty 0 4
 logging synchronous
 exit
!
end
```

B2

```
hostname B2
!
interface g0/0
 ip address 10.0.4.2 255.255.255.0
 no shutdown
 exit
!
interface loopback 0
 ip address 10.0.5.1 255.255.255.0
 no shutdown
 exit
!
router eigrp 2
 no auto-summary
 network 10.0.5.0 0.0.0.255
 network 10.0.4.0 0.0.0.255
 exit
!
line con 0
 logging synchronous
 exit
line vty 0 4
 logging synchronous
 exit
!
end
```

C1

```
hostname C1
!
ipv6 unicast-routing
!
interface g0/0
 ip address 10.0.6.2 255.255.255.0
 no shutdown
 exit
!
interface g0/1
 ip address 10.0.7.1 255.255.255.0
 no shutdown
 exit
```

```
!
router bgp 3
 no bgp default ipv4-unicast
 bgp router-id 4.0.0.0
 neighbor 10.0.6.1 remote-as 6
 !
 address-family ipv4
  neighbor 10.0.6.1 activate
  network 10.0.6.0 mask 255.255.255.0
  redistribute ospf 3
  exit
 !
 exit
!
router ospf 3
 redistribute bgp 3 subnets
 network 10.0.7.0 0.0.0.255 area 0
 exit
!
line con 0
 logging synchronous
 exit
line vty 0 4
 logging synchronous
 exit
!
end
```

C2

```
hostname C2
!
ipv6 unicast-routing
!
interface g0/0
 ip address 10.0.7.2 255.255.255.0
 no shutdown
 exit
!
interface loopback 0
 ip address 10.0.8.2 255.255.255.0
 no shutdown
 exit
!
router ospf 3
 network 10.0.7.0 0.0.0.255 area 0
 network 10.0.8.0 0.0.0.255 area 0
 exit
!
line con 0
 logging synchronous
 exit
line vty 0 4
 logging synchronous
 exit
!
end
```

D1

```
hostname D1
!
ipv6 unicast-routing
!
interface gig0/0
 ipv6 enable
 ipv6 address 1000:0:0:4::2/64
 ipv6 ospf 4 area 0
 no shutdown
 exit
!
interface gig0/1
 ipv6 enable
 ipv6 address 1000:0:0:5::1/64
 ipv6 ospf 4 area 0
 no shutdown
 exit
!
router bgp 4
 no bgp default ipv4-unicast
 bgp router-id 5.0.0.0
 neighbor 1000:0:0:4::1 remote-as 6
 neighbor 1000:0:0:5::2 remote-as 4
 !
 address-family ipv6
  neighbor 1000:0:0:4::1 activate
  neighbor 1000:0:0:5::2 activate
  network 1000:0:0:5::/64
  redistribute ospf 4
  exit
 !
 exit
!
ipv6 router ospf 4
 router-id 5.0.0.0
 redistribute bgp 4
!
end
```

D2

```
hostname D2
!
ip routing
ipv6 unicast-routing
!
interface fa0/1
 ipv6 enable
 ipv6 address 1000:0:0:5::2/64
 ipv6 ospf 4 area 0
 no shutdown
 exit
!
interface fa0/2
 ipv6 enable
 ipv6 address 1000:0:0:7::1/64
 ipv6 ospf 4 area 0
 no shutdown
 exit
!
interface loopback0
```

```
 ipv6 enable
 ipv6 address 1000:0:0:6::1/64
 ipv6 ospf 4 area 0
 no shutdown
!
router bgp 4
 no bgp default ipv4-unicast
 bgp router-id 5.1.0.0
 neighbor 1000:0:0:5::1 remote-as 4
 neighbor 1000:0:0:7::2 remote-as 5
 !
 address-family ipv6
  neighbor 1000:0:0:5::1 activate
  neighbor 1000:0:0:7::2 activate
  network 1000:0:0:5::/64
  network 1000:0:0:7::/64
  redistribute ospf 4
  exit
!
ipv6 router ospf 4
 router-id 5.1.0.0
 network 1000:0:0:11::/64
 redistribute bgp 4
!
end
```

E1

```
hostname E1
!
ip routing
ipv6 unicast-routing
!
interface gig0/0
 ipv6 enable
 ipv6 address 1000:0:0:7::2/64
 ipv6 eigrp 5
 no shutdown
 exit
!
interface gig0/1
 ipv6 enable
 ipv6 address 1000:0:0:8::1/64
 ipv6 eigrp 5
 no shutdown
 exit
!
router bgp 5
 no bgp default ipv4-unicast
 bgp router-id 6.0.0.0
 neighbor 1000:0:0:7::1 remote-as 4
 !
 address-family ipv6
  neighbor 1000:0:0:7::1 activate
  network 1000:0:0:8::/64
  redistribute eigrp 5
  exit
 !
 exit
!
ipv6 router eigrp 5
 eigrp router-id 6.0.0.0
 redistribute bgp 5 metric 10 10 255
255 1000
 exit
!
line con 0
 logging synchronous
 exit
line vty 0 4
 logging synchronous
 exit
!
end
```

E2

```
hostname E2
!
ip routing
ipv6 unicast-routing
!
interface fa0/1
 no switchport
 ipv6 enable
 ipv6 address 1000:0:0:8::2/64
 ipv6 eigrp 5
 no shutdown
 exit
!
interface loopback0
 ipv6 enable
 ipv6 address 1000:0:0:9::1/64
 ipv6 eigrp 5
 no shutdown
 exit
!
ipv6 router eigrp 5
 eigrp router-id 6.1.0.0
 exit
!
line con 0
 logging synchronous
 exit
line vty 0 4
 logging synchronous
 exit
!
end
```

ISP

```
hostname ISP
!
ip routing
ipv6 unicast-routing
!
interface fa0/2
 no sw
 ipv6 enable
 ipv6 address 1000:0:0:1::1/64
 no shutdown
```

```
 exit
!
interface fa0/3
 no sw
 ip address 10.0.3.1 255.255.255.0
 no shutdown
 exit
!
interface fa0/4
 no sw
 ip address 10.0.6.1 255.255.255.0
 no shutdown
 exit
!
interface fa0/1
 no sw
 ipv6 enable
 ipv6 address 1000:0:0:4::1/64
 no shutdown
 exit
!
router bgp 6
 no bgp default ipv4-unicast
 bgp router-id 1.0.0.0
 neighbor 1000:0:0:1::2 remote-as 1
 neighbor 10.0.3.2 remote-as 2
 neighbor 10.0.6.2 remote-as 3
 neighbor 1000:0:0:4::2 remote-as 4
 !
 address-family ipv4
  neighbor 10.0.3.2 activate
  neighbor 10.0.6.2 activate
  network 10.0.3.0 mask 255.255.255.0
  network 10.0.6.0 mask 255.255.255.0
  no auto-summary
  no synchronization
  exit-address-family
 !
 address-family ipv6
  neighbor 1000:0:0:1::2 activate
  neighbor 1000:0:0:4::2 activate
  network 1000:0:0:1::/64
  network 1000:0:0:4::/64
  exit-address-family
 !
 exit
 !
!
line con 0
 logging synchronous
 exit
line vty 0 4
 logging synchronous
 exit
!
end
```

Problem

Since I had already undergone the difficulty of having to manage the complex topology from part 1, I had fewer problems in part 2 than in part 1. However, as I did in part 1, I still had problems redistributing one protocol from another, especially in IPv6 routing protocols. For example, when I redistributed OSPF from BGP, I did not realize that I had to enter the address family mode . I thought redistribution would work even I wasn't in the address family mode. After going through a series of research, I eventually found out that the address family command allowed me to run both IPv4 and IPv6 networks on BGP.

In addition the command *Router (config-router-af)# neighbor [ipv6 address of the neighbor interface] activate* impeded me from continuing my lab. Since the commands for establishing a neighbor connection was already implemented in the *Router (config-router)#* mode, I did not realize that the address family mode required a second verification for establishing neighbor connections. Routing tables were completely empty in each device; the *show ip protocols* showed nothing but a couple of BGP routes, without any redistributions to other routes or to loopback interfaces.

Finally, the absence of the command *Router (config-router)# no bgp default ipv4-unicast* prevented me from assigning IPv6 address to different networks. Just as the command *Router (config)# ipv6 enable* needs to be issued to interfaces for configuring IPv6 addresses on them, this command is obligatory for configuring IPv6 addresses along with IPv4 addresses. As I forgot to issue this command, I began to think

that this problem was a layer issue, believing that the router that I configured BGP on did not allow configurations of IPv6 addresses.

Conclusion

Overall, I managed to implement both IPv4 and IPv6 in routing protocols that use them. Although my lack of research significantly retarded my progress with this lab, I was able to manage BGP and redistribute routing protocols more efficiently. After creating two different topologies, one with just BGP and another with IBGP that uses IPv4 and IPv6, my skills of managing BGP were further enhanced.