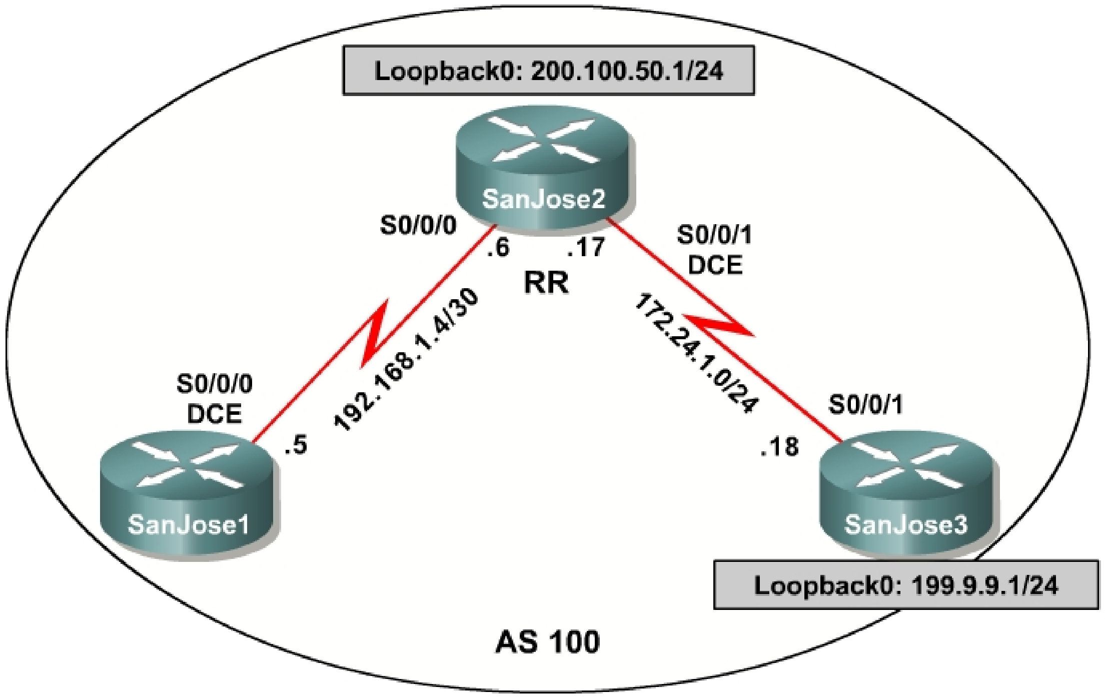
Lab 6-4 BGP Route Reflectors and Route Filters

Topology Diagram



Learning Objective

In this lab, you will configure IBGP routers to use a route reflector and a simple route filter.

Scenario

The International Travel Agency maintains a full mesh IBGP network that has quickly scaled beyond 100 routers. The company wants to implement route reflectors to work around the full mesh IBGP requirement. Configure a small cluster and observe how BGP operates in this configuration. Use IP prefix filters to control the updates between IBGP peers.

Step 1: Configure RIPv2

Build and configure the network according to the diagram, and use RIPv2 as the IGP. Do not configure the 199.9.9.0 network under the RIP process. Use **ping** to test connectivity among all interfaces. Each router should have a complete routing table.

```
SanJose1(config)#router rip
SanJose1(config-router)#version 2
SanJose1(config-router)#no auto-summary
SanJose1(config-router)#network 192.168.1.0
```

```
SanJose2(config)#router rip
SanJose2(config-router)#version 2
SanJose2(config-router)#no auto-summary
SanJose2(config-router)#network 172.24.0.0
SanJose2(config-router)#network 192.168.1.0
SanJose2(config-router)#network 200.100.50.0
```

```
SanJose3(config)#router rip
SanJose3(config-router)#version 2
SanJose3(config-router)#no auto-summary
SanJose3(config-router)#network 172.24.0.0
```

Step 2: IBGP Peers and Route Reflectors

In this lab, we will configure a route reflector. By default, a router that receives an EBGP route will advertise it to its IBGP peers. However, if it receives it through IBGP, it will not advertise it to its IBGP peers. This is a loop prevention mechanism. However, because of this behavior, the only way for all IBGP routers to receive a route once it is originated into the AS is to have a full mesh of IBGP peers. This can get messy with a large number of peers. To get around this limitation of IBGP, we can use a route reflector. A route reflector allows a topology to get around the IBGP limitation of having to have a full mesh. To do this, a route reflector specifies its neighbors as route reflector clients. When a route reflector receives an update from a route reflector client, it can pass it on to its other clients. This greatly simplifies configuration because only the route reflector needs to know all the other peers. The clients don't even know that they are clients. To them, it is just a normal IBGP peering relationship. You can even set up multiple route reflectors in a more advanced configuration for redundancy.

Configure the IBGP peers for BGP. Later, you will configure SanJose2 as the route reflector. However, first configure it to peer with both of the other routers:

```
SanJose2(config)#router bgp 100
SanJose2(config-router)#neighbor 192.168.1.5 remote-as 100
SanJose2(config-router)#neighbor 172.24.1.18 remote-as 100
SanJose2(config-router)#network 200.100.50.0
SanJose2(config-router)#network 172.24.1.0 mask 255.255.255.0
SanJose2(config-router)#network 192.168.1.4 mask 255.255.255.252
```

After SanJose2 is configured, configure the other two routers as route reflector clients. Remember that to set up clients simply configure peering between the client and the server. IBGP does not need to be configured to a full mesh.

Issue the following commands on SanJose1:

```
SanJose1(config)#router bgp 100
SanJose1(config-router)#neighbor 192.168.1.6 remote-as 100
```

Issue the following commands on SanJose3:

```
SanJose3(config)#router bgp 100
SanJose3(config-router)#neighbor 172.24.1.17 remote-as 100
```

Verify that SanJose2 has established a peering relationship with both SanJose1 and SanJose3. Troubleshoot as necessary.

SanJose1 and SanJose3 should not have established a connection. Why?

SanJose3 was not configured with the appropriate BGP **neighbor** command. As a route reflector client, SanJose1 does not need to reach an Established state with SanJose3.

Step 3: Inject an External Route into BGP

To observe the full effect of using a route reflector, configure SanJose3 to inject external routing information into BGP:

```
SanJose3(config)#int lo0
SanJose3(config-if)#ip address 199.9.9.1 255.255.255.0
SanJose3(config)#router bgp 100
SanJose3(config-router)#network 199.9.9.0
```

This configuration forces SanJose3 to inject the external route 199.9.9.0 into BGP. Use the **show ip route** command to check if SanJose2 has picked up this route through BGP. SanJose2 should have a route to 199.9.9.0.

Is the next hop for this route 172.24.1.18?

Verify that you can ping 199.9.9.1 from SanJose3. If not, troubleshoot.

Now check the routing table of SanJose1. There should not be a route to 199.9.9.0. Why?

Remember that SanJose1 is not configured to peer with SanJose3. To eliminate the need for a full IBGP mesh, SanJose2 must be configured as a route reflector. Issue the following commands on SanJose2:

```
SanJose2(config)#router bgp 100
SanJose2(config-router)#neighbor 192.168.1.5 route-reflector-client
SanJose2(config-router)#neighbor 172.24.1.18 route-reflector-client
```

Verify that an IBGP cluster was successfully created by issuing the **show ip protocols** command on SanJose2. The output of this command should indicate that SanJose2 is a route reflector.

How many clients does SanJose2 have?

Issue the **show ip protocols** command on SanJose1. The output of this command does not include information about route reflectors. Remember that SanJose1 is a client and not a route reflector server, so it is unaware of route reflection.

Finally, verify that route reflection is working by checking the routing table on SanJose1. SanJose1 will have a route to network 199.9.9.0.

Is 172.24.1.18 the IP address of the next hop of this route on the SanJose1 table?

Notice that SanJose1 is not directly connected to the IP network for the next hop. Why?
Hint: From which router did SanJose1 learn the route?

Ping 199.9.9.1 from SanJose1. This ping should be successful.

Notice that SanJose1 pings 199.9.9.1 even though the next-hop address is not on a directly connected network. For example, the next-hop address could be 172.24.1.18.

Step 4: Inject a Summary Address into BGP

For the purpose of this lab, configure SanJose3 to inject a summary address into BGP:

```
SanJose3(config)#router bgp 100
SanJose3(config-router)#aggregate-address 199.0.0.0 255.0.0.0
```

BGP should now send the supernet route 199.0.0.0/8 to SanJose2 with the attribute ATOMIC_AGGREGATE set.

On SanJose2, issue the following command:

```
SanJose2#show ip bgp 199.0.0.0
BGP routing table entry for 199.0.0.0/8, version 6
Paths: (1 available, best #1)
Bestpath transition flag: 0x208
               Advertised to non peer-group peers:
               192.168.1.5
  Local, (aggregated by 100 172.24.1.18), (Received from a RR-client)
    172.24.1.18 from 172.24.1.18 (172.24.1.18)
      Origin IGP, localpref 100, valid, internal, atomic-aggregate, best,
      ref 2
```

According to the output of this command, what address aggregated this route?

What indicates that route reflection is involved in this process?

Is there an indication that the ATOMIC_AGGREGATE attribute has been set?

SanJose2 should, in turn, reflect this route to SanJose1. Check both the routing table and BGP table on SanJose1 to be sure. Both the route to 199.9.9.0 and the supernet route 199.0.0.0 should be installed in the SanJose1 routing table and the BGP table.

The International Travel Agency has decided to filter specific routes to the 199.0.0.0/8 address space. Configure a route filter to prevent SanJose2 from sending the 199.9.9.0/24 route to its other clients, in this case to SanJose1.

Issue the following commands on SanJose2:

```
SanJose2(config)#ip prefix-list supernetonly permit 199.0.0.0/8
SanJose2(config)#ip prefix-list supernetonly permit 172.24.1.0/24
SanJose2(config)#ip prefix-list supernetonly permit 200.100.50.0/24
SanJose2(config)#router bgp 100
SanJose2(config-router)#neighbor 192.168.1.5 prefix-list supernetonly out
```

Return to SanJose1, issue the **clear ip bgp *** command, and verify that the prefix list has done its job by issuing a **show ip bgp** command. Troubleshoot as necessary.

Unlike before, where routes to 199.9.9.0 and 199.0.0.0 were present, now only one route to 199.0.0.0 in the routing and BGP tables should be seen. Troubleshoot as necessary.

Appendix A: TCL Verification

```
SanJose1#tclsh
SanJose1(tcl)#
SanJose1(tcl)#foreach address {
+>200.100.50.1
+>199.9.9.1
+>192.168.1.5
+>192.168.1.6
+>172.24.1.17
+>172.24.1.18
+>} {
+>ping $address }

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.50.1, 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 199.9.9.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/60 ms
Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 192.168.1.5, 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 192.168.1.6, 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 172.24.1.17, 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 172.24.1.18, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
SanJose1(tcl)#tclquit

SanJose2#tclsh
SanJose2(tcl)#
SanJose2(tcl)#foreach address {
+>200.100.50.1
+>199.9.9.1
+>192.168.1.5
+>192.168.1.6
+>172.24.1.17
+>172.24.1.18
+>} {
+>ping $address }

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.50.1, 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 199.9.9.1, 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 192.168.1.5, 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 192.168.1.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.24.1.17, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/58/64 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.24.1.18, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
SanJose2(tcl)#tclquit

SanJose3#tclsh
SanJose3(tcl)#
SanJose3(tcl)#foreach address {
+>200.100.50.1
+>199.9.9.1
+>192.168.1.5
+>192.168.1.6
+>172.24.1.17
+>172.24.1.18
+>} {
+>ping $address }

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.50.1, 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 199.9.9.1, 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 192.168.1.5, 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 192.168.1.6, 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 172.24.1.17, 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 172.24.1.18, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/63/92 ms
SanJose3(tcl)#tclquit
```

Final Configurations

```
SanJose1#show run
!
hostname SanJose1
!
interface Serial0/0/0
 ip address 192.168.1.5 255.255.255.252
 clock rate 64000
 no shutdown
!
router rip
 version 2
 network 192.168.1.0
 no auto-summary
!
router bgp 100
 no synchronization
 neighbor 192.168.1.6 remote-as 100
 no auto-summary
!
end

SanJose2#show run
!
hostname SanJose2
!
interface Loopback0
 ip address 200.100.50.1 255.255.255.0
!
interface Serial0/0/0
 ip address 192.168.1.6 255.255.255.252
 no shutdown
!
interface Serial0/0/1
 ip address 172.24.1.17 255.255.255.0
 clock rate 64000
 no shutdown
!
router rip
 version 2
 network 172.24.0.0
 network 192.168.1.0
 network 200.100.50.0
 no auto-summary
!
router bgp 100
 no synchronization
 network 172.24.1.0 mask 255.255.255.0
 network 192.168.1.4 mask 255.255.255.252
 network 200.100.50.0
 neighbor 172.24.1.18 remote-as 100
 neighbor 172.24.1.18 route-reflector-client
 neighbor 192.168.1.5 remote-as 100
 neighbor 192.168.1.5 route-reflector-client
 neighbor 192.168.1.5 prefix-list supernetonly out
 no auto-summary
!
ip prefix-list supernetonly seq 5 permit 199.0.0.0/8
ip prefix-list supernetonly seq 10 permit 172.24.1.0/24
ip prefix-list supernetonly seq 15 permit 200.100.50.0/24
!
end

SanJose3#show run
```

```
!
hostname SanJose3
!
interface Loopback0
 ip address 199.9.9.1 255.255.255.0
!
interface Serial0/0/1
 ip address 172.24.1.18 255.255.255.0
 no shutdown
!
router rip
 version 2
 network 172.24.0.0
 no auto-summary
!
router bgp 100
 no synchronization
 network 199.9.9.0
 aggregate-address 199.0.0.0 255.0.0.0
 neighbor 172.24.1.17 remote-as 100
 no auto-summary
end
```