Peering & IXP Lab (Part 4: Communities)

The purpose of this part of the lab is introduce BGP Communities into our lab. We are doing this for ease of management of the various prefixes we are carrying in our iBGP.

Lab Topology

The lab topology is unchanged from the topology used in the previous section.

BGP Communities

We will be updating our BGP configurations to use BGP Communities. Following the presentation, we specifically want to tag prefixes we learn from our private peers, our IXP peers, and our own prefixes we introduce into our iBGP. This way we can manage what we announce to all external networks simply by applying community filters, removing the need to maintain prefix-lists, and therefore simplifying the management of the network configuration.

Community Policy

The communities we are going to adopt are as follows:

Prefix Type	Community	Description
Our aggregate	*AS*:1000	Our assigned address block(s)
Subnets of our aggregate	*AS*:1001	Any subnets we carve out of our address block(s)
Customer independent addresses	*AS*:1005	Any address space customers are assigned independently of what they receive from us
Private Peer addresses	*AS*:1100	Any addresses our private peers send to us
IXP Peer addresses	*AS*:1200	Any addresses our IXP peers send to us

Each group will now set up these community definitions on all three of their routers. Here is an example for AS102:

```
ip community-list 1 permit 102:1000
ip community-list 2 permit 102:1001
ip community-list 3 permit 102:1005
ip community-list 4 permit 102:1100
ip community-list 5 permit 102:1200
```

We also need to tell Cisco IOS to use the industry format for BGP communities (as described in RFC1998) rather than the specification standard (as described in RFC1997). The latter represents communities as a 32-bit integer; the former represents communities as two 16-bit integers separate by a colon.

```
ip bgp-community new-format
```

Setting Community on our prefixes

We introduced our aggregate for our AS on the Core router using the BGP network statement. We are now going to augment this network statement so that the community is set on the prefix we introduce. Here is an example:

```
route-map set-aggregate-community permit 10
 set community 103:1000
!
router bgp 103
 address-family ipv4
  network 100.68.30.0 mask 255.255.255.0 route-map set-aggregate-community
 address-family ipv6
  network 2001:DB8:30::/48 route-map set-aggregate-community
!
```

Once this is done, check the BGP table. Run the:

```
show ip bgp community
```

and

```
show bgp ipv6 unicast community
```

commands. What do you see now?

All being well, you should see your IPv4 and IPv6 aggregates displayed in the output. This command shows you all prefixes which have a BGP community set on them. Then show the actual BGP table entry - you will now see an extra line in the output, indicating which community has been set. It should be in the format *AS*:1000.

Setting Community prefix learned from the private peer

We now set the community on the prefix we learn from our private peer. As with the previous example, we look for the prefix they are sending us, and tag it with the appropriate community. Here is an example for AS104 - note that we are adding to the existing configuration (which includes the *private-peer-in* route-map and the inbound prefix-list for the private peer):

```
route-map private-peer-in permit 10
 set community 104:1100
!
router bgp 104
 address-family ipv4
  neighbor 100.68.30.25 route-map private-peer-in in
!
```

```
 address-family ipv6
  neighbor 2001:DB8:30:12:: route-map private-peer-in in
!
```

Once the route-map and the BGP neighbour configuration has been updated, remember to do the inbound route-refresh. Cisco IOS does not do route-refreshes automatically.

Check the communities listed in the BGP tables as you did earlier. Do you now see the private peer prefix with a community set on it?

Setting Community prefix learned from the IXP peers

We now set the community on the prefix we learn from our IXP peers via the Route Server (or bi-laterals peers if you have chosen not to peer with the Route Server). As with the previous example, we look for the prefix they are sending us, and tag it with the appropriate community. Here is an example for AS105 - note again that we are adding to the existing configuration (which includes the *ixp-peer-in* route-map and the inbound prefix-list for the IXP peer):

```
route-map IXP-peer-in permit 10
 set community 105:1200
!
router bgp 105
 address-family ipv4
  neighbor 100.127.1.254 route-map IXP-peer-in in
!
 address-family ipv6
  neighbor 2001:DB8:FFFF:1::FE route-map IXP-peer-in in
!
```

Once the route-map and the BGP neighbour configuration has been updated, remember to do the inbound route-refresh.

Check the communities listed in the BGP tables as you did earlier. Do you now see the IXP peers' prefixes with a community set on them?

Outbound BGP Announcements Filtered by Community

The final step in this exercise is to update our outbound BGP filters to use BGP communities rather than the prefix-lists we set up earlier. Our inbound filters will remain using prefix-lists, as we want to control what our neighbours send to us. However, for outbound announcements, prefix-lists do not scale, as every time we need to announce a new prefix, we have to update the filters - which gets very tedious to maintain as the network grows larger.

The current BGP configuration should be looking something like this example shown for AS102's border router:

```
ip community-list 1 permit 102:1000
ip community-list 2 permit 102:1001
ip community-list 3 permit 102:1005
ip community-list 4 permit 102:1100
ip community-list 5 permit 102:1200
!
ip prefix-list AS102-block permit 100.68.20.0/24
ipv6 prefix-list AS102-v6block permit 2001:DB8:20::/48
!
ip prefix-list default-route permit 0.0.0.0/0
ipv6 prefix-list default-v6route permit ::/0
!
router bgp 102
 address-family ipv4
  neighbor 100.121.1.5 remote-as 121
  neighbor 100.121.1.5 description eBGP with TRANSIT 1
  neighbor 100.121.1.5 password cisco
  neighbor 100.121.1.5 prefix-list AS102-block out
  neighbor 100.121.1.5 prefix-list default-route in
!
 address-family ipv6
  neighbor 2001:18:0:11:: remote-as 121
  neighbor 2001:18:0:11:: description eBGP with TRANSIT 1
  neighbor 2001:18:0:11:: password cisco
  neighbor 2001:18:0:11:: prefix-list AS102-v6block out
  neighbor 2001:18:0:11:: prefix-list default-v6route in
!
```

and peering router:

```
ip community-list 1 permit 102:1000
ip community-list 2 permit 102:1001
ip community-list 3 permit 102:1005
ip community-list 4 permit 102:1100
ip community-list 5 permit 102:1200
!
ip prefix-list AS102-block permit 100.68.20.0/24
ipv6 prefix-list AS102-v6block permit 2001:DB8:20::/48
!
ip prefix-list AS101-block permit 100.68.10.0/24
ipv6 prefix-list AS101-v6block permit 2001:DB8:10::/48
!
ip prefix-list IXP-RS permit 100.68.10.0/24
ip prefix-list IXP-RS permit 100.68.20.0/24
ip prefix-list IXP-RS permit 100.68.30.0/24
ip prefix-list IXP-RS permit 100.68.40.0/24
ip prefix-list IXP-RS permit 100.68.50.0/24
ip prefix-list IXP-RS permit 100.68.60.0/24
!
ipv6 prefix-list IXP-v6RS permit 2001:DB8:10::/48
ipv6 prefix-list IXP-v6RS permit 2001:DB8:20::/48
ipv6 prefix-list IXP-v6RS permit 2001:DB8:30::/48
ipv6 prefix-list IXP-v6RS permit 2001:DB8:40::/48
```

```
ipv6 prefix-list IXP-v6RS permit 2001:DB8:50::/48
ipv6 prefix-list IXP-v6RS permit 2001:DB8:60::/48
!
route-map IXP-peer permit 10
 set local-preference 180
 set community 102:1200
!
route-map private-peer permit 10
 set local-preference 200
 set community 102:1100
!
router bgp 102
 address-family ipv4
  neighbor 100.68.10.25 remote-as 101
  neighbor 100.68.10.25 description eBGP with AS101
  neighbor 100.68.10.25 password cisco
  neighbor 100.68.10.25 prefix-list AS102-block out
  neighbor 100.68.10.25 prefix-list AS101-block in
  neighbor 100.68.10.25 route-map private-peer-in in
  neighbor 100.127.1.254 remote-as 65534
  neighbor 100.127.1.254 description eBGP with IXP RS
  neighbor 100.127.1.254 password ixp-rs
  neighbor 100.127.1.254 prefix-list AS102-block out
  neighbor 100.127.1.254 prefix-list IXP-RS in
  neighbor 100.127.1.254 route-map IXP-peer-in in
!
 address-family ipv6
  neighbor 2001:DB8:10:12:: remote-as 101
  neighbor 2001:DB8:10:12:: description eBGP with AS101
  neighbor 2001:DB8:10:12:: password cisco
  neighbor 2001:DB8:10:12:: prefix-list AS102-v6block out
  neighbor 2001:DB8:10:12:: prefix-list AS101-v6block in
  neighbor 2001:DB8:10:12:: route-map private-peer-in in
  neighbor 2001:DB8:FFFF:1::FE remote-as 65534
  neighbor 2001:DB8:FFFF:1::FE description eBGP with IXP RS
  neighbor 2001:DB8:FFFF:1::FE password ixp-rs
  neighbor 2001:DB8:FFFF:1::FE prefix-list AS102-v6block out
  neighbor 2001:DB8:FFFF:1::FE prefix-list IXP-v6RS in
  neighbor 2001:DB8:FFFF:1::FE route-map IXP-peer-in in
!
```

Our goal now is to replace the outbound prefix lists with a route-map which matches the community settings.

In this example for AS102, we want to replace the prefix-lists called “AS102-block” and “AS102-v6block”. Each group should now replace their equivalent prefix-list:

```
ip prefix-list AS102-block permit 100.68.20.0/24
ipv6 prefix-list AS102-v6block permit 2001:DB8:20::/48
```

with route-maps for each of our three types of BGP neighbours. For our private peer we have:

```
route-map private-peer-out permit 10
 match community 1
```

For our IXP peers we have:

```
route-map IXP-peer-out permit 10
 match community 1
```

For our upstream we have:

```
route-map upstream-out permit 10
 match community 1
```

and then replace the relevant entries in the BGP configuration on the border router:

```
no ip prefix-list AS102-block permit 100.68.20.0/24
no ipv6 prefix-list AS102-v6block permit 2001:DB8:20::/48
!
router bgp 102
 address-family ipv4
  no neighbor 100.121.1.5 prefix-list AS102-block out
  neighbor 100.121.1.5 route-map upstream-out out
!
 address-family ipv6
  no neighbor 2001:18:0:11:: prefix-list AS102-v6block out
  neighbor 2001:18:0:11:: route-map upstream-out out
!
```

and the peering router:

```
no ip prefix-list AS102-block permit 100.68.20.0/24
no ipv6 prefix-list AS102-v6block permit 2001:DB8:20::/48
!
router bgp 102
 address-family ipv4
  no neighbor 100.68.10.25 prefix-list AS102-block out
  neighbor 100.68.10.25 route-map private-peer-out out
  no neighbor 100.127.1.254 prefix-list AS102-block out
  neighbor 100.127.1.254 route-map IXP-peer-out out
!
 address-family ipv6
  no neighbor 2001:DB8:10:12:: prefix-list AS102-v6block out
  neighbor 2001:DB8:10:12:: route-map private-peer-out out
  no neighbor 2001:DB8:FFFF:1::FE prefix-list AS102-v6block out
  neighbor 2001:DB8:FFFF:1::FE route-map IXP-peer-out out
!
```

Note that we used the same route-map for both IPv4 and IPv6 BGP Peerings. This is because the route-map has no address family specific configuration in it - it is simply matching a BGP attribute that applies to both IPv4 and IPv6 address families.

Implement the route-refresh and then observe the changes to the BGP table - there should not be any change at all! (Note that Cisco IOS does not send communities to eBGP neighbours unless that feature is turned on - we don't need to do this here, as we are not using communities to signal anything to our neighbour networks - we are simply using them for our internal operational convenience.)

Now in future if we need to add any prefixes to our address announcements, we simply tag them with the correct community (ASN:1000) and that prefix will be automatically announced to all peers.

NOTE: In the real operational Internet it is quite unlikely that anyone would use the identical outbound route-map for both transit, public and private peers which is why we gave each one different names here. It is more than likely that traffic engineering needs have to be considered, and different subnets being sent to private and public peers, depending on the needs of customers. Which would mean the use of a route-map per peer (the more common case).

Conclusion

This lab has shown the differences between peering and transit, and the value that operators place on each type. Transit is last resort as it has significant operational cost. Peering is highly desirable, and this lab has shown the differences between private peering and IXP peering configurations.