

RIPE NCC
RIPE NETWORK COORDINATION CENTRE

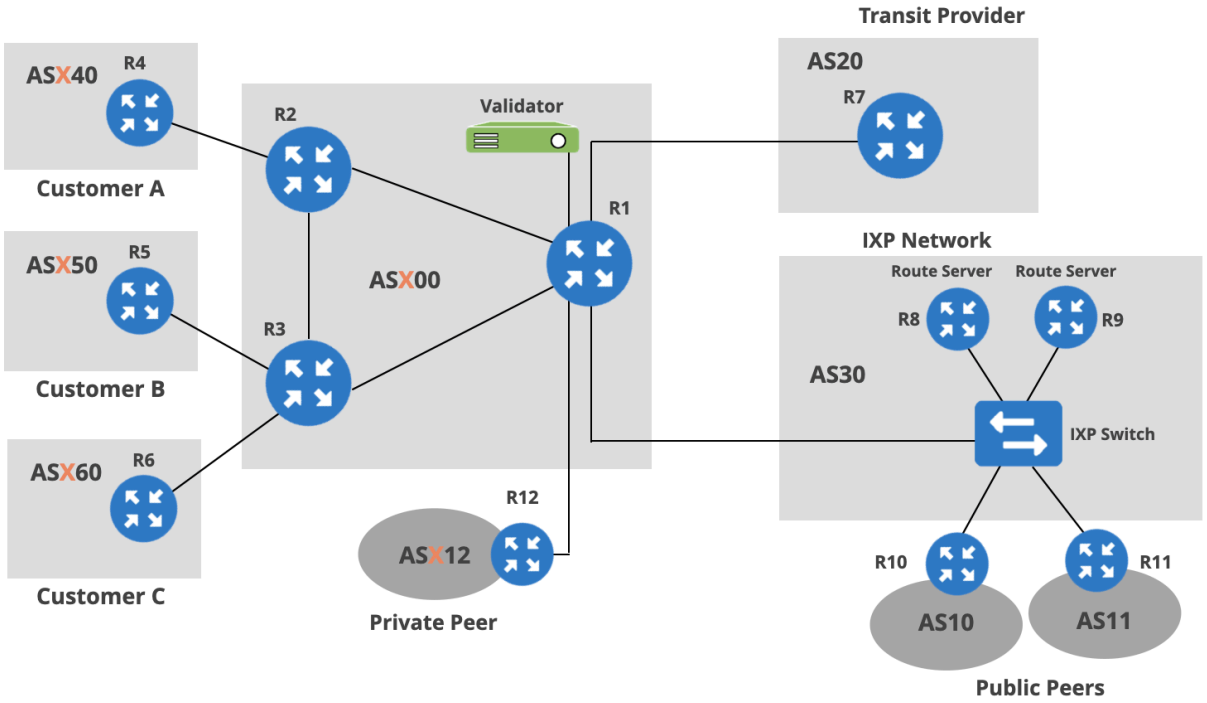
BGP Routing Security

Lab Activity Guide

Version: October 2026

Introduction and Topology

As network administrator of Sandbox Inc.'s IPv6 network, you're responsible for its design and maintenance, and for making sure it's always fully operational, resilient, and protected. As such, you decide to implement some of the Best Current Practices (BCPs) of BGP Routing Security.



The trainer will assign you a number (X) that will be used for AS numbers and prefixes.

Your network gets transit service from AS20, it has one private peer (ASX12) and several public peers (AS10 and AS11) via an IXP. Sandbox Inc. (ASX00) has three customers, Customer A, Customer B and Customer C, which have their own IPv6 address blocks and AS numbers and run BGP. As an upstream provider, Sandbox network provides a transit service to its customers and connects them to the Internet.

In the following activities, you'll be using the shown network topology diagram. You'll do configurations on your routers: R1, R2 and R3, and on the R12 router of your private peer, ASX12. The rest of the routers run eBGP and are already preconfigured with BGP routing policies to send and receive prefixes.

Your RIPE NCC Access Account

All services that the RIPE NCC offers are supported by the RIPE NCC Access Single Sign-On (SSO) system, including the hands-on lab environment used for this course (Workbench).

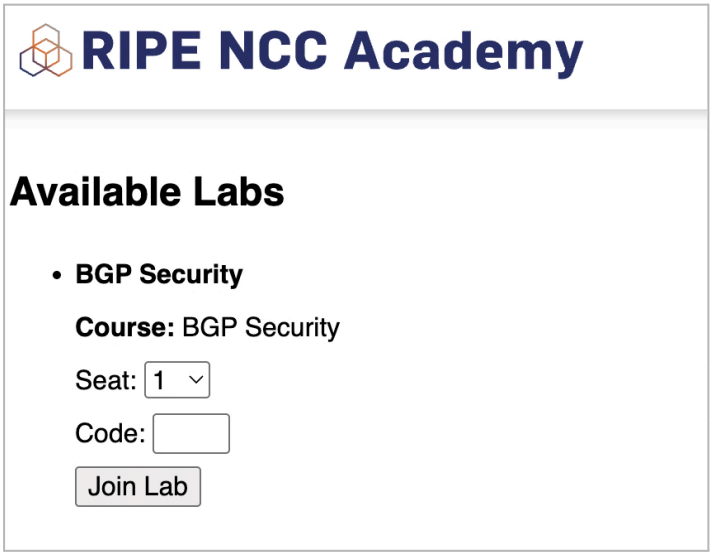
You **must** have a RIPE NCC Access account with Two-Factor Authentication (2FA) enabled to take this training course. If you do not have an account yet, create one in advance by following these **instructions**.

If you already have a RIPE NCC Access account, go to **access.ripe.net** and try to log in.
If you have any issues accessing your account, let the trainers know so they can help you.

Entering the lab environment

1. Go to **https://workbench.ripe.net** and log in using your RIPE NCC Access account.
2. The trainers will give you a **seat number** (Number **X**) and a **code** to access the lab.
3. After logging in, find the **BGP Security** lab,
4. Select your **seat numbe**r from the dropdown menu
5. Enter the **code** provided by the trainer
6. Click on the **Join lab** button to access the lab.

Important: You cannot share a RIPE NCC Access account with another participant. Each participant needs a unique account to access their seat in the hands-on labs.



You will see a web page with terminals available. Each terminal corresponds to one of the routers you will be configuring.

The commands that you will need to type are shown in `blue`. Your number assigned by the trainer and some important outputs that need to be emphasised are shown in `orange`.

The routers in the lab network are running an open-source routing software, FRR (Free Range Routing) version 8.4.2 (https://frrouting.org).

IP Allocations and AS numbers

Your AS number:	AS **X**00
Your IPv6 allocation:	2001:db8:**X**00::/48

Lab Activity 1: Securing BGP Sessions

In this lab activity, you will implement two techniques to protect BGP sessions:

- MD5
- GTSM and BGP TTL security

Activity 1.1: BGP session protection with MD5 authentication

In this lab activity, you'll configure MD5 authentication on a TCP connection between two BGP peers. You must configure the same password on both sides.

Step 1: Check your BGP sessions in R1

R1 has six BGP sessions in total: iBGP sessions with R2 and R3, and eBGP sessions with the transit provider, private peer and IXP route servers (R8 and R9).

Before starting the activity make sure that all your BGP sessions are up and running.

The output below shows the sessions are up for several minutes (check **Up/Down**) and R1 sends and/or receives prefixes from some of its peers (check **PfxSnt** and **PfxRcd**).

```
u0X-R1# show bgp summary

IPv6 Unicast Summary (VRF default):
BGP router identifier  X.1.1.1, local AS number X00 vrf-id 0
BGP table version 20
RIB entries 31, using 5952 bytes of memory
Peers 6, using 4303 KiB of memory

Neighbor            V         AS   MsgRcvd   MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd   PfxSnt Desc
2001:db8:30:30::8   4         30     16372     14331        0    0    0 01w2d22h           14 (Policy) N/A
2001:db8:30:30::9   4         30     16384     14331        0    0    0 01w2d22h           14 (Policy) N/A
2001:db8:X00::2     4        X00     14331     14338        0    0    0 01w2d22h            3        16 N/A
2001:db8:X00::3     4        X00     14331     14338        0    0    0 01w2d22h            1        16 N/A
2001:db8:X00:17::7  4         20     14334     14331        0    0    0 01w2d22h            7 (Policy) N/A
2001:db8:X00:112::12 4       X12     14332     14331        0    0    0 01w2d22h     (Policy)        2 N/A

Total number of neighbors 6
```

Step 2: Configure MD5 authentication in R1 with R12 (private peer)

```
u0X-R1# conf
u0X-R1(config)# router bgp X00
u0X-R1(config-router)# neighbor 2001:db8:X00:112::12 password Test123!
u0X-R1(config-router)# end
```

Step 3: On R1, check your BGP sessions again

Check the state of the BGP sessions with your private peer's router. Is the eBGP session with R12 still up?

```
u0X-R1# show bgp summary

IPv6 Unicast Summary (VRF default):
BGP router identifier  X.1.1.1, local AS number X00 vrf-id 0
BGP table version 20
RIB entries 26, using 4992 bytes of memory
Peers 6, using 4303 KiB of memory

Neighbor            V         AS   MsgRcvd   MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd   PfxSnt Desc
2001:db8:30:30::8   4         30     16385     14343        0    0    0 01w2d23h           14 (Policy) N/A
2001:db8:30:30::9   4         30     16397     14343        0    0    0 01w2d23h           14 (Policy) N/A
2001:db8:X00::2     4        X00     14342     14349        0    0    0 01w2d22h            3       16 N/A
2001:db8:X00::3     4        X00     14342     14349        0    0    0 01w2d22h            1       16 N/A
2001:db8:X00:17::7  4         20     14346     14343        0    0    0 01w2d23h            7 (Policy) N/A
2001:db8:X00:112::12 4       X12     14341     14341        0    0    0 00:02:40      Connect        0 N/A

Total number of neighbors 6
```

No, it's not. An eBGP session with R12 fails and it stays in Connect state. This is because MD5 is enabled on one side of the BGP session only, on R1, but not on the peer router. You must configure the same password on both sides, otherwise the BGP session will never come up!

Step 4: Configure MD5 on R12

Go to R12 and enable MD5 with the same password. You can find R12 in the **Private Peer** tab in the left hand menu of the lab.

```
u0X-R12# conf
u0X-R12(config)# router bgp X12
u0X-R12(config-router)# neighbor 2001:db8:X00:112::1 password Test123!
u0X-R12(config-router)# end
```

Step 5: On R1, check the BGP session with R12 again

From the output you can see that the BGP session is established and MD5 authentication is enabled!

```
u0X-R1# show bgp neighbors 2001:db8:X00:112::12
BGP neighbor is 2001:db8:X00:112::12, remote AS X12, local AS X00, external link
  Local Role: undefined
  Remote Role: undefined
Hostname: u0X-R12
  BGP version 4, remote router ID X.12.12.12, local router ID X.1.1.1
  BGP state = Established, up for 00:00:44
BGP state = Established, up for 00:01:10
```

```
  Last read 00:00:10, Last write 00:00:10
  Hold time is 180 seconds, keepalive interval is 60 seconds
  Configured hold time is 180 seconds, keepalive interval is 60 seconds
  Configured conditional advertisements interval is 60 seconds
  Neighbor capabilities:
    4 Byte AS: advertised and received
.
.

Local host: 2001:db8:X00:112::1, Local port: 179
Foreign host: 2001:db8:X00:112::12, Foreign port: 53200
.
.
BGP connection: shared network
BGP Connect Retry Timer in Seconds: 120
Peer Authentication Enabled
Read thread: on  Write thread: on  FD used: 31
```

Summary

In this activity, you enabled MD5 authentication between R1 and your private peer's BGP speaker R12, and configured them with the same pre-shared secret key.

Since MD5 is configured, both routers will generate MD5 digest for each segment sent on a TCP connection and verify it to ensure the authenticity of the message. Although TCP-MD5 is widely deployed in many networks due to its availability in vendors' equipment, it has some limitations which made it obsolete by TCP-AO.

It's recommended to use TCP-AO to secure BGP sessions if it's supported by your vendor. In the lab network, we're running Free Range Routing (FRR) software which is a Linux-based routing implementation. It doesn't support TCP-AO yet since it's waiting for TCP-AO support in the kernel to offer it for routing protocols.

Activity 1.2: BGP session protection with GTSM and BGP TTL security

In this activity, you will implement the Generalised TTL Security Mechanism (GTSM) in an eBGP session. Like MD5 protection, TTL security must be configured on both ends of a BGP session.

Step 1: Enable GTSM on R1

Go to R1 and enable GTSM on the eBGP session with R12.

```
u0X-R1# conf
u0X-R1(config)# router bgp X00
u0X-R1(config-router)# neighbor 2001:db8:X00:112::12 ttl-security hops 1
u0X-R1(config-router)# end
```

Step 2: Check your eBGP session with R12

In R1, check your eBGP session with R12 and see if GTSM is enabled in the output. Is the session up?

```
u0X-R1# show bgp neighbors 2001:db8:X00:112::12
BGP neighbor is 2001:db8:X00:112::12, remote AS X12, local AS X00, external link
  Local Role: undefined
  Remote Role: undefined
Hostname: u0X-R12
  BGP version 4, remote router ID 0.0.0.0, local router ID X.1.1.1
  BGP state = Active
  Last read 00:01:38, Last write 00:01:38
  Hold time is 180 seconds, keepalive interval is 60 seconds

.

 Last reset 00:00:40,  Waiting for peer OPEN
  External BGP neighbor may be up to 1 hops away.
BGP Connect Retry Timer in Seconds: 120
Next connect timer due in 82 seconds
Peer Authentication Enabled
Read thread: off  Write thread: off  FD used: -1
```

No, it's not up. In the output you can see that the BGP session with R12 fails as it is in **Active** state. There is also another log added to the output: "**External BGP neighbour may be up to 1 hops away.**" This indicates that GTSM is enabled and the eBGP peer must be directly connected. This makes it impossible for remote attackers to deliver spoofed packets with TTL=255 via interfaces that are not directly connected.

Do not forget to configure GTSM on R12 as well. Remember that GTSM should be applied on a per-peer basis to provide protection against spoofed BGP messages. When enabled, the eBGP session is secured in the incoming direction only and has no effect on outgoing IP packets or the remote device.

Step 3: Enable GTSM on R12 router

On the R12 router, enable GTSM on the eBGP session with R1.

```
u0X-R12# conf
u0X-R12(config)# router bgp X12
u0X-R12(config-router)# neighbor 2001:db8:X00:112::1 ttl-security hops 1
u0X-R12(config-router)# end
```

Step 4: Check your eBGP session with R12 again

Check the eBGP session with R12 again on R1. It may take a little longer than usual for the session to get established. You can clear the session to speed up the process.

```
u0X-R1# clear bgp 2001:db8:X00:112::12
u0X-R1# show bgp neighbors 2001:db8:X00:112::12
BGP neighbor is 2001:db8:X00:112::12, remote AS X12, local AS X00, external link
  Local Role: undefined
  Remote Role: undefined
Hostname: u0X-R12
  BGP version 4, remote router ID X.12.12.12, local router ID X.1.1.1
  BGP state = Established, up for 00:00:17
  Last read 00:00:15, Last write 00:00:15
  Hold time is 180 seconds, keepalive interval is 60 seconds
  Configured hold time is 180 seconds, keepalive interval is 60 seconds
  Configured conditional advertisements interval is 60 seconds
  Neighbor capabilities:
    4 Byte AS: advertised and received
    Extended Message: advertised and received
.
.
 External BGP neighbor may be up to 1 hops away.
Local host: 2001:db8:X00:112::1, Local port: 58426
Foreign host: 2001:db8:X00:112::12, Foreign port: 179
.
.
```

Summary

In this activity, you enabled GTSM and TTL security on the eBGP session in R1 with your private peer, R12. Enabling GTSM protects a router's control plane from CPU utilisation-based attacks caused by forged IP packets. It checks the TTL (IPv4) or Hop-limit (IPv6) of incoming IP packets and makes sure that they have not been spoofed.

Lab Activity 2: Creating BGP Prefix Filters

In the following activities, you'll create BGP filters with your transit provider, public/private peers, and your customers, based on the recommendations we have seen in the course for the different peering relationships. During those activities you'll also see different filtering methods such as prefix-list filters and filtering based on BGP communities.

Activity 2.1: Filtering with transit providers and peers

As we have seen on the course, the recommendation for full routing networks is to apply filtering policies on border routers for each eBGP session. This will protect your network and your peers from the effects of misconfigurations. On the **inbound** direction, if the full routing table (FRT) is desired from the transit provider, the recommendation is to filter prefixes that are not globally routable, prefixes not allocated by IANA (IPv6 only), routes that are too specific, your own prefixes, IXP LAN prefixes, and the default route (if not requested from transit provider).

The same recommendation is valid for the routes received from public and private peers.

In the **outbound** direction, you should only send prefixes that belong to your network and your customer's networks to your eBGP peers. To prevent routing issues due to misconfiguration, you should also filter the following in your outbound policies:

- Prefixes that are not globally routable
- Prefixes that are too specific
- IXP LAN prefixes, and the default route

In our lab, you get transit service from AS20 (transit provider), which connects you to the rest of the Internet. Also, your company exists at the IXP location (AS30) and has two public BGP peers through the eBGP connection with IXP route servers (R8 and R9). In addition to public peers, you also have one private peer, AS**X**12.

The R1 router has eBGP sessions with transit provider (R7) and IXP route servers (R8 and R9). On those eBGP sessions, a default inbound policy is applied which accepts any route so that we can see which routes are coming from transit and IXP peers. However, there is no outbound policy applied yet.

In this lab activity, you'll configure an inbound and outbound policy for eBGP sessions with your transit provider and public peers, based on the mentioned recommendations.

Check your eBGP session with the transit provider and IXP Route Servers

```
u0X-R1# show bgp summary

IPv6 Unicast Summary (VRF default):
BGP router identifier X.1.1.1, local AS number X00 vrf-id 0
BGP table version 26
RIB entries 41, using 7872 bytes of memory
Peers 6, using 4303 KiB of memory

Neighbor            V         AS   MsgRcvd   MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd   PfxSnt Desc
2001:db8:30:30::8   4         30     18232     15968        0    0    0 01w4d02h           14 (Policy) N/A
2001:db8:30:30::9   4         30     18255     15968        0    0    0 01w4d02h           14 (Policy) N/A
2001:db8:X00::2     4        X00     15969     15975        0    0    0 01w4d02h            3       16 N/A
2001:db8:X00::3     4        X00     15969     15975        0    0    0 01w4d02h            1       16 N/A
2001:db8:X00:17::7  4         20     15975     15978        0    0    0 01w4d02h            7 (Policy) N/A
2001:db8:X00:112::12 4       X12     15971     15971        0    0    0 1d02h50m     (Policy)       2  N/A

Total number of neighbors 6
```

You can see prefixes received (State/PfxRcd) and nothing sent (PfxSnt) from both route server peers (AS 30) and the transit provider (AS 20). Remember that **(Policy)** means that no policy has been configured on the eBGP session allowing the exchange of routing information.

Check BGP routes received from the transit provider

```
u0X-R1# show bgp neighbors 2001:db8:X00:17::7 received-routes
BGP table version is 26, local router ID is X.1.1.1, vrf id 0
Default local pref 100, local AS X00
...
   Network          Next Hop            Metric LocPrf Weight Path
*> ::/0             2001:db8:X00:17::7
                                             0             0 20 i
*> 2001:db8::/32    2001:db8:X00:17::7
                                             0             0 20 i
*> 2001:db8:65:1::/64
                    2001:db8:X00:17::7
                                                           0 20 65 i
*> 2001:db8:66::/48 2001:db8:X00:17::7
                                                           0 20 66 i
*> 2001:ff65::/32   2001:db8:X00:17::7
                                                           0 20 65 i
*> 2001:ff75::/32   2001:db8:X00:17::7
                                                           0 20 65 i
*> 2001:ff85::/32   2001:db8:X00:17::7
                                                           0 20 65 i

Total number of prefixes 7
```

All participants are connected to the same transit provider and IXP. They do not filter routes so in the previous and following command outputs you may see more prefixes coming from other participants.

Check BGP routes received from IXP peers

```
u0X-R1# show bgp neighbors 2001:db8:30:30::8 received-routes
BGP table version is 41, local router ID is X.1.1.1, vrf id 0
Default local pref 100, local AS X00
...
   Network          Next Hop            Metric LocPrf Weight Path
*> ::/0             2001:db8:30:30::10
                                             0             0 10 i
*> 2001:db8:65:1::/64
                    2001:db8:30:30::65
                                             0             0 65 i
*> 2001:db8:66::/48 2001:db8:30:30::66
                                             0             0 66 i
*> 2001:db8:bbbb::/48
                    2001:db8:30:30::10
                                             0             0 10 i
*> 2001:db8:bbbb:1::/64
                    2001:db8:30:30::10
                                             0             0 10 i
*> 2001:db8:bbbb:2::/64
                    2001:db8:30:30::10
                                             0             0 10 i
*> 2001:db8:cccc::/48
                    2001:db8:30:30::11
                                             0             0 11 i
*> 2001:db8:cccc:1::/64
                    2001:db8:30:30::11
                                             0             0 11 i
*> 2001:db8:cccc:2::/64
                    2001:db8:30:30::11
                                             0             0 11 i
*> 2001:ff65::/32   2001:db8:30:30::65
                                             0             0 65 i
*> 2001:ff75::/32   2001:db8:30:30::65
                                             0             0 65 i
*> 2001:ff85::/32   2001:db8:30:30::65
                                             0             0 65 i
*> 2001:ffc1::/32   2001:db8:30:30::11
                                             0             0 11 23456 5000 5000 4000 i
*> 2001:ffc2::/32   2001:db8:30:30::11
                                             0             0 11 23456 5000 5000 4000 i

Total number of prefixes 14
```

You should see the same routes received from the other route server (R9) in the IXP (show bgp neighbors 2001:db8:30:30::9 received-routes).

Step 1: Create an inbound policy on R1

In this lab activity you'll create a loose inbound policy on R1 which filters the default route, too specific prefixes (the ones longer than /48), prefixes that belong to your own AS and IXP LAN.

In addition, other routes should be filtered from the transit provider or IXP peers as well, such as private blocks, reserved blocks and the prefixes not allocated by IANA. Since we're using documentation prefixes in the lab, BOGON filters will not be included in this filter configuration.

Step 1.1: Create a prefix-list for a default route (::/0)

```
u0X-R1# conf
u0X-R1(config)# ipv6 prefix-list DEFAULT-v6 seq 5 permit ::/0
```

Step 1.2: Create a prefix-list for more specific prefixes

```
u0X-R1(config)# ipv6 prefix-list TOO-SPECIFIC-v6 seq 5 permit ::/0 ge 49
```

Step 1.3: Create a prefix-list matching your own prefixes

```
u0X-R1(config)# ipv6 prefix-list ASX00-PREFIXES seq 5 permit 2001:db8:X00::/48
```

Step 1.4: Create a prefix-list for IXP LAN

```
u0X-R1(config)# ipv6 prefix-list IXP-LAN seq 5 permit 2001:db8:30:30::/64
```

Step 1.5: Create a route-map for the transit provider

Create a route-map, **INBOUND-TRANSIT**, to discard the routes in the prefix-lists created and allow any other route and also a default route.

```
u0X-R1(config)# route-map INBOUND-TRANSIT deny 20
u0X-R1(config-route-map)# match ipv6 address prefix-list TOO-SPECIFIC-v6
u0X-R1(config-route-map)# route-map INBOUND-TRANSIT deny 30
u0X-R1(config-route-map)# match ipv6 address prefix-list ASX00-PREFIXES
u0X-R1(config-route-map)# route-map INBOUND-TRANSIT deny 40
u0X-R1(config-route-map)# match ipv6 address prefix-list IXP-LAN
u0X-R1(config-route-map)# route-map INBOUND-TRANSIT permit 60
u0X-R1(config-route-map)# match ipv6 address prefix-list ANY
u0X-R1(config-route-map)# exit
```

Step 1.6: Create a route-map for IXP

Create a route-map, **INBOUND-IX**, to discard the routes in the prefix-lists created and allow any other route.

```
u0X-R1(config)# route-map INBOUND-IX deny 10
u0X-R1(config-route-map)# match ipv6 address prefix-list DEFAULT-v6
u0X-R1(config-route-map)# route-map INBOUND-IX deny 20
u0X-R1(config-route-map)# match ipv6 address prefix-list TOO-SPECIFIC-v6
u0X-R1(config-route-map)# route-map INBOUND-IX deny 30
u0X-R1(config-route-map)# match ipv6 address prefix-list ASX00-PREFIXES
u0X-R1(config-route-map)# route-map INBOUND-IX deny 40
```

```
u0X-R1(config-route-map)# match ipv6 address prefix-list IXP-LAN
u0X-R1(config-route-map)# route-map INBOUND-IX permit 60
u0X-R1(config-route-map)# match ipv6 address prefix-list ANY
u0X-R1(config-route-map)# exit
```

Step 2: Apply the inbound policy to transit and IXP peers

```
u0X-R1(config)# router bgp X00
u0X-R1(config-router)# address-family ipv6 unicast
u0X-R1(config-router-af)# neighbor 2001:db8:X00:17::7 route-map INBOUND-TRANSIT in
u0X-R1(config-router-af)# neighbor 2001:db8:30:30::8 route-map INBOUND-IX in
u0X-R1(config-router-af)# neighbor 2001:db8:30:30::9 route-map INBOUND-IX in
u0X-R1(config-router-af)# end
```

Meanwhile in the existing configuration, ‘prefix-list ANY’ is already applied on eBGP sessions with transit providers and IXP peers. But this configuration will be ignored once route-map is applied. This is because the order of preference will favour the route-map.

Step 3: Check BGP routes from the transit provider and IXP peers again

If you look at the BGP sessions, you can see a lower number of prefixes received from the route servers in the IXP (AS30) and the transit provider (AS20).

```
u0X-R1# show bgp summary

IPv6 Unicast Summary (VRF default):
BGP router identifier X.1.1.1, local AS number X00 vrf-id 0
BGP table version 39
RIB entries 41, using 7872 bytes of memory
Peers 6, using 4303 KiB of memory

Neighbor            V         AS   MsgRcvd   MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd   PfxSnt Desc
2001:db8:30:30::8   4         30     18255     15988        0    0    0 01w4d02h            8 (Policy) N/A
2001:db8:30:30::9   4         30     18278     15988        0    0    0 01w4d02h            8 (Policy) N/A
2001:db8:X00::2     4        X00     15989     15997        0    0    0 01w4d02h            3         11 N/A
2001:db8:X00::3     4        X00     15989     15997        0    0    0 01w4d02h            1         11 N/A
2001:db8:X00:17::7  4         20     15995     15998        0    0    0 01w4d02h            6 (Policy) N/A
2001:db8:X00:112::12 4       X12     15991     15991        0    0    0 1d03h10m     (Policy)       2  N/A

Total number of neighbors 6
```

You can see the specific prefixes being filtered out after applying the new inbound policy:

```
u0X-R1# show bgp neighbors 2001:db8:X00:17::7 filtered-routes
BGP table version is 39, local router ID is X.1.1.1, vrf id 0
Default local pref 100, local AS X00
...
   Network          Next Hop            Metric LocPrf Weight Path
```

```
*> 2001:db8:65:1::/64
                    2001:db8:X00:17::7
                                                          0 20 65 i
Total number of prefixes 1
```

```
u0X-R1# show bgp neighbors 2001:db8:30:30::8 filtered-routes
 BGP table version is 39, local router ID is X.1.1.1, vrf id 0
Default local pref 100, local AS X00
...
   Network          Next Hop            Metric LocPrf Weight Path
*> ::/0             2001:db8:30:30::10
                                             0             0 10 i
*> 2001:db8:65:1::/64
                    2001:db8:30:30::65
                                             0             0 65 i
*> 2001:db8:bbbb:1::/64
                    2001:db8:30:30::10
                                             0             0 10 i
*> 2001:db8:bbbb:2::/64
                    2001:db8:30:30::10
                                             0             0 10 i
*> 2001:db8:cccc:1::/64
                    2001:db8:30:30::11
                                             0             0 11 i
*> 2001:db8:cccc:2::/64
                    2001:db8:30:30::11
                                             0             0 11 i

Total number of prefixes 6
```

You should see the same routes being filtered from the other route server (R9) in the IXP (show bgp neighbors 2001:db8:30:30::9 filtered-routes).

Step 4: Create an outbound policy on R1

In the outbound direction, you should only send to your eBGP peers appropriate prefixes. Those are the prefixes that belong to your network and to your customer's networks. In this step you'll create an outbound policy on R1 towards the transit provider and the IXP peers.

Step 4.1: Create a prefix-list for your customers' prefixes

```
u0X-R1# conf
u0X-R1(config)# ipv6 prefix-list CUSTOMER-A-PREFIXES seq 5 permit 2001:db8:X40:1::/64
u0X-R1(config)# ipv6 prefix-list CUSTOMER-A-PREFIXES seq 10 permit 2001:db8:X40:2::/64
u0X-R1(config)# ipv6 prefix-list CUSTOMER-A-PREFIXES seq 15 permit 2001:db8:X40::/48
u0X-R1(config)# ipv6 prefix-list CUSTOMER-B-PREFIXES seq 5 permit 2001:db8:X50:1::/64
u0X-R1(config)# ipv6 prefix-list CUSTOMER-B-PREFIXES seq 10 permit 2001:db8:X50:2::/64
```

```
u0X-R1(config)# ipv6 prefix-list CUSTOMER-B-PREFIXES seq 15 permit 2001:db8:X50::/48
```

Step 4.2: Create a route-map

Create a route-map, OUTBOUND that only advertises your prefixes and your customers' prefixes.

```
u0X-R1(config)# route-map OUTBOUND permit 10
u0X-R1(config-route-map)# match ipv6 address prefix-list ASX00-PREFIXES
u0X-R1(config-route-map)# route-map OUTBOUND permit 20
u0X-R1(config-route-map)# match ipv6 address prefix-list CUSTOMER-A-PREFIXES
u0X-R1(config-route-map)# route-map OUTBOUND permit 30
u0X-R1(config-route-map)# match ipv6 address prefix-list CUSTOMER-B-PREFIXES
u0X-R1(config-route-map)# exit
```

Step 5: Apply the outbound policy to transit provider and IXP peers

```
u0X-R1(config)# router bgp X00
u0X-R1(config-router)# address-family ipv6 unicast
u0X-R1(config-router-af)# neighbor 2001:db8:X00:17::7 route-map OUTBOUND out
u0X-R1(config-router-af)# neighbor 2001:db8:30:30::8 route-map OUTBOUND out
u0X-R1(config-router-af)# neighbor 2001:db8:30:30::9 route-map OUTBOUND out
u0X-R1(config-router-af)# end
```

Step 6: Check BGP routes advertised to transit provider and IXP peers

```
u0X-R1# show bgp neighbors 2001:db8:X00:17::7 advertised-routes
BGP table version is 39, local router ID is X.1.1.1, vrf id 0
Default local pref 100, local AS X00
...
   Network          Next Hop            Metric LocPrf Weight Path
*> 2001:db8:X00::/48
                    ::                       0         32768 i
*> 2001:db8:X40::/48
                    ::                            100      0 X40 i
*> 2001:db8:X40:1::/64
                    ::                            100      0 X40 i
*> 2001:db8:X40:2::/64
                    ::                            100      0 X40 i

Total number of prefixes 4
```

```
u0X-R1# show bgp neighbors 2001:db8:30:30::8 advertised-routes
BGP table version is 39, local router ID is X.1.1.1, vrf id 0
Default local pref 100, local AS X00
...
   Network          Next Hop            Metric LocPrf Weight Path
```

```
*> 2001:db8:X00::/48
                    ::                        0         32768 i
*> 2001:db8:X40::/48
                    ::                              100      0 X40 i
*> 2001:db8:X40:1::/64
                    ::                              100      0 X40 i
*> 2001:db8:X40:2::/64
                    ::                              100      0 X40 i

Total number of prefixes 4
```

You should see the same routes being advertised to the other route server (R9) in the IXP (show bgp neighbors 2001:db8:30:30::9 advertised-routes).

At the moment, only your own prefixes and the prefixes from Customer A are advertised to the Transit and the IXP, because those are the only ones received by R1. In the next activity, you will configure the policies for Customer B that will also allow those routes to be advertised.

Activity 2.2: Filtering with customers

The **inbound filtering policy** with end customers is straightforward: "**Accept only customer prefixes and discard the rest".**

So, if you know the prefixes of your customers and their customers (this might be based on information you get directly from your customer, IRR and/or RPKI data), then you can apply an inbound policy accordingly.

But, in case the customer advertises too many prefixes and/or there is not a reliable knowledge of all prefixes originated from the customer and their customers, then generic filters can be used to filter the following prefixes:

- Prefixes that are not globally routable.
- Prefixes not allocated by IANA (IPv6 only).
- Routes that are too specific.
- Prefixes belonging to the local AS, IXP LAN prefixes, and the default route.

However, the **outbound filtering policy** with customers may vary according to the routes the customer wants to receive. If the customer is expecting only the default route, this can be done easily by applying a filter allowing the default route only. In case the customer wants to receive the full routing (if it is multihomed or if they want to have a view of the Internet table), a policy can be applied on the BGP peering to filter prefixes that are not globally routable, routes that are too specific, and the default route.

In this lab activity, you'll configure both inbound and outbound filters with Customer B. **Customer B** wants to receive the **full Internet routing** table. Also, it is assumed that customer prefixes are known. So, you'll create an inbound filter to accept only customer prefixes from your Customer B, and will discard the rest.

Customer A has a stub network and **asks for a default route only**. Everything is already configured for this customer.

Customer A Prefixes:

- 2001:db8:X40::/48
- 2001:db8:X40:1::/64
- 2001:db8:X40:2::/64

Customer B Prefixes:

- 2001:db8:X50::/48
- 2001:db8:X50:1::/64
- 2001:db8:X50:2::/64

Step 1: Check your eBGP session with customer routers

Make sure that the eBGP sessions in R3 with Customer B (router R5) is up and running. Do you receive any routes from your customer?

```
u0X-R3# show bgp summary

IPv6 Unicast Summary (VRF default):
BGP router identifier X.3.3.3, local AS number X00 vrf-id 0
BGP table version 15
RIB entries 36, using 6912 bytes of memory
Peers 4, using 2868 KiB of memory

Neighbor           V         AS   MsgRcvd   MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd   PfxSnt Desc
2001:db8:X00::1    4        X00       186       179        0    0    0 02:56:59           11        1 N/A
2001:db8:X00::2    4        X00       181       181        0    0    0 02:56:58            3        1 N/A
2001:db8:X00:35::5 4        X50       182       182        0    0    0 02:57:12     (Policy) (Policy) N/A
2001:db8:X00:36::6 4        X60       181       180        0    0    0 02:57:13            1 (Policy) N/A

Total number of neighbors 4
```

The BGP session is up, but no prefixes received or advertised from/to the customer B router yet, and you see **(Policy)** in PfxRcd and PfxSnt counters. The reason for this is that no inbound or outbound policy has been configured on the eBGP session with the customer yet.

FRR open-source routing software supports RFC 8212 - Default External BGP (eBGP) Route Propagation Behavior without Policies. So according to this policy, BGP speakers do not accept or send routes on eBGP sessions, unless specifically configured to do so.

Step 2: Create an outbound policy with customers

Step 2.1: On R3, configure an outbound policy for Customer B

R3 will be configured with an outbound policy which advertises any route to Customer B's router, R5. In case the full routing table is sent to a customer, some prefixes such as default route (if it's not desired by the customer), too-specific prefixes and the ones that are not globally routable (BOGONs) can be filtered.

Create a route-map on R3 to filter the default route (::/0) and specific prefixes (smaller than /48 for IPv6) from being advertised to Customer B. However, do not filter BOGONs***. Because in our lab setup, IPv6 documentation prefix is used which is from the reserved space and one of the BOGON prefixes.***

```
u0X-R3# conf
u0X-R3(config)# ipv6 prefix-list ANY seq 10 permit any
u0X-R3(config)# ipv6 prefix-list DEFAULT-v6 seq 5 permit ::/0
u0X-R3(config)# ipv6 prefix-list TOO-SPECIFIC-v6 seq 5 permit ::/0 ge 49
```

```
u0X-R3(config)# route-map CUSTOMER-B-OUT deny 10
u0X-R3(config-route-map)# match ipv6 address prefix-list DEFAULT-v6
u0X-R3(config-route-map)# route-map CUSTOMER-B-OUT deny 20
u0X-R3(config-route-map)# match ipv6 address prefix-list TOO-SPECIFIC-v6
u0X-R3(config-route-map)# route-map CUSTOMER-B-OUT permit 40
u0X-R3(config-route-map)# match ipv6 address prefix-list ANY
u0X-R3(config-route-map)# exit
```

The last step is to apply the route-map as an outbound policy for the neighbour R5.

```
u0X-R3(config)# router bgp X00
u0X-R3(config-router)# address-family ipv6 unicast
u0X-R3(config-router-af)# neighbor 2001:db8:X00:35::5 route-map CUSTOMER-B-OUT out
u0X-R3(config-router-af)# end
```

Step 2.2: On R3, check advertised routes to Customer B

```
u0X-R3# show bgp neighbors 2001:db8:X00:35::5 advertised-routes
BGP table version is 15, local router ID is x.3.3.3, vrf id 0
Default local pref 100, local AS X00
...
   Network          Next Hop            Metric LocPrf Weight Path
*> 2001:db8::/32    ::                       100      0 20 i
*> 2001:db8:66::/48 ::                       100      0 66 i
*> 2001:db8:X00::/48
                    ::                       100      0 i
```

```
*> 2001:db8:X40::/48
                    ::                          100      0 X40 i
*> 2001:db8:X60::/48
                    ::                                   0 X60 i
*> 2001:db8:bbbb::/48
                    ::                          100      0 10 i
*> 2001:db8:cccc::/48
                    ::                          100      0 11 i
*> 2001:ff65::/32   ::                          100      0 65 i
*> 2001:ff75::/32   ::                          100      0 65 i
*> 2001:ff85::/32   ::                          100      0 65 i
*> 2001:ffc1::/32   ::                          100      0 11 23456 5000 5000 4000 i
*> 2001:ffc2::/32   ::                          100      0 11 23456 5000 5000 4000 i
...
```

You should see that too-specific prefixes, smaller than a /48, and the default route (::/0) are filtered out and not advertised to Customer B.

Note that you may also see other participant's prefixes being sent to the customer, because they are received by R1 and distributed to R3 internally (iBGP).

Step 3: Create an inbound policy with Customer B

Step 3.1: On R3, create an inbound BGP policy for Customer B prefixes

On R3, create an IPv6 prefix-list to only accept customers' prefixes from Customer B (R5) router.

```
u0X-R3# conf
u0X-R3(config)# ipv6 prefix-list CUSTOMER-B-PREFIXES seq 5 permit 2001:db8:X50:1::/64
u0X-R3(config)# ipv6 prefix-list CUSTOMER-B-PREFIXES seq 10 permit 2001:db8:X50:2::/64
u0X-R3(config)# ipv6 prefix-list CUSTOMER-B-PREFIXES seq 15 permit 2001:db8:X50::/48
```

On R3, create a route-map, CUSTOMER-B-IN, to allow only the routes in the prefix-list created and also set a community X00:101

```
u0X-R3(config)# route-map CUSTOMER-B-IN permit 10
u0X-R3(config-route-map)# match ipv6 address prefix-list CUSTOMER-B-PREFIXES
u0X-R3(config-route-map)# set community X00:101
```

Step 3.2: Apply this inbound BGP policy to the eBGP session with Customer B

```
U0X-R3(config)# router bgp X00
u0X-R3(config-router)# address-family ipv6 unicast
u0X-R3(config-router-af)# neighbor 2001:db8:X00:35::5 route-map CUSTOMER-B-IN in
u0X-R3(config-router-af)# end
```

Step 3.3: Check received routes from Customer B

Run "show bgp neighbors XXX received-routes" command on R3 to see the prefixes received from customers' router R5.

We can see many prefixes announced by R5, in addition to Customer B's prefixes:

```
u0X-R3# show bgp neighbors 2001:db8:X00:35::5 received-routes
BGP table version is 21, local router ID is X.3.3.3, vrf id 0
Default local pref 100, local AS X00
...
   Network          Next Hop            Metric LocPrf Weight Path
*> 2001:db8::/32    2001:db8:X00:35::5
                                                      0 X50 X00 20 i
*> 2001:db8:66::/48 2001:db8:X00:35::5
                                                      0 X50 X00 66 i
*> 2001:db8:X00::/48
                    2001:db8:X00:35::5
                                                      0 X50 X00 i
*> 2001:db8:X40::/48
                    2001:db8:X00:35::5
                                                      0 X50 X00 X40 i
*> 2001:db8:X50::/48
                    2001:db8:X00:35::5
                                               0      0 X50 i
*> 2001:db8:X50:1::/64
                    2001:db8:X00:35::5
                                               0      0 X50 i
*> 2001:db8:X50:2::/64
                    2001:db8:X00:35::5
                                               0      0 X50 i
*> 2001:db8:bbbb::/48
                    2001:db8:X00:35::5
                                                      0 X50 X00 10 i
*> 2001:db8:cccc::/48
                    2001:db8:X00:35::5
                                                      0 X50 X00 11 i
*> 2001:ff65::/32   2001:db8:X00:35::5
                                                      0 X50 X00 65 i
*> 2001:ff75::/32   2001:db8:X00:35::5
                                                      0 X50 X00 65 i
*> 2001:ff85::/32   2001:db8:X00:35::5
                                                      0 X50 X00 65 i
*> 2001:ffc1::/32   2001:db8:X00:35::5
                                                      0 X50 X00 11 23456 5000 5000 4000 i
*> 2001:ffc2::/32   2001:db8:X00:35::5
                                                      0 X50 X00 11 23456 5000 5000 4000 i

Total number of prefixes 15 (12 filtered)
```

If we look at what is being filtered out by the configured filters, we can see that it's filtering everything apart from Customer B's prefixes.

```
u0X-R3# show bgp neighbors 2001:db8:X00:35::5 filtered-routes
BGP table version is 21, local router ID is X.3.3.3, vrf id 0
Default local pref 100, local AS X00
...
   Network          Next Hop            Metric LocPrf Weight Path
*> 2001:db8::/32    2001:db8:X00:35::5
                                                      0 X50 X00 20 i
*> 2001:db8:66::/48 2001:db8:X00:35::5
                                                      0 X50 X00 66 i
*> 2001:db8:X00::/48
                    2001:db8:X00:35::5
```

```
                                                                    0 X50 X00 i
*> 2001:db8:X40::/48
                    2001:db8:X00:35::5
                                                                    0 X50 X00 X40 i
*> 2001:db8:X60::/48
                    2001:db8:X00:35::5
                                                                    0 X50 X00 X60 i
*> 2001:db8:bbbb::/48
                    2001:db8:X00:35::5
                                                                    0 X50 X00 10 i
*> 2001:db8:cccc::/48
                    2001:db8:X00:35::5
                                                                    0 X50 X00 11 i
*> 2001:ff65::/32   2001:db8:X00:35::5
                                                                    0 X50 X00 65 i
*> 2001:ff75::/32   2001:db8:X00:35::5
                                                                    0 X50 X00 65 i
*> 2001:ff85::/32   2001:db8:X00:35::5
                                                                    0 X50 X00 65 i
*> 2001:ffc1::/32   2001:db8:X00:35::5
                                                                    0 X50 X00 11 23456 5000 5000 4000 i
*> 2001:ffc2::/32   2001:db8:X00:35::5
                                                                    0 X50 X00 11 23456 5000 5000 4000 i

Total number of prefixes 12
```

In summary, you have implemented the recommendation to accept customers' prefixes for customer B in ASX00.

Activity 2.3: Troubleshooting BGP community filtering with a private peer

BGP communities are very powerful attributes when implementing routing policies. Using BGP community-based policies allows you to automatically advertise prefixes to upstream providers, IXs, or peers without having to update policies every time you connect a new eBGP customer.

In this scenario, you tag your prefixes received from your customers with a specific community X00:101 at the borders of your network and your own prefixes X00:100. You advertise only those routes to your private peer's router, R12. The outbound BGP policy for the private peer (ASX12) is already configured on R1 based on BGP communities. R1 should announce your own prefix (ASX00) as well as the prefixes from customers A, B, and C to your private peer (ASX12).

Communities
Your own prefixes - X00:100
Your customer's prefixes - X00:101

Customer C Prefix:
2001:db8:X60::/48

Check the existing configuration for BGP communities and the route-map. The configuration should appear similar to the following.

```
u0X-R1# show running-config
...
bgp community-list standard CUSTOMERS-PREFIXES seq 5 permit X00:101
bgp community-list standard OWN-PREFIXES seq 5 permit X00:100

...
route-map TO-PEERING-PARTNERS permit 20
 match community OWN-PREFIXES
exit
route-map TO-PEERING-PARTNERS permit 30
 match community CUSTOMERS-PREFIXES
exit
...
```

However, your peer (ASX12) has contacted you and asked you to check your side, because they are not receiving any prefixes from your customer C (ASX60).
In this lab activity, you will check the filtering based on BGP communities, troubleshoot the issue, and help your peer resolve the problem.

Step 1: On R1, check the prefixes advertised to your peer (ASX12) and the related BGP policy

```
u0X-R1# show bgp neighbors 2001:db8:X00:112::12 advertised-routes
...
   Network          Next Hop            Metric LocPrf Weight Path
*> 2001:db8:X00::/48
                    ::                       0         32768 i
*> 2001:db8:X40::/48
                    ::                            100      0 X40 i
*> 2001:db8:X50::/48
                    ::                            100      0 X50 i
Total number of prefixes 3
```

After checking on R1, you discover that the prefixes from your customer C (ASX60) are definitely not being announced to your peer (ASX12).

Step 2: On R1, check the communities on prefixes from customers A and C

```
u0X-R1# show bgp 2001:db8:X40::/48
BGP routing table entry for 2001:db8:X40::/48, version 40
...
      Community: X00:101
```

```
...
```

```
u0X-R1# show bgp 2001:db8:X60::/48
BGP routing table entry for 2001:db8:X60::/48, version 40
...
...
```

For the prefix from customer A, you see the community X00:101, however for customer C, you do not see it.

Step 3: On R3, verify and fix the incoming BGP policy for Customer C

```
u0X-R3# show running-config
...
route-map CUSTOMER-C-IN permit 10
 match ipv6 address prefix-list CUSTOMER-C-PREFIXES
exit
...
```

After checking the configuration on R3, you discover that your intern did not set the community X00:101 for incoming prefixes from Customer C in the route-map. You will fix this by adding the missing community.

```
u0X-R3# conf
u0X-R3(config)# route-map CUSTOMER-C-IN permit 10
u0X-R3(config-route-map)# set community X00:101
u0X-R3(config-route-map)# end
```

The community X00:101 has been added to the route-map for the incoming prefixes from Customer

Step 4: Check BGP routes on the private peer's router, R12

```
u0X-R12# show bgp neighbors 2001:db8:X00:112::1 received-routes
BGP table version is 6, local router ID is X.12.12.12, vrf id 0
Default local pref 100, local AS X12
...
   Network          Next Hop            Metric LocPrf Weight Path
*> 2001:db8:X60::/48
                    2001:db8:X00:112::1
                                                           0 X00 X60 i
...
```

You should now see the prefix from Customer C as well, because you have fixed the BGP community configuration on R3.
Your peer AS**X**12 is glad that you are now announcing all your and your customers' routes, improving Internet connectivity!

Lab Activity 3: Filtering AS-Path and Number of Prefixes

Activity 3.1: Filtering routes with AS-path filters

BGP routes can also be filtered based on AS path information. This method is widely used and highly scalable.

It is recommended that you filter BOGON Autonomous Systems Numbers (ASNs). BOGON ASNs are private and reserved ASNs that should never appear in the global routing table. You should reject routes that have BOGON ASNs anywhere in the AS Path. AS23456 (AS_TRANS) is one of these BOGON ASNs and therefore should be filtered.

In this lab activity, create an AS path filter to discard routes that have AS23456 (AS_TRANS) in the AS path and apply it to your eBGP session with your public peer, AS11.

Step 1: On R1, check your BGP table

```
u0X-R1# show bgp
BGP table version is 76, local router ID is X.1.1.1, vrf id 0
Default local pref 100, local AS X00
...
   Network          Next Hop            Metric LocPrf Weight Path
...

*  2001:ffc1::/32   fe80::a8c1:abff:fe8d:3baf
                                          0             0 11 23456 5000 5000 4000 i
*>                  fe80::a8c1:abff:fe8d:3baf
                                          0             0 11 23456 5000 5000 4000 i
*  2001:ffc2::/32   fe80::a8c1:abff:fe8d:3baf
                                          0             0 11 23456 5000 5000 4000 i
*>                  fe80::a8c1:abff:fe8d:3baf
                                          0             0 11 23456 5000 5000 4000 i
...
```

If you look at the AS PATHs, there are up to two routes, with two paths each one, that include the BOGON ASN 23456 (AS_TRANS).

Step 2: Create an AS PATH filter to discard routes with AS23456

```
u0X-R1# conf
u0X-R1(config)# bgp as-path access-list AS-TRANS seq 5 deny _23456_
u0X-R1(config)# bgp as-path access-list AS-TRANS seq 10 permit .*
```

The above configuration filters out (denies) any AS PATH with AS23456 anywhere in the AS-PATH attribute and allows others.

Step 3: Apply the AS Path filter to eBGP sessions with IXP Route servers (R8 and R9)

```
u0X-R1(config)# router bgp X00
u0X-R1(config-router)# address-family ipv6 unicast
u0X-R1(config-router-af)# neighbor 2001:db8:30:30::8 filter-list AS-TRANS in
u0X-R1(config-router-af)# neighbor 2001:db8:30:30::9 filter-list AS-TRANS in
u0X-R1(config-router-af)# end
```

In the current configuration, route-map INBOUND is already applied to eBGP sessions with IXP route servers (R8 and R9). When the filter-list is applied together with the route-map configuration, both are active. When both a route-map and a filter-list are applied inbound on the same session, they are processed sequentially: the route-map runs first, and only the routes it permits are then passed to the filter-list. So, according to the configuration above, in addition to the filtered prefixes from Route Servers, the routes with AS23456 (AS TRANS) in the AS Path will be discarded as well.

Step 4: On R1, check your BGP table again

```
u0X-R1# show bgp
```

You should see that prefixes with AS23456 in the AS PATH do not exist anymore.
The same principle can be applied to filter out the BOGON ASNs we showed in the course.

Activity 3.2: Limiting the number of accepted prefixes

Limiting the number of prefixes accepted on an eBGP session provides protection in case too many routes are sent by your upstream or by your peers. It is important to review the limits regularly and increase it if needed, since the number of the routes on the Internet grows very fast. Some vendors provide two thresholds to monitor this limit, while the first threshold will only trigger a log, the higher threshold will shut down BGP peering.

In this lab activity, configure the maximum number of prefixes that will be accepted from your IXP peers on the R1 router

Step 1: On R1, check the number of prefixes learned from IXP peers

```
u0X-R1# show bgp neighbors 2001:db8:30:30::8 routes
```

When you run the command above, you'll see several prefixes received from the IXP peer R8. You can do the same with R9 (2001:db8:30:30::9).

Step 2: Configure maximum prefix limit for Route Server R8

Configure a maximum prefix limit for the routes learned from the IXP Route Server R8. Here in the example, it is set to 3 because we have a small number of routes. To see the prefixes limit in action you need to use a number lower than the number of prefixes you saw in the previous step for R8.

```
u0X-R1# conf
u0X-R1(config)# router bgp X00
u0X-R1(config-router)# address-family ipv6
u0X-R1(config-router-af)# neighbor 2001:db8:30:30::8 maximum-prefix 3
u0X-R1(config-router-af)# end
```

Step 3: Check your BGP session with IXP Route Server R8

```
u0X-R1# show bgp summary neighbor 2001:db8:30:30::8

IPv6 Unicast Summary (VRF default):
BGP router identifier X.1.1.1, local AS number X00 vrf-id 0
...
Neighbor          V         AS   MsgRcvd   MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd   PfxSnt Desc
2001:db8:30:30::8 4         30      3587      3138        0    0    0 00:02:06 Idle (PfxCt)        0 N/A

Displayed neighbors 1
Total number of neighbors 6
```

You can see that the BGP session between R1 and R8 is in **Idle (PfxCt)** state, related to the prefix count. Let's see more details:

```
U0X-R1# show bgp neighbors 2001:db8:30:30::8
BGP neighbor is 2001:db8:30:30::8, remote AS 30, local AS X00, external link
  Local Role: undefined
  Remote Role: undefined
  BGP version 4, remote router ID 192.0.2.8, local router ID X.1.1.1
  BGP state = Idle
  Last read 00:01:36, Last write 00:01:27
.
.
For address family: IPv6 Unicast
.
  Incoming update prefix filter list is *ANY
  Incoming update AS path filter list is *AS-TRANS
  Route map for incoming advertisements is *INBOUND-IX
  Route map for outgoing advertisements is *OUTBOUND
  0 accepted prefixes
  Maximum prefixes allowed 3
```

```
  Threshold for warning message 75%

  Connections established 1; dropped 1
  Last reset 00:01:07,  Notification sent (Cease/Maximum Number of Prefixes Reached)
  Peer had exceeded the max. no. of prefixes configured.
  Reduce the no. of prefix and clear ip bgp 2001:db8:30:30::8 to restore peering
  External BGP neighbor may be up to 1 hops away.
...
```

It indicates that the reason for the Idle state is that the peer has exceeded the maximum number of configured prefixes (3).

Step 4: Remove the maximum-prefix limit configuration on R1

```
u0X-R1# conf
u0X-R1(config)# router bgp X00
u0X-R1(config-router)# address-family ipv6
u0X-R1(config-router-af)# no neighbor 2001:db8:30:30::8 maximum-prefix 3
u0X-R1(config-router-af)# end
```

Let's see if things return to the previous state after removing the prefixes limit.

Step 5: Check again the BGP session with IXP route server R8

```
u0X-R1# show bgp neighbors 2001:db8:30:30::8
BGP neighbor is 2001:db8:30:30::8, remote AS 30, local AS X00, external link
  Local Role: undefined
  Remote Role: undefined
  BGP version 4, remote router ID 192.0.2.8, local router ID X.1.1.1
  BGP state = Established, up for 00:01:28
  Last read 00:00:31, Last write 00:00:28
.
.
  Connections established 1; dropped 1
  Last reset 00:07:31,   Notification sent (Cease/Maximum Number of Prefixes Reached)
.
```

The BGP session with R8 is established again. If you check the received routes, you will see the previously received routes again. The effect of limiting the prefixes accepted from a peer was not to accept up to that number, but to disconnect the BGP session, so no routes are exchanged.

In a real environment, the limit number would be much larger, allowing the expected number of routes and allowing for some growth over time, but checking it regularly. The goal is to protect the BGP speaker resources from running out because of one BGP session with a sudden increase in the number of announced routes.

Lab Activity 4: Setting up BGP Roles and Exploring OTC

Activity 4.1: Configure BGP roles with your customers and configure a strict mode.

Step 1: On R3, check the eBGP session with Customer B (R5)

```
u0X-R3# show bgp neighbors 2001:db8:X00:35::5
BGP neighbor is 2001:db8:X00:35::5, remote AS X50, local AS X00, external link
  Local Role: undefined
  Remote Role: undefined
...
  BGP state = Established, up for 01:46:22
...
```

You can see that the BGP session is established, and the Local Role for R3 and the Remote Role for R5 are shown as undefined.

Step 2: Configure BGP role “provider” for eBGP session on R3 facing to Customer B (R5)

Configure a BGP role “provider” since your ASX00 is Upstream provider for your customer B with ASX50 (R5).

```
u0X-R3# conf
u0X-R3(config)# router bgp X00
u0X-R3(config-router)# neighbor 2001:db8:X00:35::5 local-role provider
u0X-R3(config-router)# end
```

Check your BGP session with your customer B with ASX50 (R5).

```
u0X-R3# show bgp neighbors 2001:db8:X00:35::5
BGP neighbor is 2001:db8:X00:35::5, remote AS X50, local AS X00, external link
  Local Role: provider
  Remote Role: undefined
...
  BGP state = Established, up for 00:00:08
...
```

This eBGP session was restarted in order to send a new BGP OPEN message with the ‘role’ capability.

Step 3: Configure a BGP role “provider” for the eBGP session on R5 (Customer B) facing your router R3

Configure a BGP role “provider” on R5

```
u0X-R5# conf
u0X-R5(config)# router bgp X50
u0X-R5(config-router)# neighbor 2001:db8:X00:35::3 local-role provider
u0X-R5(config-router)# end
```

Check this eBGP session on R5

```
u0X-R5# show bgp neighbors 2001:db8:X00:35::3
BGP neighbor is 2001:db8:X00:35::3, remote AS X00, local AS X50, external link
  Local Role: provider
  Remote Role: undefined
...
  BGP state = Idle
...
 Last reset 00:00:13,  Notification sent (OPEN Message Error/Role Mismatch)
```

Please note that the BGP session changes to the Idle state because both R3 and R5 are configured with the same role “provider” on both sides. The notification message also helps identify the issue as a Role Mismatch.
We intentionally configured R5 incorrectly - for **R5** the Local Role should be “customer”.

Step 4: Configure a BGP role “customer” for the eBGP session on R5 (Customer B) facing your router R3

Configure the BGP role “customer” on R5 because your AS**X**00 is the upstream provider for Customer B (AS**X**50).

```
u0X-R5# conf
u0X-R5(config)# router bgp X50
u0X-R5(config-router)# neighbor 2001:db8:X00:35::3 local-role customer
u0X-R5(config-router)# end
```

To avoid waiting for the “BGP connect retry timer” to expire, on **R3** use the command below before checking this eBGP session.

```
u0X-R3# clear bgp 2001:db8:X00:35::5
u0X-R3# show bgp neighbors 2001:db8:X00:35::5
BGP neighbor is 2001:db8:X00:35::5, remote AS X50, local AS X00, external link
  Local Role: provider
  Remote Role: customer
...
  BGP state = Established, up for 00:00:07
...
```

You can see that the eBGP session is established again and also you see the status of Local Role - provider and the status of Remote Role - customer. These two roles correspond, that's why the BGP session is established.

You can also check this session on router R5. You should see the Local Role - “customer” and the Remote Role - “provider”.

Step 5: Remove a BGP role for the eBGP session on R5 (Customer B) facing your router R3

Remove the BGP role “customer” on R5.

```
u0X-R5# conf
u0X-R5(config)# router bgp X50
u0X-R5(config-router)# no neighbor 2001:db8:X00:35::3 local-role customer
u0X-R5(config-router)# end
```

Check this eBGP session on R3

```
u0X-R3# show bgp neighbors 2001:db8:X00:35::5
BGP neighbor is 2001:db8:X00:35::5, remote AS X50, local AS X00, external link
  Local Role: provider
  Remote Role: undefined
...
  BGP state = Established, up for 00:00:06
...
```

You can see that the eBGP session has been reset, the Local Role is set to provider and the Remote Role is undefined.
Despite this, the session is still successfully established. If one side does not have a role configured, the session will be established by default.
This is the default behavior described in RFC 9234 (when strict mode is not used). Let’s see what happens when the strict mode is used.

Step 6: Configure eBGP role with a strict mode for eBGP session on R3 facing your Customer B (R5)

Configure a BGP role “provider” with a strict mode.

```
u0X-R3# conf
u0X-R3(config)# router bgp X00
u0X-R3(config-router)# neighbor 2001:db8:X00:35::5 local-role provider strict-mode
u0X-R3(config-router)# end
```

Check this eBGP session on R3.

```
u0X-R3# show bgp neighbors 2001:db8:X00:35::5
BGP neighbor is 2001:db8:X00:35::5, remote AS X50, local AS X00, external link
```

```
  Local Role: provider
  Remote Role: undefined
...
  BGP state = Idle
...
  Last reset 00:00:10,  Notification sent (OPEN Message Error/Role Mismatch)
...
```

This eBGP session changes to the Idle state because on R3 the Local Role is configured as “provider” with strict mode enabled, while on R5 no Local Role is configured. When strict mode is used, the remote side must also configure a corresponding BGP role. Otherwise, the eBGP session will not be established.

Step 7: Correct BGP role on R5 (Customer B) to match strict mode on R3

Let's imagine that you are now the customer of R5, and you will configure a Local Role on R5 to fix the issue.
Configure a BGP role “customer” because AS**X**50 (R5) has a connection to its upstream provider AS**X**00 (your R3):

```
u0X-R5# conf
u0X-R5(config)# router bgp X50
u0X-R5(config-router)# neighbor 2001:db8:X00:35::3 local-role customer
u0X-R5(config-router)# end
```

If you don’t want to wait for a timer, please clear this BGP session on **R3.**

```
u0X-R3# clear bgp 2001:db8:X00:35::5
```

Then check the session on R3:

```
u0X-R3# show bgp neighbors 2001:db8:X00:35::5
BGP neighbor is 2001:db8:X00:35::5, remote AS X50, local AS X00, external link
  Local Role: provider
  Remote Role: customer
...
  BGP state = Established, up for 00:00:10
...
```

The BGP session has been established because the customer (R5) now has a BGP Role configured on their side.

Activity 4.2: Make route-leaks from IX and from upstream to a private peer.

Step 1: Configure current route-maps

We will make a configuration mistake on purpose that will cause route leaks to peer ASX12. In real-world situations, this can happen due to various scenarios.

```
u0X-R1# conf
u0X-R1(config)# route-map INBOUND-TRANSIT permit 60
u0X-R1(config-route-map)# set community X00:101
u0X-R1(config-route-map)# route-map INBOUND-IX permit 60
u0X-R1(config-route-map)# set community X00:101
u0X-R1(config-route-map)# end
```

Step 2: Check BGP routes on the private peer's router, R12

```
u0X-R12# show bgp neighbors 2001:db8:X00:112::1 routes
BGP table version is 6, local router ID is X.12.12.12, vrf id 0
Default local pref 100, local AS X12
...
   Network          Next Hop            Metric LocPrf Weight Path
*> 2001:db8::/32    fe80::a8c1:abff:fee6:e643
                                                           0 X00 20 i
*> 2001:db8:66::/48 fe80::a8c1:abff:fee6:e643
                                                           0 X00 66 i
*> 2001:db8:X00::/48
                    fe80::a8c1:abff:fee6:e643
                                             0             0 X00 i
*> 2001:db8:X40::/48
                    fe80::a8c1:abff:fee6:e643
                                                           0 X00 X40 i
*> 2001:db8:X50::/48
                    fe80::a8c1:abff:fee6:e643
                                                           0 X00 X50 i
*> 2001:db8:X60::/48
                    fe80::a8c1:abff:fee6:e643
                                                           0 X00 X60 i
*> 2001:db8:bbbb::/48
                    fe80::a8c1:abff:fee6:e643
                                                           0 X00 10 i
*> 2001:db8:cccc::/48
                    fe80::a8c1:abff:fee6:e643
                                                           0 X00 11 i
*> 2001:ff65::/32   fe80::a8c1:abff:fee6:e643
                                                           0 X00 65 i
*> 2001:ff75::/32   fe80::a8c1:abff:fee6:e643
                                                           0 X00 65 i
*> 2001:ff85::/32   fe80::a8c1:abff:fee6:e643
                                                           0 X00 65 i
```

You should see prefixes from AS20 and from IX and also from your customers. Also check in detail prefixes which you receive on R12

```
u0X-R12# show bgp 2001:db8::/32
BGP routing table entry for 2001:db8::/32, version 90
...
  X00 20
  ...
    Origin IGP, valid, external, otc 20, best (First path received)
...

u0X-R12# show bgp 2001:db8:bbbb::/48
...
 X00 10
...
      Origin IGP, valid, external, best (First path received)
...
```

You can see that the prefix 2001:db8::/32 from AS20 has an OTC attribute because the AS20 (Provider) installed the BGP role “provider” on the BGP session facing the ASX00 (R1). On IX nobody installed a BGP role on BGP sessions from Route Servers (R8 & R9) facing to ASX00 (R1) and you don’t see OTC attribute on the prefix 2001:db8:bbbb::/48.

Step 3: Configure BGP role on R12 facing to R1

```
u0X-R12# conf
u0X-R12#(config)# router bgp X12
u0X-R12#(config-router)# neighbor 2001:db8:X00:112::1 local-role peer
u0X-R12#(config-router)# end
```

We installed the BGP role - peer. Let's check which routes learned from ASX00 (R1)

```
u0X-R12# show bgp neighbors 2001:db8:X00:112::1 routes
BGP table version is 6, local router ID is X.12.12.12, vrf id 0
Default local pref 100, local AS X12
...
   Network          Next Hop            Metric LocPrf Weight Path
*> 2001:db8:66::/48 fe80::a8c1:abff:fee6:e643
                                                      0 X00 66 i
*> 2001:db8:X00::/48
                    fe80::a8c1:abff:fee6:e643
                                         0            0 X00 i
*> 2001:db8:X40::/48
                    fe80::a8c1:abff:fee6:e643
                                                      0 X00 X40 i
*> 2001:db8:X50::/48
```

```
                    fe80::a8c1:abff:fee6:e643
                                                          0 X00 X50 i
*> 2001:db8:X60::/48
                    fe80::a8c1:abff:fee6:e643
                                                          0 X00 X60 i
*> 2001:db8:bbbb::/48
                    fe80::a8c1:abff:fee6:e643
                                                          0 X00 10 i
*> 2001:db8:cccc::/48
                    fe80::a8c1:abff:fee6:e643
                                                          0 X00 11 i
*> 2001:ff65::/32   fe80::a8c1:abff:fee6:e643
                                                          0 X00 65 i
*> 2001:ff75::/32   fe80::a8c1:abff:fee6:e643
                                                          0 X00 65 i
*> 2001:ff85::/32   fe80::a8c1:abff:fee6:e643
                                                          0 X00 65 i
```

Why don't you see the prefix 2001:db8::/32?

When the ASX12 receives a prefix 2001:db8::/32 with OTC and has a BGP role on the link with the ASX00, the R12 can detect a route leak because R12 is waiting for all prefixes with OTC X00.

Thus, if the BGP role was installed correctly on routers R1 or on IX, these route leaks from IX through ASX00 to ASX12 could be detected on routers R12 or R1 and fixed automatically by BGP too.

Let’s set BGP roles with eBGP peers on R1 during the next activity 4.3.

Activity 4.3: Configure BGP roles with your upstream, IX, private peer and check again these route-leaks.

Step 1: Check the prefixes which R1 announces to R12

```
u0X-R1# show bgp neighbors 2001:db8:X00:112::12 advertised-routes
...
   Network          Next Hop            Metric LocPrf Weight Path
*> 2001:db8::/32    ::                                   0 20 i
*> 2001:db8:66::/48 ::                                   0 66 i
*> 2001:db8:X00::/48
                    ::                       0         32768 i
*> 2001:db8:X40::/48
                    ::                            100      0 X40 i
*> 2001:db8:X50::/48
                    ::                            100      0 X50 i
*> 2001:db8:X60::/48
                    ::                            100      0 X60 i
*> 2001:db8:bbbb::/48
                    ::                                     0 10 i
*> 2001:db8:cccc::/48
```

```
                    ::                                      0 11 i
*> 2001:ff65::/32   ::                                      0 65 i
*> 2001:ff75::/32   ::                                      0 65 i
*> 2001:ff85::/32   ::                                      0 65 i
...
```

Step 2: Configure BGP roles for eBGP sessions on R1

```
u0X-R1# conf
u0X-R1(config)# router bgp X00
u0X-R1(config-router)# neighbor 2001:db8:X00:17::7 local-role customer
u0X-R1(config-router)# neighbor 2001:db8:30:30::8 local-role rs-client
u0X-R1(config-router)# neighbor 2001:db8:30:30::9 local-role rs-client
u0X-R1(config-router)# neighbor 2001:db8:X00:112::12 local-role peer
u0X-R1(config-router)# end
```

Step 3: Check the BGP roles for eBGP session R1-R12

```
u0X-R1# show bgp neighbors 2001:db8:X00:112::12
BGP neighbor is 2001:db8:X00:36::6, remote AS X60, local AS X00, external link
  Local Role: peer
  Remote Role: peer
...
  BGP state = Established, up for 00:00:45
...
```

Step 4: Check OTC attribute for prefixes from upstream and IX

```
u0X-R1# show bgp 2001:db8::/32
BGP routing table entry for 2001:db8::/32, version 113
...
  20
  ...
    Origin IGP, valid, external, otc 20, best (First path received)
...

u0X-R1# show bgp 2001:db8:bbbb::/48
...
 10
...
      Origin IGP, metric 0, valid, external, otc 30, best (Older Path)
...
```

You should see the OTC attribute for these prefixes with different OTC values. Why is the OTC 30 for 2001:db8:bbbb::/48 and not the OTC 10?

Because R1 received one of these prefixes from the ISP (AS20) and the other from the IXP router server (AS30), which is why you see these OTCs.

Step 5: Check the prefixes which R1 announces to R12 again

```
u0X-R1# show bgp neighbors 2001:db8:X00:112::12 advertised-routes
...
   Network          Next Hop            Metric LocPrf Weight Path
*> 2001:db8:X00::/48
                    ::                       0         32768 i
*> 2001:db8:X40::/48
                    ::                            100      0 X40 i
*> 2001:db8:X50::/48
                    ::                            100      0 X50 i
*> 2001:db8:X60::/48
                    ::                            100      0 X60 i
...
```

You should not see that R1 announces prefixes from IX and from Upstream to Peer R12.

The configuration of BGP roles, together with OTC, helps to automatically mitigate route leaks, even in the case of a misconfigured BGP policy on R1 towards its peer R12.

Lab Activity 5: RPKI

In the following lab activities we will see both sides of RPKI: The authorisation by signing Route Origin Authorisations (ROAs) and the validation by configuring Route Origin Validation (ROV).

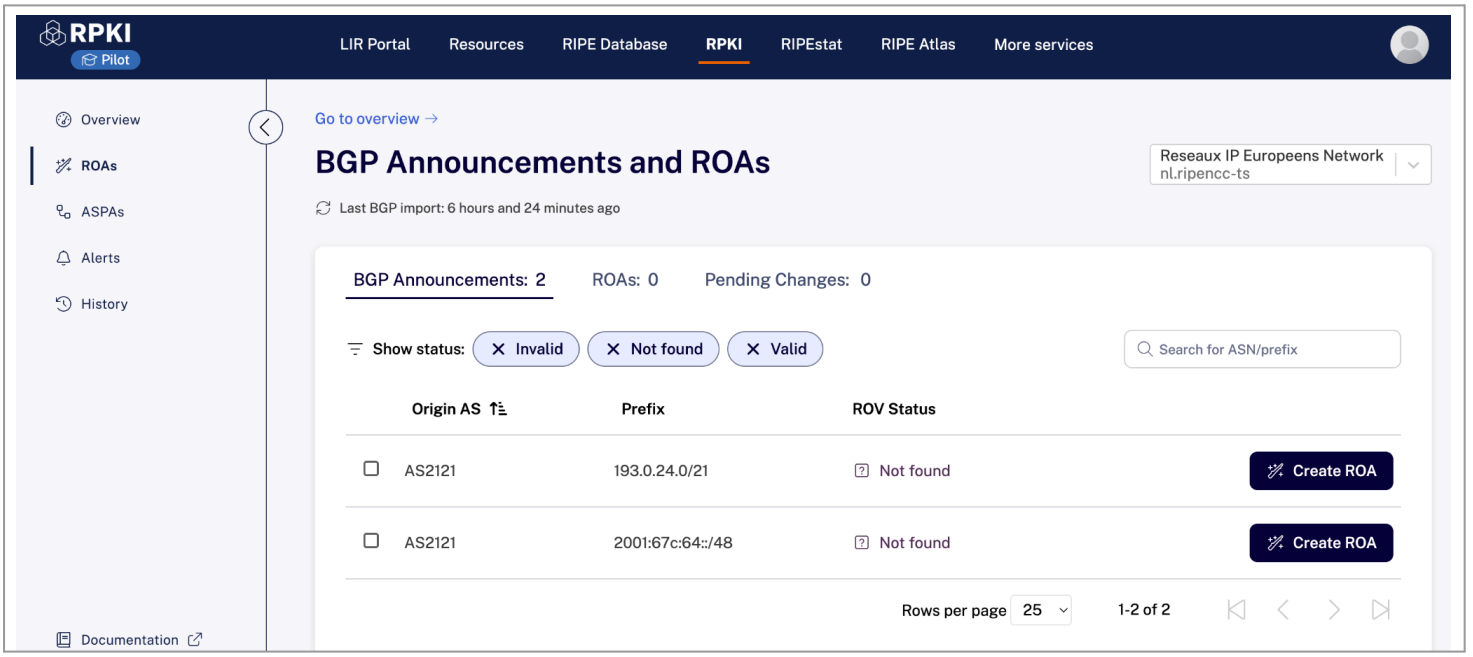
Activity 5.1: Creating ROAs

In this lab activity, you are going to create ROAs for **your LIR's prefixes** in a **test environment.** Log in to **the test RPKI Dashboard** with your LIR account, if you don't remember your LIR account login, the trainer will also provide a demo by logging in to the LIR Portal and creating ROAs for the nl.ripencc-ts LIR account.

On the left hand side, click on "Resource Certification" and then "Dashboard".

Pilot RPKI Dashboard: https://dashboard.rpki.localcert.ripe.net/

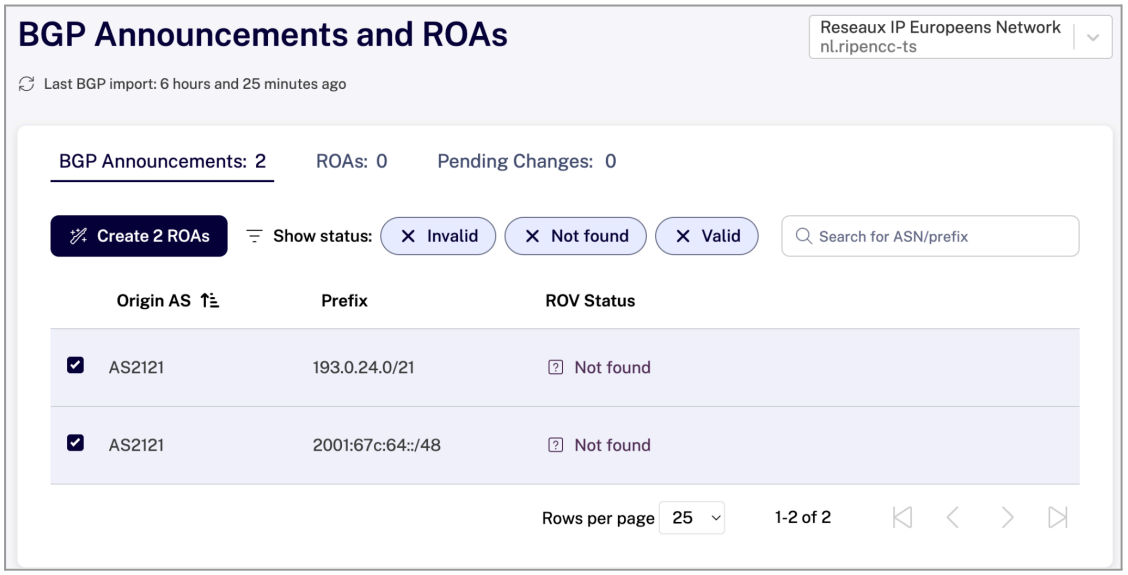
Step 1: Check your BGP announcements

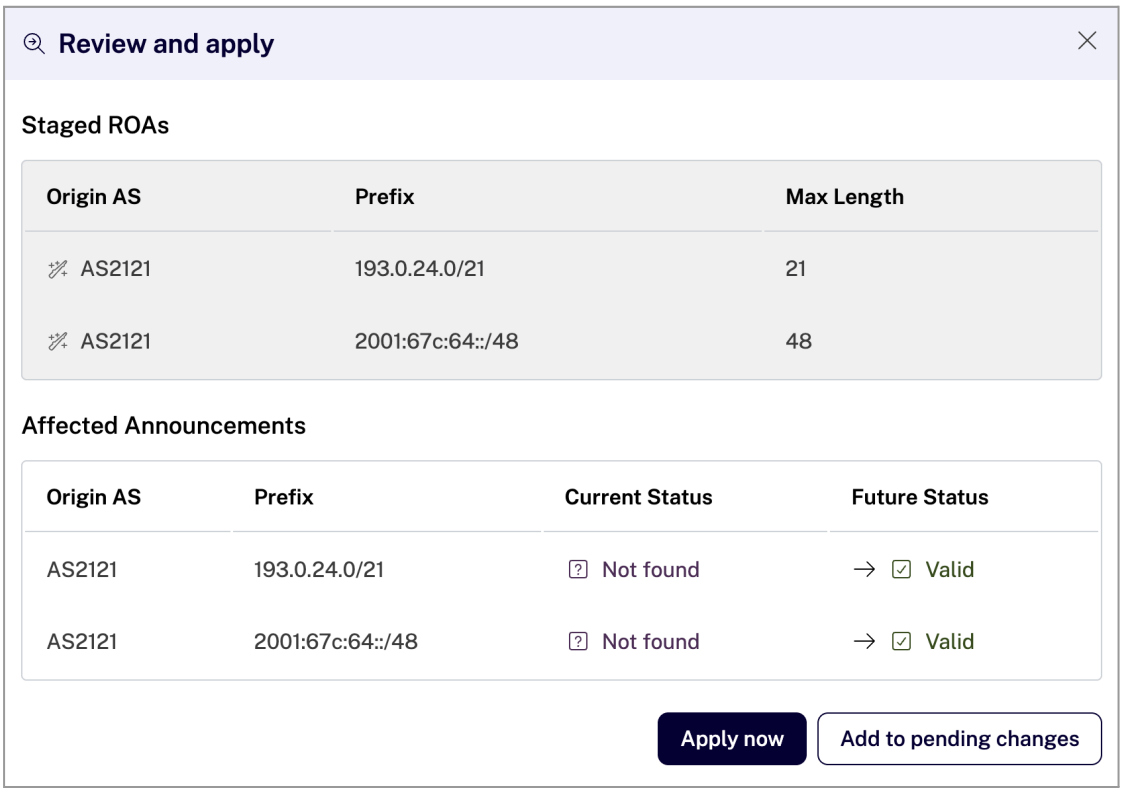


The RPKI dashboard shows the prefixes collected by RIPE NCC's Routing Information Service (RIS). In the example image, you can see the announcements for the RIPE NCC's AS2121. You will see your LIR's prefixes in your output.

Step 2: Creating ROAs

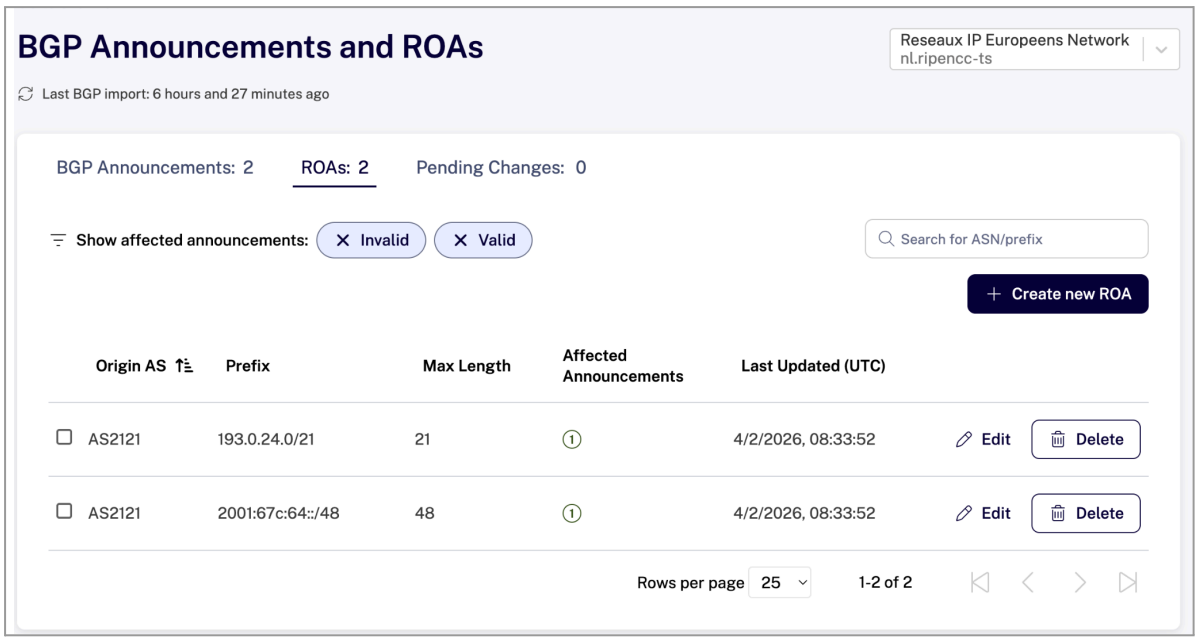
In the RPKI dashboard, you can select the BGP announcements you want to create ROAs for, and then click on “Create ROAs”.





You can review the changes before finally publishing the ROAs by clicking on “Apply now”. In the example, the two prefixes will change from Unknown to Valid.

Step 3: Verify that ROAs have been created

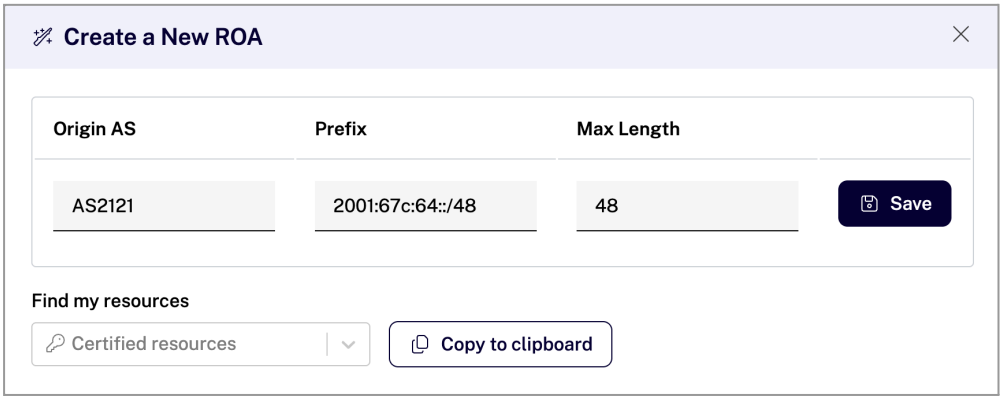


By default, it creates ROAs for the exact prefix, so the more specific length allowed is the same as the prefix length.

Step 4: Create a more specific ROA for one of your IPv4/IPv6 prefixes

Pick one of **your LIR's prefixes** and decide on a more specific one (i.e /24 or even /25 for IPv4, and /36 or /40 for IPv6). Then, create a ROA with your assigned AS number, AS**X**00, as the origin AS.

You can do this using the "+ Create New ROA" button, then saving, and finally reviewing and publishing changes.



You have now published ROAs for all or some of your prefixes in the RIPE NCC repository. You are protecting your address space from origin hijacks, but that protection will only happen in networks using BGP and RPKI to validate their routes. This is called BGP Route Origin Validation. You will configure ROV in the next lab activity.

Activity 5.2: BGP Origin Validation

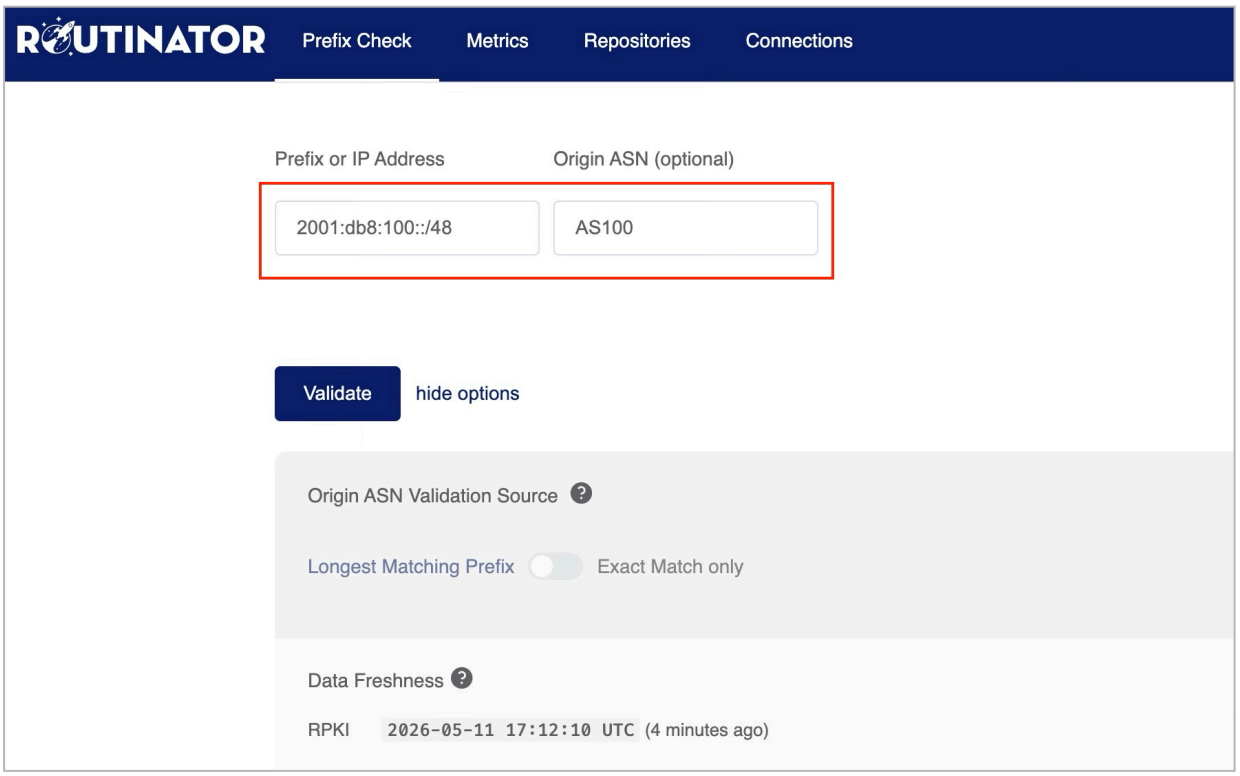
BGP Origin Validation (BGP OV) is BGP filtering using the information in ROAs. It is used to validate the origin of BGP announcements by using RPKI infrastructure. This verifies if the network originating a prefix is really authorised to originate it in BGP or not.

In this lab activity, you will implement BGP Origin Validation and discard BGP announcements with “Invalid” RPKI status.

The **Routinator** validator is preinstalled and pre-configured, and already running in your network. It connects to the RPKI test repository via RRDP and downloads the ROAs which are pre-created and signed for in all users’ networks. Once the RPKI data is downloaded, the validator verifies the signatures on all objects and outputs the valid route origins as a list. Each object in this list contains an IP prefix, a maximum length, and an origin AS number. This object is referred to as **Validated ROA Payload (VRP)**. The collection of VRPs is known as the **validated cache.**

Step 1: Check valid ROAs on Routinator’s GUI

Click “**Validator UI**” in the lab dashboard to see if a valid ROA exists for your resources, which authorises your ASN to originate your prefix in BGP. Use your IPv6 address block (2001:db8:X00::/48) and your assigned ASN (X00). Click “Validate”.





The output shows that the valid ROA for AS100 prefix exists for your allocated IPv6 address block and the assigned ASN.

Step 2: Connect your validator (Routinator) to router R1

In our lab setup, only one validator is available at IPv6 address 2001:db8:X00:1f::f. In a production network, however, it is recommended to deploy at least two validators, because the validation state may change to NotFound for all routes if RPKI-enabled routers lose connectivity to the validators. As a result, routes that should be rejected may instead be accepted.

Here the polling_period timer tells the router how long to wait before the next attempt to poll the validator (RPKI cache). In this lab, set the timer to 3,600 seconds, which is the default value recommended by RFC 8210.

```
u0X-R1# conf
u0X-R1#(config)# rpki
u0X-R1(config-rpki)# rpki polling_period 3600
u0X-R1(config-rpki)# rpki cache 2001:db8:X00:1f::f 3323 preference 1
u0X-R1(config-rpki)# end
```

Step 3: Display the connection status to the validator

```
u0X-R1# show rpki cache-connection
Connected to group 1
rpki tcp cache 2001:db8:X00:1f::f 3323 pref 1 (connected)
```

This command displays which validators are connected to the router using the RTR protocol.

Step 4: Check the RPKI prefix table on R1

```
u0X-R1# show rpki prefix-table
RPKI/RTR prefix table
Prefix                                  Prefix Length  Origin-AS
10.3.1.0                                   24 -  24          203
10.4.1.0                                   24 -  24          204
10.1.1.0                                   24 -  24          201
...
2001:ff03::                                32 -  32          103
2001:ff16::                                32 -  32          116
2001:db8:412::                             48 -  48          412
2001:db8:440::                             48 -  48          440
2001:db8:460::                             48 -  48          460
2001:db8:500::                             48 -  48          500
2001:db8:540::                             48 -  48          540
2001:db8:550::                             48 -  48          550
2001:db8:560::
...
Number of IPv4 Prefixes: 90
Number of IPv6 Prefixes: 259
```

This command displays the RPKI prefix table which contains the VRPs, the validated prefixes received from the validator.

Step 5: Create a BGP hijack

To help you understand how BGP origin validation works, you will now simulate a hijack. Announce IPv6 address block of User(**X + 1**). Remember that the prefix 2001:db8:**X**00::/48 is allocated to user **X**. As more participants do this step, you'll see more and more hijacked prefixes in your BGP table.

```
u0X-R1# conf
u0X-R1#(config)# ipv6 route 2001:db8:(X+1)00::/48 null0
u0X-R1#(config)# ipv6 prefix-list ASX00-PREFIXES seq 10 permit 2001:db8:(X+1)00::/48
u0X-R1#(config)# router bgp X00
u0X-R1#(config-router)# address-family ipv6
u0X-R1#(config-router-af)# network 2001:db8:(X+1)00::/48
u0X-R1#(config-router-af)# end
```

Step 6: Check the validation result

Check your BGP table and see that all the hijacked prefixes are tagged with "Invalid" RPKI status. A BGP route will be tagged with **RPKI Invalid status** unless it's originated by an ASN which is authorised to do so and is registered in the RPKI system with a ROA.

In addition to RPKI Invalids, you'll see BGP routes with "Valid" and "Not Found" RPKI status:

```
u0X-R1# show bgp
BGP table version is 21, local router ID is X.1.1.1, vrf id 0
Default local pref 100, local AS X00
...
RPKI validation codes: V valid, I invalid, N Not found

   Network          Next Hop            Metric LocPrf Weight Path
...
N*> ::/0             fe80::a8c1:abff:fe43:29e9
                                         0             0 20 i
N*> 2001:db8::/32    fe80::a8c1:abff:fe43:29e9
                                         0             0 20 i
N*  2001:db8:66::/48 fe80::a8c1:abff:fe43:29e9
                                                       0 20 66 i
...

V*> 2001:db8:X00::/48
                     ::                  0         32768 i
V*>i2001:db8:X40::/48
                     2001:db8:X00::2     0    100      0 X40 i
I*>i2001:db8:X40:1::/64
                     2001:db8:X00::2     0    100      0 X40 i
...

V*  2001:db8:(X+1)00::/48
                     fe80::a8c1:abff:fee3:c0b3
                                         0             0 (X+1)00 i
V*                   fe80::a8c1:abff:fee3:c0b3
                                         0             0 (X+1)00 i
V*                   fe80::a8c1:abff:fe63:41f7
                                                       0 20 (X+1)00 i
I*>                  ::                   0         32768 i

...
```

However, as you have not configured a filter to drop RPKI Invalids yet, this route is still taken into consideration for BGP path selection. It may be chosen as the best route and actively used for traffic forwarding. Let's see how to discard Invalid BGP routes.

Step 7: Discard BGP RPKI invalids

Update the existing route-map "INBOUND" that has been created in the previous activities. Add a new "deny" line which discards "RPKI Invalid BGP routes" and accept all prefixes that have RPKI "Valid" and "Not Found" states and have passed previous controls.

You should not discard RPKI Not Found BGP routes. RPKI "Not Found" state indicates that there is no ROA for a certain prefix in the RPKI repositories. ROAs do not exist for more than two thirds of all prefixes on the Internet. If you therefore drop BGP announcements with RPKI

"Not Found" state, you will lose access to a huge part of the Internet, so you need to accept them.

```
u0X-R1# conf
u0X-R1(config)# route-map INBOUND-IX deny 50
u0X-R1(config-route-map)# match rpki invalid
u0X-R1(config)# route-map INBOUND-TRANSIT deny 50
u0X-R1(config-route-map)# match rpki invalid
u0X-R1(config-route-map)# end
```

Make sure the changes apply to the BGP sessions:

```
u0X-R1# clear bgp 2001:db8:X00:17::7 in
u0X-R1# clear bgp 2001:db8:30:30::8 in
u0X-R1# clear bgp 2001:db8:30:30::9 in
```

Step 8: Check R1 BGP table again

The BGP routes with RPKI Invalid status (which have been learned from the transit provider and the IXP route servers) do not appear in the BGP table anymore. You only see Valid, Not Found, and Invalid for the prefixes that you're the origin of the hijack or coming from other peers.

```
u0X-R1# show bgp
...
RPKI validation codes: V valid, I invalid, N Not found

   Network          Next Hop            Metric LocPrf Weight Path
N*  2001:db8:66::/48 fe80::a8c1:abff:fe05:df70
                                              0             0 66 i
N*>                  fe80::a8c1:abff:fe05:df70
                                              0             0 66 i
N*                   fe80::a8c1:abff:fe63:41f7
                                                            0 20 66 i
V*> 2001:db8:X00::/48
                     ::                       0         32768 i
V*>i2001:db8:X40::/48
                     2001:db8:X00::2          0    100      0 X40 i
...

V*  2001:db8:(X+1)00::/48
                     fe80::a8c1:abff:fee3:c0b3
                                              0             0 (X+1)00 i
V*                   fe80::a8c1:abff:fee3:c0b3
                                              0             0 (X+1)00 i
V*                   fe80::a8c1:abff:fe63:41f7
                                                            0 20 (X+1)00 i
...
```

Note that, while it is recommended in many places to set higher local-preference to RPKI Valids, and lower local-preference to RPKI Unknowns, there are some concerns with this approach. It is considered harmful to manipulate BGP Path Attributes (for example LOCAL_PREF or COMMUNITY) based on the RPKI Origin Validation state.
Making BGP Path Attributes dependent on RPKI Validation states can introduce needless brittleness in the global routing system.