

IPv6 Infrastructure Lab - Routing

Setting up OSPF

Dual stack OSPF in each AS

In each AS configure OSPF routing. As we are running a dual stack network, we need to run OSPFv2 for IPv4 routes and OSPFv3 for IPv6 routes. This means setting up the OSPF process, marking **internal** interfaces as non-passive, and then configuring each **internal** interface and the loopback with the OSPF process. All the routers in an AS will be in the same OSPF *area 0* and use the same OSPF process ID.

OSPF should be configured on internal interfaces **only**. You do not want to set up adjacencies with devices outside your AS. Make sure that there are no *ip[v6] ospf* commands on external interfaces. A side effect of this is that external link addresses will not appear in the IGP (see the next section discussion iBGP deployment).

For example, the Core router has three interfaces connected to other routers in their AS, and would have the following (where X is replaced by the Group number):

```
router ospf 10X
  passive-interface default
  no passive-interface fastethernet 0/0
  no passive-interface gigabit 1/0
  no passive-interface gigabit 2/0
  log-adjacency-changes
!
ipv6 router ospf 10X
  passive-interface default
  no passive-interface fastethernet 0/0
  no passive-interface gigabit 1/0
  no passive-interface gigabit 2/0
  log-adjacency-changes
!
interface fastethernet 0/0
  ip ospf 10X area 0
  ipv6 ospf 10X area 0
!
interface gigabit 1/0
  ip ospf 10X area 0
  ipv6 ospf 10X area 0
!
interface gigabit 2/0
  ip ospf 10X area 0
  ipv6 ospf 10X area 0
!
interface loopback 0
  ip ospf 10X area 0
  ipv6 ospf 10X area 0
```

!

Notes:

- **Passive-interface default** makes sure that OSPF does not attempt to set up adjacencies on any interfaces apart from those specified by the **no passive-interface** commands.

- The number following **[ipv6] router ospf** is a process ID and is used inside the router only (so it can be any number). But for this lab we recommend the OSPF process ID be the same as the AS number (which is the convention used by a number of Network Operators).

Hint: look at your OSPFv2 configuration and compare it with the OSPFv3 configuration. It should be identical (apart from the IPv6 commands). The topology is the same so the configuration of both OSPFs should be the same too.

OSPF on Point-to-Point Ethernet Links.

We need to modify OSPF's behaviour on point-to-point broadcast media links, such as Ethernet, when there are only two devices on that media. If we declare such a situation to be point to point, then OSPF does not try and determine a designated or backup designated router; furthermore, there will be an improvement/simplification in SPF calculations and memory requirements on the router.

Those router teams which have OSPF configured over an Ethernet interface should now convert OSPF to point-to-point mode, for example:

```
interface fastethernet 0/0
 ip ospf network point-to-point
 ipv6 ospf network point-to-point
```

The result will be that the DR or BDR entry in the status column of **"show ip[v6]ospf neighbor"* will disappear, to be replaced with a FULL. The link is now treated like a point-to-point serial connection.

Ping Test

Check the routes via OSPF. Make sure you can see all the networks within your AS. Ping all loopback interfaces within your AS. Use the **"show ip[v6] ospf neighbor"* and **"show ip[v6] route"* commands. If you cannot see the other routers in your AS, you will not be able to bring up BGP in the next steps.

Telnet source address

Most Network Operators use the router Loopback address for administrative purposes as well as the anchor point for their network's iBGP sessions. In this step we will configure telnet so that it uses the loopback interface as the source address for all telnet packets (IPv4 and IPv6) originated by the router.

```
ip telnet source-interface loopback 0
```

To check that this has worked, telnet from your router to a neighbouring router and then enter the *"who"* command. You will see that you are logged in, and the source address will be displayed. For example, using telnet from C1 to B1 gives:

```
C1>who
  Line      User      Host(s)      Idle Location
*  2 vty 0    seclab      idle         00:00:00 100.68.10.2
```

Save the configuration

Don't forget to save the configuration to NVRAM!

Configuring iBGP

The next step is to configure iBGP mesh between the four routers in each autonomous system. We'll use a route-reflector set up to handle this, as this is very common practice today, and full mesh iBGP does not scale, as was covered in the presentations.

We will make the core router the route reflector, as is standard practice. The border router will be a client, the access router will be a client, and the peering router will also be a client. Before setting up the iBGP route reflector, we need to consider the following:

- border router connects to transit ISP, so can in theory receive the entire global routing table from the upstream (either as a default route, or as every individual announced prefix).
- peering and access routers connect to customers, private and public peers - the peering router should only hear routes that the peers should be able to receive. A common mistake is for peering routers in service provider backbones to carry the full Internet routing table, resulting in bandwidth hijack, misrouted traffic, and so on. The access router only needs routes local to the AS and the default route - it should never need to carry Internet routes.

Configuring Core Router iBGP

First we set up the core router. We create two peer groups, one for the standard iBGP mesh, the other for use with the access and peering routers. Here is an example:

```
router bgp 10X
  bgp deterministic-med
  no bgp default ipv4-unicast
  !
  address-family ipv4
    distance bgp 200 200 200
    neighbor ibgp-partial peer-group
    neighbor ibgp-partial description Local Routes only
    neighbor ibgp-partial remote-as 10X
    neighbor ibgp-partial update-source loopback0
    neighbor ibgp-partial next-hop-self
    neighbor ibgp-partial send-community
    neighbor ibgp-partial route-reflector-client
    neighbor ibgp-partial filter-list 10 out
    neighbor ibgp-full peer-group
    neighbor ibgp-full description Local Routes only
    neighbor ibgp-full remote-as 10X
    neighbor ibgp-full update-source loopback0
```

```
neighbor ibgp-full next-hop-self
neighbor ibgp-full send-community
neighbor ibgp-full route-reflector-client
neighbor 100.68.X0.1 peer-group ibgp-full
neighbor 100.68.X0.1 description iBGP with BX
neighbor 100.68.X0.3 peer-group ibgp-partial
neighbor 100.68.X0.3 description iBGP with PX
neighbor 100.68.X0.4 peer-group ibgp-partial
neighbor 100.68.X0.4 description iBGP with AX
!
address-family ipv6
distance bgp 200 200 200
neighbor ibgp-partialv6 peer-group
neighbor ibgp-partialv6 remote-as 10X
neighbor ibgp-partialv6 description Local Routes only
neighbor ibgp-partialv6 update-source Loopback0
neighbor ibgp-partialv6 send-community
neighbor ibgp-partialv6 route-reflector-client
neighbor ibgp-partialv6 next-hop-self
neighbor ibgp-partialv6 filter-list 10 out
neighbor ibgp-fullv6 peer-group
neighbor ibgp-fullv6 remote-as 10X
neighbor ibgp-fullv6 description Local Routes only
neighbor ibgp-fullv6 update-source Loopback0
neighbor ibgp-fullv6 send-community
neighbor ibgp-fullv6 route-reflector-client
neighbor ibgp-fullv6 next-hop-self
neighbor 2001:DB8:X0::1 peer-group ibgp-fullv6
neighbor 2001:DB8:X0::1 description iBGP with BX
neighbor 2001:DB8:X0::3 peer-group ibgp-partialv6
neighbor 2001:DB8:X0::3 description iBGP with PX
neighbor 2001:DB8:X0::4 peer-group ibgp-partialv6
neighbor 2001:DB8:X0::4 description iBGP with AX
!
ip as-path access-list 10 permit ^$
```

Notice how we have set up one peer-group called `ibgp-partial` for use with the access and peering routers - its only difference from the peer-group called `ibgp-full` is that it has one additional line `only permit` prefixes originated by the local AS to go to that router. So if the upstream provider sends a default route, or any prefixes from the global BGP table, they will now not make their way to the peering router. While we have used an AS-path filter here, we could also use BGP communities (much more scalable!).

Originating Prefixes

We will now originate our prefixes into iBGP. We will only do this on the core router (common practice is to originate prefixes on the core routers in a network operator's backbone, never on the peering or border routers). So, returning to the core router, we now add in network statements to cover our IPv4 and IPv6 address blocks. Here is an example:

```
router bgp 10X
```

```

address-family ipv4
  network 100.68.X0.0 mask 255.255.255.0
address-family ipv6
  network 2001:DB8:X0::/48
!
ip route 100.68.X0.0 255.255.255.0 Null0
ipv6 route 2001:DB8:X0::/48 Null0

```

Don't forget the pull up routes for the aggregate - the network statement in Cisco IOS only tells BGP to put that address block into BGP if the match block is in the global RIB - and the simplest way to install it in the global RIB is to set up a static route pointing to the Null0 interface.

Configuring Access and Peering Router iBGP

We now turn to the access and peering routers, and will configure iBGP on those as well. We'll follow the same ideas as we used for the Core router, only the access and peering routers are route reflector clients. Here is a configuration example for either the access or peering routers:

```

router bgp 10X
  bgp deterministic-med
  no bgp default ipv4-unicast
!
address-family ipv4
  distance bgp 200 200 200
  neighbor ibgp-rr peer-group
  neighbor ibgp-rr description iBGP with RR
  neighbor ibgp-rr remote-as 10X
  neighbor ibgp-rr update-source loopback0
  neighbor ibgp-rr next-hop-self
  neighbor ibgp-rr send-community
  neighbor 100.68.X0.2 peer-group ibgp-rr
  neighbor 100.68.X0.2 description iBGP with CX
!
address-family ipv6
  < do the same for IPv6 >
!

```

Configuring Border Router iBGP

We now turn to the border router, and will configure iBGP on that as well. We'll follow the same ideas as we used for the Core router, only the Border router is a route reflector client. Here is a configuration example:

```

router bgp 10X
  bgp deterministic-med
  no bgp default ipv4-unicast
!
address-family ipv4
  distance bgp 200 200 200
  neighbor ibgp-rr peer-group
  neighbor ibgp-rr description iBGP with RR

```

```
neighbor ibgp-rr remote-as 10X
neighbor ibgp-rr update-source loopback0
neighbor ibgp-rr next-hop-self
neighbor ibgp-rr send-community
neighbor 100.68.X0.2 peer-group ibgp-rr
neighbor 100.68.X0.2 description iBGP with CX
!
address-family ipv6
< do the same for IPv6 >
!
```

Notice that the peer-group is identical to the one used on the Peering Router.

Improving Routing Security

There are a few things we need to tidy up here before we continue with the lab.

Peering Router

The peering router is just that, a router that peers with other network operators. It does not provide any transit. The peers should only see the routes that you want them to see. We've made sure of this by putting in a route filter on the core router so that the peering router can only see locally originated routes. But it is also a good idea to null route the default route, as we will soon be distributing a default route around the AS using IS-IS. So on the peering router we now do:

```
ip route 0.0.0.0 0.0.0.0 null0
ipv6 route ::/0 null0
```

Now if any of the IXP participants point a default route to the local network, the traffic will simply be dumped in the Null interface of the peering router. Only traffic for specific destinations which are available in the routing table on the IXP router will be forwarded to the rest of the network. This is a very important **network security** requirement.

Border Router

The border router connects to the upstream provider, and therefore gives us access to the whole Internet. The upstream provider will usually send us a default route by eBGP (yet to be set up). Once we hear this default route, how should it be propagated around the autonomous system?

It can be propagated using iBGP, but that tends to be non-optimal, certainly when trying to load balance between two or more transit providers, as the BGP best path is just that. If we distribute the default by the IGP instead, then at least the default route becomes the nearest exit, to the nearest border router. So we will now configure this - for example:

```
router ospf 10X
 default-information originate
!
ipv6 router ospf 10X
 default-information originate
```

```
!
```

We now should be ready to proceed with the next part of the lab.

Adding a customer route

Consult the address plan and now set up customer routes on the Access Router. We'll just use one IPv4 and one IPv6 customer per Autonomous System. Take the address assigned for the "End User Space 1"; as we don't have any access routers we'll simply point a static route for the customer route to the Null interface on the Access router. We'll use a /26 for IPv4 and a /52 for IPv6 out of the "End User Space 1" assignments.

Here is an example for the access router.

```
ip route 100.68.X0.64 255.255.255.192 null0
ipv6 route 2001:DB8:X0:4000::/52 null0
!
router bgp 10X
  address-family ipv4
    network 100.68.X0.64 mask 255.255.255.192
  address-family ipv6
    network 2001:DB8:X0:4000::/52
!
```

Once this has been configured you should then see the customer IPv4 /26 and IPv6 /52 visible in the iBGP for the AS. Check on the Border, Peering and Core routers, and make sure the prefix is visible. Use these commands:

```
show ip bgp
show bgp ipv6 unicast
```

Configuring the link to the Transit Provider

The next step is to set up eBGP with the Transit Provider. The lab instructors will be running the routers for the two Transit Providers and will have already configured them.

Physical Link

Follow the guidelines in the [IP Address Plan](address-plan) document to configure the link to the upstream. Make sure that you can ping the upstream's router using both IPv4 and IPv6 - if it doesn't work, investigate why.

Here is an example configuration for AS106:

```
interface fastethernet 0/0
  description Link to TR2
  ip address 100.122.1.10 255.255.255.252
```

```
ipv6 address 2001:19:0:12::1/127
!
```

OSPF

Do not configure OSPFv2 or OSPFv3 towards the upstream provider! They are not part of your autonomous system.

External BGP

We now configure eBGP with the upstream. Again, the configuration on the two transit routers will have already been completed by the instructors, so once configured, the eBGP session should just come up and work.

Here is a configuration example for AS101:

```
router bgp 101
 address-family ipv4
  neighbor 100.121.1.1 remote-as 121
  neighbor 100.121.1.1 description eBGP with TRANSIT 1
 !
 address-family ipv6
  neighbor 2001:18:0:10:: remote-as 121
  neighbor 2001:18:0:10:: description eBGP with TRANSIT 1
 !
```

Once this has been configured, you should now see routes coming from the upstream provider, and you should be able to see your aggregate being sent to your upstream. The commands to see what you are receiving are:

```
show ip bgp neigh 100.121.1.1 routes
show bgp ipv6 uni neigh 2001:18:0:10:: routes
```

and to show what you are sending:

```
show ip bgp neigh 100.121.1.1 advertised-routes
show bgp ipv6 uni neigh 2001:18:0:10:: advertised-routes
```

Confirmation

Check on the Core and Peering Router what you now see in the BGP table. Are there differences? Can you explain what they are, and why?

What does:

```
show ip bgp 0.0.0.0 0.0.0.0
show ip route 0.0.0.0 0.0.0.0
show ipv6 route ::/0
show bgp ipv6 uni ::/0
```

show you on the core router and on the peering router?

You will notice that the default route is being propagated by BGP throughout the AS.

- On the core router, it will complain of a **RIB failure** when you look at the BGP table, because OSPF has a default route with a lower protocol distance.
- On the border router, it will complain of a **RIB failure** when you look at the BGP table, because there is a static default route to the Null interface.

While there is nothing wrong with a **RIB failure**, we can just remove the default from being propagated by the iBGP. To do this, we go back to the eBGP session, look for the default, tag it with the *no-advertise* community, and then the border router will no longer announce the default by iBGP.

Here is a configuration example for AS102:

```
ip prefix-list default-route permit 0.0.0.0/0
!
route-map tag-default permit 10
  match ip address prefix-list default-route
  set community no-advertise
!
route-map tag-default permit 20
!
ipv6 prefix-list default-v6route permit ::/0
!
route-map tag-v6default permit 10
  match ipv6 address prefix-list default-v6route
  set community no-advertise
!
route-map tag-v6default permit 20
!
router bgp 102
  address-family ipv4
    neighbor 100.121.1.5 route-map tag-default in
  address-family ipv6
    neighbor 2001:18:0:11:: route-map tag-v6default in
!
```

To confirm, has the default route now disappeared from the BGP table on the Core and Peering routers? If not, did you remember to do route-refresh after you applied the above configuration?

Testing

Once this has been completed, test the connectivity. Can you reach the other groups in the class? You should be able to ping all the IPv4 and IPv6 loopbacks across the whole classroom.

Can you see the Internet too? The lab has IPv4 connectivity to the Internet - check that this works by trying a few pings or trace routes to well known destinations (e.g. to 8.8.8.8).

From:

<https://bgp4all.com/pfs/> - **Philip Smith's Internet Development Site**

Permanent link:

<https://bgp4all.com/pfs/training/itu-ipv6-kh/1-routing>

Last update: **2016/06/19 00:52**

