P6 - BGP2 (multi-homing)

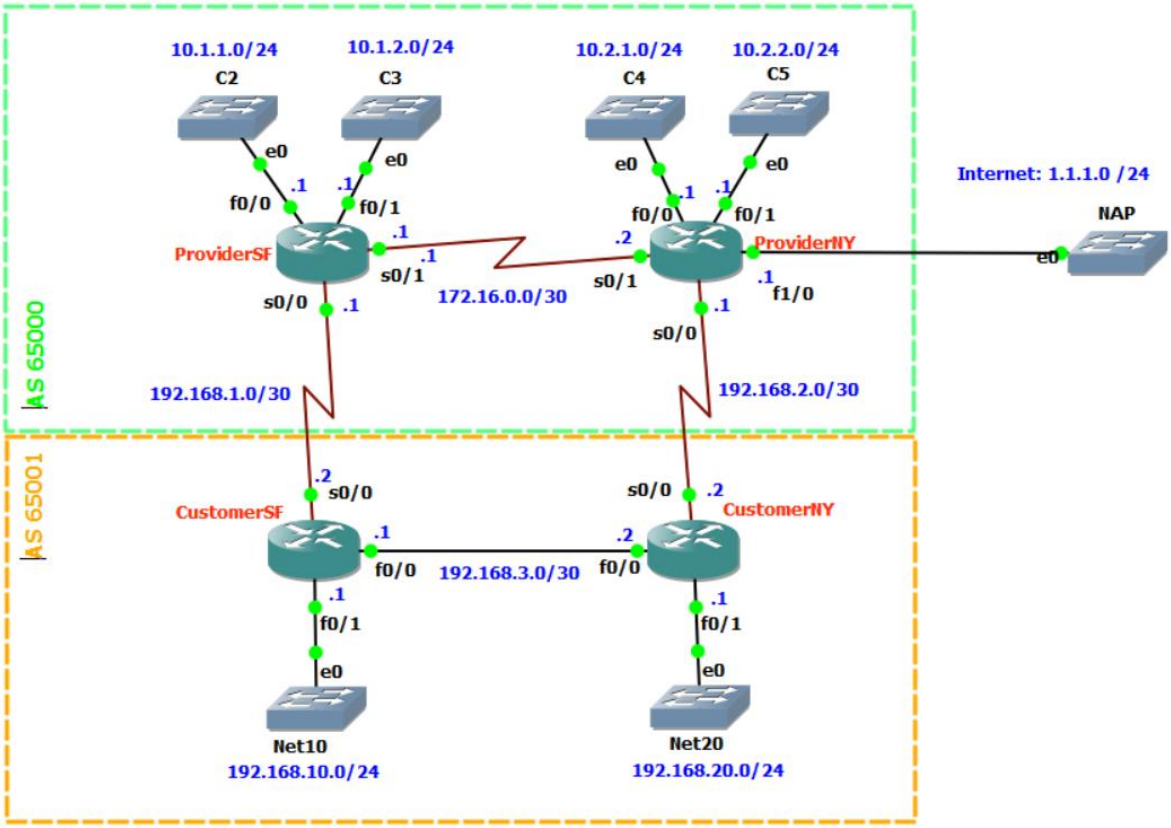
Luis Carlos Leaño Martin

Diego Gutiérrez Aldrete

ITESO

INTRODUCTION

In this practice, we worked on the configuration proposed by Pier Carlo Chiodi, which he read from the book “Internet Routing Architectures, 2nd Edition”. This was done to understand the use of confederations in BGP for multi-homing environments. In the diagram, we can see that there are two different autonomous systems (AS), which function as one since both have an internet exit (1.1.1.1.0) through a route. These divisions are known as confederations.



Confederations in BGP are a technique to improve the scalability and management of large autonomous networks by dividing a large autonomous system (AS) into smaller sub-AS that communicate with each other using eBGP but operate as a single AS from the outside. This simplifies the configuration and reduces the number of iBGP sessions required.

Advantages of BGP Confederations:

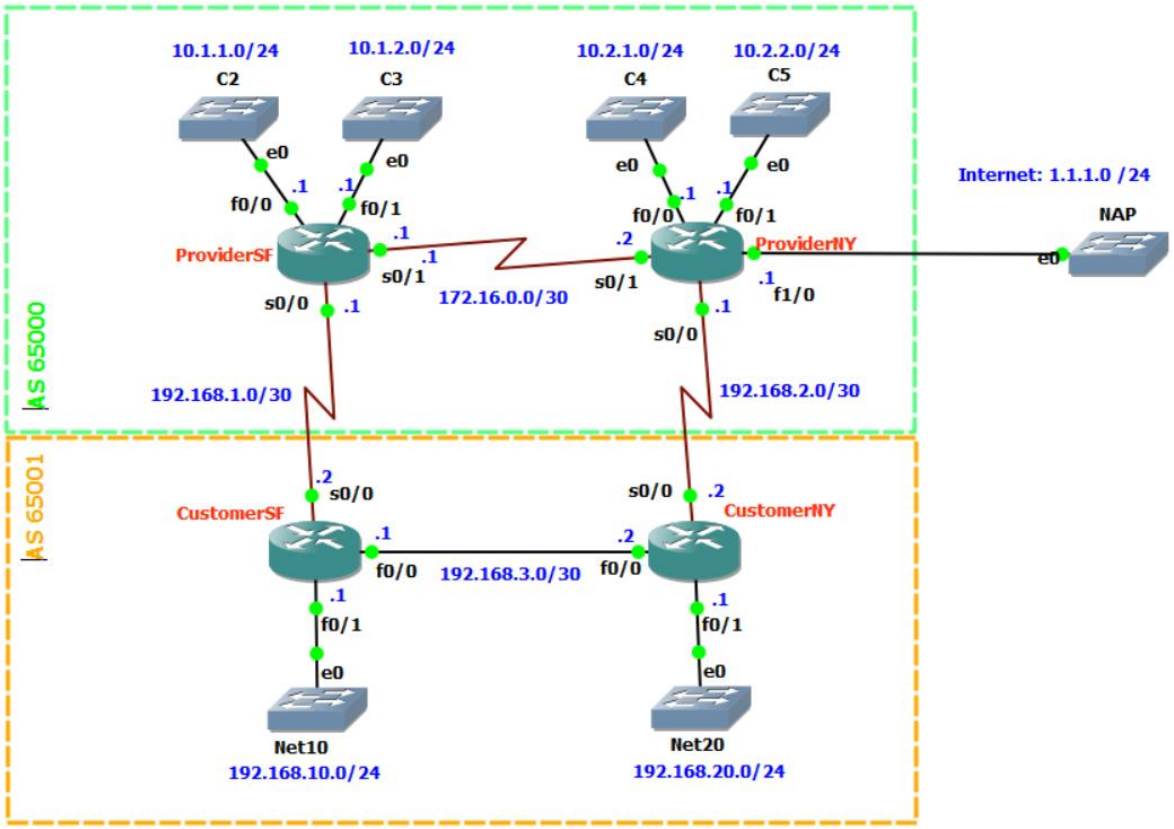
- **Improved Scalability**: Reduction in complexity and the number of iBGP sessions.
- **More Flexible Policy Management**: Each sub-AS can apply its own routing policies.

- **Reduced Convergence**: Shorter convergence times and better handling of sessions and routes.

Just as it has advantages, it also has its disadvantages, as its use generates greater complexity when configuring the network because everything needs to be well-structured to avoid routing errors and ensure correct communication between the sub-AS. In this case, to ensure that communication, we modified the next-hop attributes using “next-hop-self,” indicating that routers within the same autonomous system can use the IP of the router announcing the route as the “next hop.”

Additionally, route maps were configured, a tool that helped us define access policies by designated routes or deny them based on "matches" and "sets".

DEVELOPMENT



As we can see in this topology, we have two AS (Autonomous Systems), which are 65000 and 65001. We implemented the BGP routing protocol to establish communication, and, through **route-maps**, we declared the best routes to reach the subnets of C2 and C3 via **providerSF**, and C4 and C5 via **providerNY**, as well as the internet also via NY.

On the client side, we achieved this by configuring diferente local preference values on the prefixes. In the SF router, we set C2 and C3 with a local preference of 300, and any others are configured with a local preference of 200. In the NY router, we set C4 and C5 with a local preference of 300, and any other routes with 250. We do this using route-maps.

What we achieve with this is prioritizing these specific routes to optimize performance and efficiency, allowing us to adjust routing policies without making any physical changes to the network.

We managed to send the packets we want through the most efficient route, and the other routes remain efficient but are now handled by BGP preference.

What is the "next-hop-self" command configured for?

It is used to modify the next hop of the routes that a router advertises to its iBGP neighbors (neighbors within the same Autonomous System).

Without the next-hop-self command, the next hop could be an address outside the AS, which could cause problems if the internal routers do not have a route to that external next hop.

By using next-hop-self, you avoid the need for all internal routers to know how to get to the original next hop.

This simplifies configuration and reduces the complexity of the internal routing table.

Configuration

CutsomerSF:

```
router bgp 65001
 no synchronization
 bgp log-neighbor-changes
 network 192.168.10.0
 neighbor 192.168.1.1 remote-as 65000
 neighbor 192.168.1.1 route-map EBGP-ProvSF-IN in
 neighbor 192.168.1.1 route-map EBGP-ProvSF-OUT out
 neighbor 192.168.3.2 remote-as 65001
 neighbor 192.168.3.2 next-hop-self
 no auto-summary
!
ip forward-protocol nd
!
!
no ip http server
no ip http secure-server
!
!
ip prefix-list C2 seq 5 permit 10.1.1.0/24
!
ip prefix-list C3 seq 5 permit 10.1.2.0/24
!
ip prefix-list Net10 seq 5 permit 192.168.10.0/24
no cdp log mismatch duplex
!
route-map EBGP-ProvSF-IN permit 10
 match ip address prefix-list C2 C3
 set local-preference 300
!
route-map EBGP-ProvSF-IN permit 20
 set local-preference 200
!
route-map EBGP-ProvSF-OUT permit 10
 match ip address prefix-list Net10
 set metric 200
!
route-map EBGP-ProvSF-OUT permit 20
 set metric 300
```

CustomerNY:

```
router bgp 65001
 no synchronization
 bgp log-neighbor-changes
 network 192.168.20.0
 neighbor 192.168.2.1 remote-as 65000
 neighbor 192.168.2.1 route-map EBGP-ProvNY-IN in
 neighbor 192.168.2.1 route-map EBGP-ProvNY-OUT out
 neighbor 192.168.3.1 remote-as 65001
 neighbor 192.168.3.1 next-hop-self
 no auto-summary
!
ip forward-protocol nd
!
!
no ip http server
no ip http secure-server
!
!
ip prefix-list C4 seq 5 permit 10.2.1.0/24
!
ip prefix-list C5 seq 5 permit 10.2.2.0/24
!
ip prefix-list Net20 seq 5 permit 192.168.20.0/24
no cdp log mismatch duplex
!
route-map EBGP-ProvNY-IN permit 10
 match ip address prefix-list C4 C5
 set local-preference 300
!
route-map EBGP-ProvNY-IN permit 20
 set local-preference 250
!
route-map EBGP-ProvNY-OUT permit 10
 match ip address prefix-list Net20
 set metric 200
!
route-map EBGP-ProvNY-OUT permit 20
 set metric 250
```

ProvideerSF:

```
router bgp 65000
 no synchronization
 bgp log-neighbor-changes
 network 10.1.1.0 mask 255.255.255.0
 network 10.1.2.0 mask 255.255.255.0
 neighbor 172.16.0.2 remote-as 65000
 neighbor 172.16.0.2 next-hop-self
 neighbor 192.168.1.2 remote-as 65001
 no auto-summary
```

ProvideerNY:

```
router bgp 65000
 no synchronization
 bgp log-neighbor-changes
 network 1.1.1.0 mask 255.255.255.0
 network 10.2.1.0 mask 255.255.255.0
 network 10.2.2.0 mask 255.255.255.0
 neighbor 172.16.0.1 remote-as 65000
 neighbor 172.16.0.1 next-hop-self
 neighbor 192.168.2.2 remote-as 65001
 no auto-summary
```

EVIDENCE

CutsomerSF:

BGP routing table

```
CustomerSF#show ip route bgp
     1.0.0.0/24 is subnetted, 1 subnets
B       1.1.1.0 [200/0] via 192.168.3.2, 00:02:16
B    192.168.20.0/24 [200/0] via 192.168.3.2, 00:02:16
     10.0.0.0/24 is subnetted, 4 subnets
B       10.2.1.0 [200/0] via 192.168.3.2, 00:02:16
B       10.1.2.0 [20/0] via 192.168.1.1, 00:02:16
B       10.2.2.0 [200/0] via 192.168.3.2, 00:02:16
B       10.1.1.0 [20/0] via 192.168.1.1, 00:02:16
CustomerSF#
```

Ping Internet 1.1.1.0:

```
CustomerSF#ping 1.1.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
CustomerSF#
```

Ping Internet 1.1.1.0 source 192.168.10.1:

```
CustomerSF#ping 1.1.1.1 source 192.168.10.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.1, timeout is 2 seconds:
Packet sent with a source address of 192.168.10.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/46/56 ms
CustomerSF#
```

This happens because in the BGP configuration the only network we advertised was Net10 (192.168.10.0) that is why we can reach the Internet and with the route maps we tell it to go through NY and not through SF as we see in the trace.

```
CustomerSF#trace 1.1.1.1 source 192.168.10.1

Type escape sequence to abort.
Tracing the route to 1.1.1.1

  1 192.168.3.2 16 msec 32 msec 40 msec
  2 192.168.2.1 36 msec 56 msec 32 msec
CustomerSF#
```

CustomerNY:

BGP routing table

```
CustomerNY#show ip route bgp
     1.0.0.0/24 is subnetted, 1 subnets
B       1.1.1.0 [20/0] via 192.168.2.1, 00:03:55
B    192.168.10.0/24 [200/0] via 192.168.3.1, 00:03:51
     10.0.0.0/24 is subnetted, 4 subnets
B       10.2.1.0 [20/0] via 192.168.2.1, 00:03:55
B       10.1.2.0 [200/0] via 192.168.3.1, 00:03:51
B       10.2.2.0 [20/0] via 192.168.2.1, 00:03:55
B       10.1.1.0 [200/0] via 192.168.3.1, 00:03:51
CustomerNY#
```

ProvideerSF:

BGP routing table

```
ProviderSF#sh ip route bgp
     1.0.0.0/24 is subnetted, 1 subnets
B       1.1.1.0 [200/0] via 172.16.0.2, 00:04:44
B    192.168.10.0/24 [20/200] via 192.168.1.2, 00:04:44
B    192.168.20.0/24 [200/200] via 172.16.0.2, 00:04:44
     10.0.0.0/24 is subnetted, 4 subnets
B       10.2.1.0 [200/0] via 172.16.0.2, 00:04:44
B       10.2.2.0 [200/0] via 172.16.0.2, 00:04:44
ProviderSF#
```

```
ProviderSF#sh ip route bgp | inc 192
B    192.168.10.0/24 [20/200] via 192.168.1.2, 00:07:36
B    192.168.20.0/24 [200/200] via 172.16.0.2, 00:07:36
ProviderSF#
```

ProvideerNY:

BGP routing table

```
ProviderNY#sh ip route bgp
B    192.168.10.0/24 [200/200] via 172.16.0.1, 00:04:42
B    192.168.20.0/24 [20/200] via 192.168.2.2, 00:04:46
     10.0.0.0/24 is subnetted, 4 subnets
B       10.1.2.0 [200/0] via 172.16.0.1, 00:04:42
B       10.1.1.0 [200/0] via 172.16.0.1, 00:04:42
ProviderNY#
```

```
ProviderNY#sh ip route bgp | inc 192
B    192.168.10.0/24 [200/200] via 172.16.0.1, 00:08:03
B    192.168.20.0/24 [20/200] via 192.168.2.2, 00:08:07
ProviderNY#
```

COCLUSION

Luis Carlos Leaño Martín:

In this BGP (Multi-Homing) practice, we learned how to configure a scenario involving two providers. What I found most valuable was learning about the **"next-hop self"** command, which allows us to modify the next-hop attribute within our own AS (Autonomous System). Without this command, the next-hop could be an address outside our AS, requiring static routes or additional configurations to determine the correct path. By using "next-hop self," we ensured proper internal routing without unnecessary complexity.

Additionally, we implemented **route-maps** to manage routes more efficiently by assigning a higher local preference to specific routes. This enabled us to prioritize certain paths, resulting in a more efficient and easier-to-manage configuration. Overall, this practice enhanced our ability to manipulate BGP routing policies for optimal network performance.

Diego Gutiérrez Aldrete:

Well, we had already used route-maps, so we already knew that its operation is to modify the routes allowing specific traffic and thus modifying the preferences that may be in the routers.

What was something new and interesting was the use of a topology with confederations and obviously the use of BGP with Multi-homing. I found it very interesting that with the specifications we put on it and with the use of "next-hop-self" some routers could not reach the INTERNET by the way we configured them to follow, but certain networks, when announced of the proper routes, could do so.

SOURCES

- Cisco Systems. (n.d.). *Border Gateway Protocol (BGP) Case Studies*. Cisco. Retrieved November 5, 2024, from https://www.cisco.com/c/es_mx/support/docs/ip/border-gateway-protocol-bgp/26634-bgp-toc.html
- Cisco. (n.d.). Route-map definition. Cisco Community. Retrieved November 2, 2024, from https://community.cisco.com/t5/routing/route-map-definition/td-p/1864524
- Juniper Networks. (n.d.). BGP confederations for scaling. Juniper Networks Documentation. Retrieved November 2, 2024, from https://www.juniper.net/documentation/mx/es/software/junos/bgp/topics/topic-map/bgp-confederations-for-scaling.html
- NetworkLessons. (n.d.). BGP confederation explained. NetworkLessons. Retrieved November 2, 2024, from https://networklessons.com/bgp/bgp-confederation-explained
- Pierky. (n.d.). GNS3 lab: BGP multihoming to a single provider with partial routing. Pierky's Blog. Retrieved November 2, 2024, from https://blog.pierky.com/gns3-lab-bgp-multihoming-to-a-single-provider-with-partial-routing/