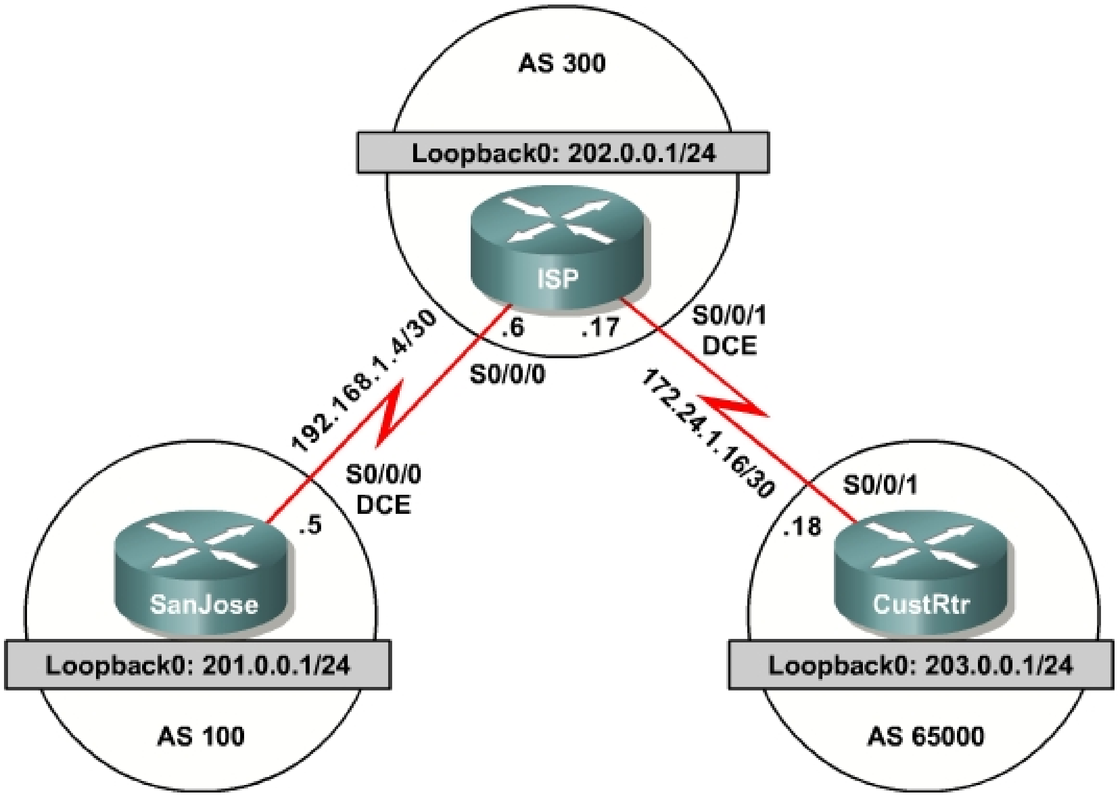
Lab 6-2 Using the AS_PATH Attribute

Topology Diagram



Learning Objective

In this lab, you will use BGP commands to prevent private AS numbers from being advertised to the outside world. You will also use the AS_PATH attribute to filter BGP routes based on their source AS numbers.

Scenario

The International Travel Agency's ISP has been assigned an AS number of 300. This provider uses BGP to exchange routing information with several customer networks. Each customer network is assigned an AS number from the private range, such as AS 65000. Configure ISP to remove the private AS numbers within the AS_Path information from CustRtr. In addition, the ISP would like to prevent its customer networks from receiving route information from International Travel Agency's AS 100. Use the AS_PATH attribute to implement this policy.

Step 1: IP Addressing

Build and configure the network according to the diagram, but do not configure a routing protocol.

Use **ping** to test the connectivity between the directly connected routers. Note that SanJose cannot reach the customer network for CustRtr. It cannot reach it by the IP address in the link leading to CustRtr nor the loopback interface 202.0.0.1/24.

Note: SanJose will not be able to reach the customer network for ISP, CustRtr. It will not be able to reach it by the IP address in the link leading to the CustRtr, nor the loopback interface, 202.0.0.1/24.

Step 2: Configure BGP

Configure BGP for normal operation. Enter the appropriate BGP commands on each router so that they identify their BGP neighbors and advertise their loopback networks:

```
SanJose(config)#router bgp 100
SanJose(config-router)#neighbor 192.168.1.6 remote-as 300
SanJose(config-router)#network 201.0.0.0

ISP(config)#router bgp 300
ISP(config-router)#neighbor 192.168.1.5 remote-as 100
ISP(config-router)#neighbor 172.24.1.18 remote-as 65000
ISP(config-router)#network 202.0.0.0

CustRtr(config)#router bgp 65000
CustRtr(config-router)#neighbor 172.24.1.17 remote-as 300
CustRtr(config-router)#network 203.0.0.0
```

Verify that these routers have established the appropriate neighbor relationships by issuing the **show ip bgp neighbors** command on each router.

Step 3: Remove the Private AS

Check SanJose's routing table by using the **show ip route** command. SanJose should have a route to both 202.0.0.0 and 203.0.0.0. Troubleshoot, if necessary.

Ping the 203.0.0.1 address from SanJose. Why does this fail?

Ping again, this time as an extended **ping**, sourcing from the Loopback 0 interface as follows:

```
SanJose#ping
Protocol [ip]:
Target IP address: 203.0.0.1
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
```

```
Source address or interface: 201.0.0.1
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 64/64/68 ms
```

Check the BGP table from SanJose by using the **show ip bgp** command. Note the AS path for the 203.0.0.0 network. The AS 65000 should be listed in the path to 203.0.0.0. Why is this a problem?

```
BGP table version is 4, local router ID is 201.0.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i –
internal   Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 201.0.0.0        0.0.0.0                  0         32768 i
*> 202.0.0.0        192.168.1.6              0             0 300 i
*> 203.0.0.0        192.168.1.6                            0 300 65000 i
```

Configure ISP to strip the private AS numbers from BGP routes exchanged with SanJose. Use the following commands:

```
ISP(config)#router bgp 300
ISP(config-router)#neighbor 192.168.1.5 remove-private-as
```

After issuing these commands, use the **clear ip bgp *** command on SanJose to reestablish the BGP relationship between the three routers.

Wait several seconds, and then return to SanJose to check its routing table.

Does SanJose still have a route to 203.0.0.0?

SanJose should be able to ping 203.0.0.0.

Now check the BGP table on SanJose. The AS_PATH to the 203.0.0.0 network should be AS 300.

```
BGP table version is 8, local router ID is 201.0.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i –
internal   Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 201.0.0.0        0.0.0.0                  0         32768 i
*> 202.0.0.0        192.168.1.6              0             0 300 i
*> 203.0.0.0        192.168.1.6                            0 300 i
```

Step 4: Use the AS_PATH Attribute to Filter Routes

As a final configuration, use the AS_PATH attribute to filter routes based on their origin. In a complex environment, this attribute can be used to enforce routing policy. In this case, the provider router, ISP, must be configured so that it does not propagate routes that originate from AS 100 to the customer router, CustRtr.

First, configure a special kind of access list to match BGP routes with an AS_PATH attribute that both begins and ends with the number 100. Enter the following commands on ISP:

```
ISP(config)#ip as-path access-list 1 deny ^100$
ISP(config)#ip as-path access-list 1 permit .*
```

AS-path access lists are read like regular access lists, in that they are read through in order and have an implicit deny at the end. Rather than matching an address in each statement, like a conventional access-list, they match on something called regular expressions. Regular expressions are a way of matching text patterns, and have many uses. In this case, we will using them in the AS-path access list to match text patterns in AS-paths.

The first command above uses the **^** character to indicate that the AS_PATH must begin with the given number 100. The **$** character indicates that the AS_PATH attribute must also end with 100. Essentially, this statement matches only paths that are sourced from AS 100. Other paths, which might include AS 100 along the way, will not match this list.

In the second statement, the **.** character is a wildcard, and the ***** symbol stands for a repetition of the wildcard. Together, **.*** matches any value of the AS_PATH attribute, which in effect permits any update that has not been denied by the previous **access-list** statement.

For more details on configuring regular expressions on Cisco routers, use the following link:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122cgcr/ftersv_c/ftsappx/tcfaapre.htm

Now that the access list has been configured, apply it as follows:

```
ISP(config)#router bgp 300
ISP(config-router)#neighbor 172.24.1.18 filter-list 1 out
```

The **out** keyword specifies that the list is applied to routing information sent to this neighbor.

Use the **clear ip bgp *** command to reset the routing information. Wait several seconds, and then check the routing table for ISP. The route to 201.0.0.0 should be in the routing table.

Check the routing table for CustRtr. It should not have a route to 201.0.0.0 in its routing table.

Return to ISP and verify that the filter is working as intended. Issue the command **show ip bgp regexp ^100$**.

The output of this command shows all matches for the regular expressions that were used in the access list. The path to 201.0.0.0 matches the access list and is filtered from updates to CustRtr.

```
ISP#show ip bgp regexp ^100$
BGP table version is 4, local router ID is 202.0.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i –
internal   Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 201.0.0.0        192.168.1.5              0             0 100 i
```

Appendix A: TCL Output

```
tclsh

foreach address {
201.0.0.1
202.0.0.1
203.0.0.1
192.168.1.5
192.168.1.6
172.24.1.17
172.24.1.18
} {
ping $address }

SanJose#tclsh
SanJose(tcl)#
SanJose(tcl)#foreach address {
+>201.0.0.1
+>202.0.0.1
+>203.0.0.1
+>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 201.0.0.1, 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 202.0.0.1, 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 203.0.0.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
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 = 28/28/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.24.1.17, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.24.1.18, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
SanJose(tcl)#**tclquit**


ISP#**tclsh**
ISP(tcl)#
ISP(tcl)#foreach address {
+>201.0.0.1
+>202.0.0.1
+>203.0.0.1
+>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 201.0.0.1, 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 202.0.0.1, 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 203.0.0.1, 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 = 28/28/32 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 = 56/57/64 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 = 56/58/68 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 = 28/28/32 ms
ISP(tcl)#**tclquit**

CustRtr#**tclsh**
CustRtr(tcl)#
CustRtr(tcl)#foreach address {
+>201.0.0.1
+>202.0.0.1
+>203.0.0.1
+>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 201.0.0.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 202.0.0.1, 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 203.0.0.1, 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 0 percent (0/5)
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.1.6, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
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 = 56/57/64 ms
CustRtr(tcl)#tclquit
```

Final Configurations

```
SanJose#show run
!
hostname SanJose
!
interface Loopback0
 ip address 201.0.0.1 255.255.255.0
!
!
interface Serial0/0/0
 ip address 192.168.1.5 255.255.255.252
 clock rate 64000
 no shutdown
!
router bgp 100
 no synchronization
 network 201.0.0.0
 neighbor 192.168.1.6 remote-as 300
 no auto-summary
!
end

ISP#show run
!
hostname ISP
```

```
!
interface Loopback0
 ip address 202.0.0.1 255.255.255.0
!
interface Serial0/0/0
 ip address 192.168.1.6 255.255.255.252
 no shutdown
!
interface Serial0/0/1
 ip address 172.24.1.17 255.255.255.252
 clock rate 64000
 no shutdown
!
router bgp 300
 no synchronization
 network 202.0.0.0
 neighbor 172.24.1.18 remote-as 65000
 neighbor 172.24.1.18 filter-list 1 out
 neighbor 192.168.1.5 remote-as 100
 neighbor 192.168.1.5 remove-private-as
 no auto-summary
!
ip as-path access-list 1 deny ^100$
ip as-path access-list 1 permit .*
!
end

CustRtr#show run
!
hostname CustRtr
!
interface Loopback0
 ip address 203.0.0.1 255.255.255.0
!
interface Serial0/0/1
 ip address 172.24.1.18 255.255.255.252
 no shutdown
!
router bgp 65000
 no synchronization
 network 203.0.0.0
 neighbor 172.24.1.17 remote-as 300
 no auto-summary
!
end
```