Engineering
Simplicity

Apstra ConnectorOps Setup Guide

Published
2025-10-28

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

Apstra ConnectorOps Setup Guide

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at https://support.juniper.net/support/eula/. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

About This Guide

This guide walks you through the setup and deployment of Apstra ConnectorOps. ConnectorOps pulls topology metadata and routing details from Apstra, then pushes configurations to your SRXs. ConnectorOps is a service that runs in a VM/host/server with access to both Apstra and the Juniper SRXs. The service continuously monitors Apstra Blueprints containing Generic Systems with the tag "srx" for CRUD events. When an event occurs, ConnectorOps automatically pushes the latest configurations to the SRXs. This ensures that your SRXs always immediately enforce the latest routing configurations.

1

CHAPTER

Introduction

IN THIS CHAPTER

Introduction

Apstra ConnectorOps leverages Juniper Apstra's intent-based networking (IBN) features to automate the configuration of Juniper SRX firewalls. The service continuously monitors Apstra blueprints for any CRUD operations—from initial deployment to ongoing device changes—and automatically translates these updates into SRX configurations using Netconf over SSH.

ConnectorOps pulls topology metadata, routing instances, and BGP peering details from Apstra, then renders corresponding SRX Junos configurations. You provide SRX device credentials, VRF details, and high-availability (HA) loopback information through Property Sets. With this input, ConnectorOps automatically generates and pushes routing configurations to the SRX. Changes to your fabric are immediately reflected in the appropriate firewall configuration.

2

CHAPTER

Prerequisites

IN THIS CHAPTER

Prerequisites

There are many ways to set up a topology to work with Apstra ConnectorOps. For a step-by-step guide on how to set up the exact topology used in this guide, see "Set Up the Example ConnectorOps Topology" on page 29.

If you are using your own topology, the following prerequisites are required for Apstra ConnectorOps to identify SRX attached blueprints in Apstra and generate configurations.

> **NOTE**: Currently, for MNHA topologies, ConnectorOps supports up to two SRXs. The SRXs can be physical SRXs, or vSRXs.

Blueprint prerequisites:

- DC fabric with border leaf connected to SRX in Multinode High Availability (MNHA) mode is onboarded to Apstra as Blueprint.
- Loopback IP and ASN’s assigned to both the SRX devices
- Device Profile created, and assigned Interface Map to SRX’s. For more information, see "Create a Device Profile and Assign an Interface Map to the SRX" on page 41.
- Connectivity Template created between SRX and borderleaf with appropriate IPLink, BGP Peer, and Routing Policy details. For more information, see "To Create and Assign a Connectivity Template Between the SRXs and Border Leaves" on page 30.
- Create DCI/Over the Top or External Gateway between SRX and border leaf. For more information, see "To Create External (Over the Top) Gateways in Apstra, and Designate them as Firewalls for Each SRX" on page 33.
- User-created Routing Zones, assigned to appropriate nodes
- Virtual Networks created with associated routing zones, and assigned to hosts. For more information, see "To Create VNs and Assign Routing Zones to Nodes" on page 34.

Apstra ConnectorOps prerequisites:

- Both SRX’s onboarded to Apstra Blueprint as Generic Systems with tag “srx”
- Import Global Property Set with the following information: SRX credentials, VRF details, `commitConfig` onto Blueprint. This will be used as input by ConnectorOps
- The VM/host/server where ConnectorOps is installed has access to both Apstra and the SRXs
- Netconf over SSH is enabled on SRXs

3

CHAPTER

Supported Topologies

IN THIS CHAPTER

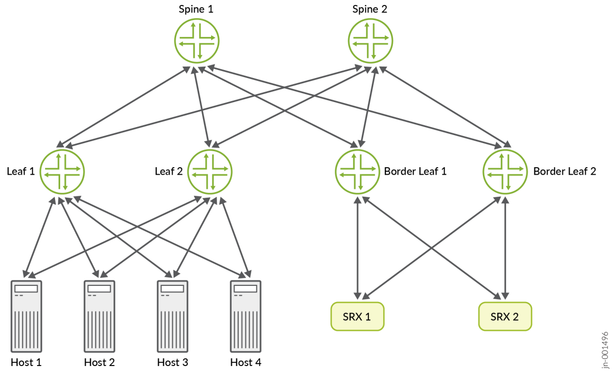
Supported Topologies

Apstra ConnectorOps supports non-HA (single SRX hanging from border leaf) or Multinode High Availability (MNHA) (two peer SRXs) mode topologies.

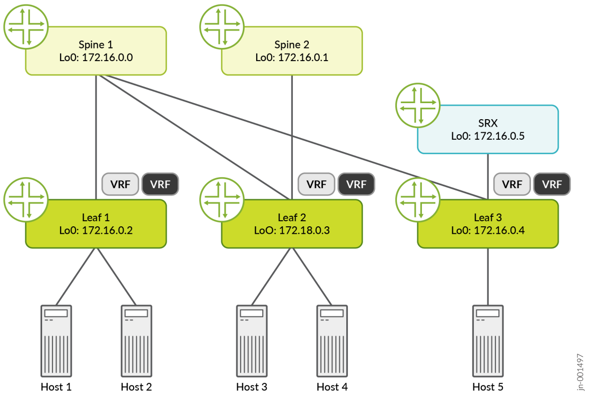
MNHA mode topology

This MNHA topology shows a dual border-leaf design with two SRX Firewalls for high-availability. Each SRX connects to a both border leaves, to provide fabric redundancy.

Traffic from local hosts moves through leaf/spine layers to the border, then passes through either SRX for inspection and policy enforcement. The redundant paths ensure that the fabric can keep routing and enforcing security policies even if a border leaf or SRX fails.



Non-HA topology



In this topology, the EVPN-VXLAN fabric is designed for edge-routed bridging (ERB), with SRXs functioning in an enhanced border leaf (EBL) position. EBL extends the traditional border leaf role to provide traffic inspection for VXLAN tunnels at the network edge.

VXLAN traffic originating at the LEAF-1 device traverses through the SRX Series Firewalls that function as EBLs. In this use case, the SRX Series Firewall is placed at the border, that is, at the entry and exit point of the campus or data centre, to provide stateful inspection to the VXLAN encapsulated packets traversing through it.

Host-1 is mapped to VN-Blue with its own VRF, and Host-4 to VN-Green with an isolated VRF.

When pinging from Host-1 to Host-4 (or the reverse), the traffic is routed from Host-1 to LEAF-1, lands on LEAF-3 via the spine layer, and establishes a VXLAN tunnel between LEAF-1 and LEAF-3. Upon reaching LEAF-3, the traffic is automatically directed to the SRX for policy enforcement and inter-VRF route-leak inspection. Following inspection, the traffic is returned through LEAF-3, forwarded to LEAF-2 via the spines, and ultimately lands Host-4.

4

CHAPTER

Deploy ConnectorOps and Verify Connectivity

IN THIS CHAPTER

Deploy Apstra ConnectorOps and Verify Connectivity

NOTE: In this example, we install ConnectorOps in a separate VM with access to both Apstra and the SRXs. When installing ConnectorOps, ensure that the VM/host/server where ConnectorOps is installed can ping the Apstra VM and the SRXs.

NOTE: ConnectorOps does not automatically create Security Policies on SRX devices. For example Security Policies used in this exact topology, see "Configure Security Policies" on page 37.

1. Create a directory for the ConnectorOps image.

   ```
   mkdir ~/connectorops
   ```

2. Download the latest Apstra ConnectorOps image onto the VM/host/server.

   Use the following download link under Application Tools: https://support.juniper.net/support/downloads/?p=apstra.

3. Install Docker and Docker Compose.

4. Load the Docker image.

   ```
   root@test-vm:~/connectorops$ docker load -i apstra-connectorops-x86_64-6.0.0.image.tgz
   f52af47487c7:
   f52af47487c7: Loading layer [==================================================>]  327.7kB/
   327.7kB
   10e9b9e72178: Loading layer [==================================================>]  40.96kB/
   40.96kB
   9ed5d5f339b1: Loading layer [==================================================>]  2.396MB/
   2.396MB
   ff5700ec5418: Loading layer [==================================================>]  10.24kB/
   10.24kB
   d52f02c6501c: Loading layer [==================================================>]  10.24kB/
   10.24kB
   e624a5370eca: Loading layer [==================================================>]  10.24kB/
   10.24kB
   1a73b54f556b: Loading layer [==================================================>]  10.24kB/
   ```

```
10.24kB
8f2b2d741d53: Loading layer [==================================================>]  10.24kB/
10.24kB
9959f8de3336: Loading layer [==================================================>]  245.8kB/
245.8kB
54a7f7f831d9: Loading layer [==================================================>]  13.06MB/
13.06MB
4abacc923ad4: Loading layer [==================================================>]  5.908MB/
5.908MB
d46a475bb22c: Loading layer [==================================================>]  17.85MB/
17.85MB
Loaded image: apstra-connectorops:6.0.0-1.0.19
```

5. Create the `docker-compose.yml` file.

   The following is an example `docker-compose.yml` file. Note the line `"<host-port>:8080"`. This is the host-side port. When you run `docker compose`, the host port is exposed and any REST API calls listen to this port, which is redirected to the ConnectorOps service port (8080).

```
services:
  connector-ops:
    image: <image-name>:<tag>
    container_name: connector-ops
    environment:
      - AOS_URL=<aos-url>
      - AOS_USER=<aos-username>
      - AOS_PASSWORD=<aos-password>
      - CONFIGURE_SRX=enable
    ports:
      - "<host-port>:8080"
    volumes:
      - <path-to-host-directory>:/app/data
    restart: always
```

6. Start ConnectorOps.

```
root@test-vm:~/connectorops$ docker-compose up -d
```

   The service begins generating configs for your Blueprint.

After starting ConnectorOps, proceed to "Create and Import a Global Property Set" on page 12.

5

CHAPTER

Create and Import a Property Set

IN THIS CHAPTER

Create and Import a Global Property Set

SUMMARY

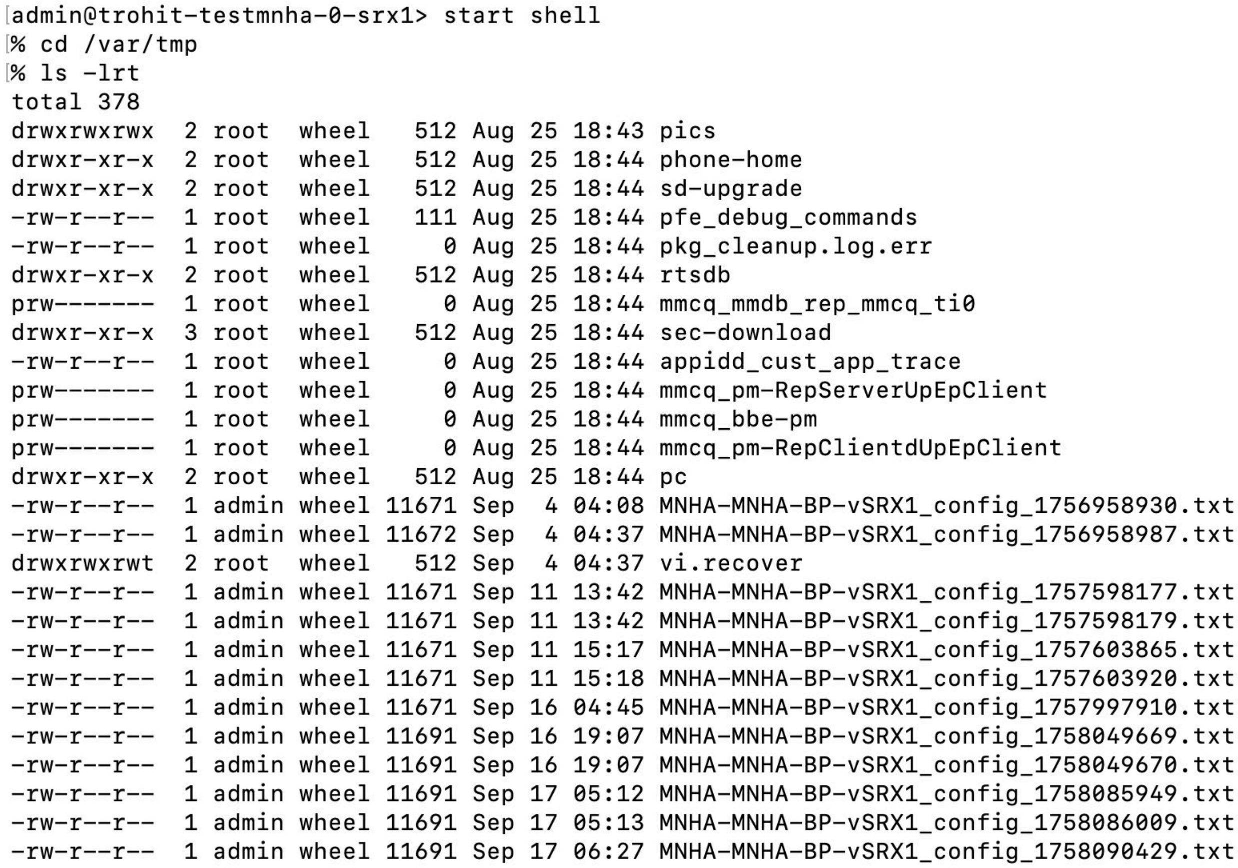
Follow these steps to create and import a Global Property Set. Apstra ConnectorOps uses this Property Set as input to render SRX configurations.

You must define a Property Set with input information for ConnectorOps to automate configuration generation to the SRXs. ConnectorOps uses this Property Set as the data source for configuration rendering and to establish connectivity between devices.

You must configure a Property Set with the "**connector_ops_config**" prefix. The property set must include the following:

- **"SRX_details"**: SRX device details
- **"commitConfig"**: A Boolean flag (true, false). If set to “true”, ConnectorOps pushes the configuration and commits it on the SRX. If set to “false”, ConnectorOps saves the config file on the SRX but doesn’t commit, so that you can review the file before committing. The config file is saved in the following path: `/var/tmp/MNHA-<blueprint_name>-<device_name>_config_<timestamp>.txt`.

  The config files are located in `/var/tmp`.



```
[admin@trohit-testmnha-0-srx1> start shell
[% cd /var/tmp
[% ls -lrt
 total 378
 drwxrwxrwx  2 root  wheel   512 Aug 25 18:43 pics
 drwxr-xr-x  2 root  wheel   512 Aug 25 18:44 phone-home
 drwxr-xr-x  2 root  wheel   512 Aug 25 18:44 sd-upgrade
 -rw-r--r--  1 root  wheel   111 Aug 25 18:44 pfe_debug_commands
 -rw-r--r--  1 root  wheel     0 Aug 25 18:44 pkg_cleanup.log.err
 drwxr-xr-x  2 root  wheel   512 Aug 25 18:44 rtsdb
 prw-------  1 root  wheel     0 Aug 25 18:44 mmcq_mmdb_rep_mmcq_ti0
 drwxr-xr-x  3 root  wheel   512 Aug 25 18:44 sec-download
 -rw-r--r--  1 root  wheel     0 Aug 25 18:44 appidd_cust_app_trace
 prw-------  1 root  wheel     0 Aug 25 18:44 mmcq_pm-RepServerUpEpClient
 prw-------  1 root  wheel     0 Aug 25 18:44 mmcq_bbe-pm
 prw-------  1 root  wheel     0 Aug 25 18:44 mmcq_pm-RepClientdUpEpClient
 drwxr-xr-x  2 root  wheel   512 Aug 25 18:44 pc
 -rw-r--r--  1 admin wheel 11671 Sep  4 04:08 MNHA-MNHA-BP-vSRX1_config_1756958930.txt
 -rw-r--r--  1 admin wheel 11672 Sep  4 04:37 MNHA-MNHA-BP-vSRX1_config_1756958987.txt
 drwxrwxrwt  2 root  wheel   512 Sep  4 04:37 vi.recover
 -rw-r--r--  1 admin wheel 11671 Sep 11 13:42 MNHA-MNHA-BP-vSRX1_config_1757598177.txt
 -rw-r--r--  1 admin wheel 11671 Sep 11 13:42 MNHA-MNHA-BP-vSRX1_config_1757598179.txt
 -rw-r--r--  1 admin wheel 11671 Sep 11 15:17 MNHA-MNHA-BP-vSRX1_config_1757603865.txt
 -rw-r--r--  1 admin wheel 11671 Sep 11 15:18 MNHA-MNHA-BP-vSRX1_config_1757603920.txt
 -rw-r--r--  1 admin wheel 11671 Sep 16 04:45 MNHA-MNHA-BP-vSRX1_config_1757997910.txt
 -rw-r--r--  1 admin wheel 11691 Sep 16 19:07 MNHA-MNHA-BP-vSRX1_config_1758049669.txt
 -rw-r--r--  1 admin wheel 11691 Sep 16 19:07 MNHA-MNHA-BP-vSRX1_config_1758049670.txt
 -rw-r--r--  1 admin wheel 11691 Sep 17 05:12 MNHA-MNHA-BP-vSRX1_config_1758085949.txt
 -rw-r--r--  1 admin wheel 11691 Sep 17 05:13 MNHA-MNHA-BP-vSRX1_config_1758086009.txt
 -rw-r--r--  1 admin wheel 11691 Sep 17 06:27 MNHA-MNHA-BP-vSRX1_config_1758090429.txt
```

- **"sharedLoopbackIP"**: Shared loopback IP address between peer SRX.

- **"vrf"**: VRF assignment details for route leaking and policy configuration.

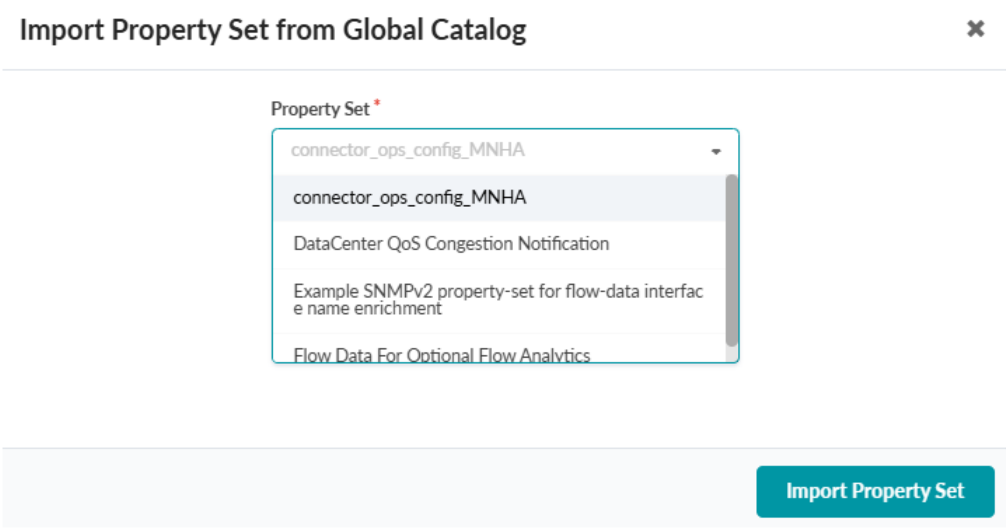
The following is an example Property Set:

```
{
  "SRX_details": [
    {
      "managementIP": "5d11s7.englab.juniper.net:38601",
      "name": "static-cops-03-0-srx1",
      "password": "admin@123",
      "username": "admin"
    },
    {
      "managementIP": "5d11s7.englab.juniper.net:30737",
      "name": "static-cops-03-0-srx2",
      "password": "admin@123",
      "username": "admin"
    }
  ],
  "commitConfig": true,
  "sharedLoopbackIP": "172.27.1.0",
  "vrf": {
    "animals": [
      {
        "name": "cats"
      },
      {
        "name": "dogs"
      }
    ],
    "color": [
      {
        "advertise_routes": [
          "210.210.210.0/24"
        ],
        "name": "green"
      },
      {
        "advertise_routes": [
          "200.200.200.0/24"
        ],
        "name": "red"
      }
    ]
  }
}
```

For `managementIP`, specify the SRX management address as either an IP or hostname. If SSH access uses a non-default port, specify it as `host:port` (for example, `192.0.2.1:2222`). ConnectorOps uses port 22 by default if no port is provided.

1. Navigate to **Design** > **Property Sets** > **Create Property Set**.
   The Create Property Set window displays.
2. Enter a **Name** and define a Property Set configuration in Values.

   Make sure to follow the proper naming convention.
3. From within your Blueprint, navigate to **Staged** > **Catalog** > **Property Sets** > **Import Property Set**.
   The Import Property Set from Global Catalog window displays.
4. Select the Property Set you created from the dropdown, and click **Import Property Set**.



Import Property Set from Global Catalog

Property Set *

connector_ops_config_MNHA

- connector_ops_config_MNHA
- DataCenter QoS Congestion Notification
- Example SNMPv2 property-set for flow-data interface name enrichment
- Flow Data For Optional Flow Analytics

This triggers an event, and ConnectorOps uses the Property Set as input to generate the configuration.

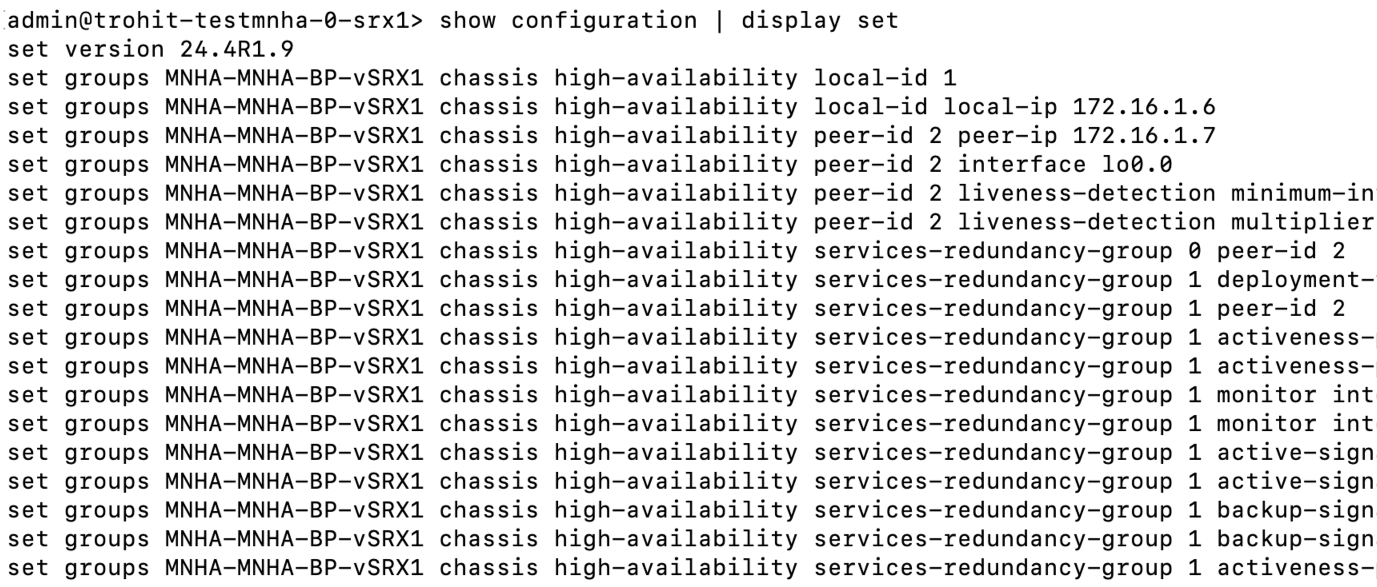
5. Verify that ConnectorOps has generated a Global Configlet by navigating to **Design** > **Configlets**.

   Configlets beginning with "**configlet**" prefixes are automatically generated by ConnectorOps during its operational workflow.

6. (Optional) Log into your SRX and verify that the custom configs showing Group names that match the generated configs are present.

   The group names should match the name of the Global Configlet.

```
show configuration | display set
```



```
admin@trohit-testmnha-0-srx1> show configuration | display set
set version 24.4R1.9
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability local-id 1
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability local-id local-ip 172.16.1.6
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability peer-id 2 peer-ip 172.16.1.7
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability peer-id 2 interface lo0.0
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability peer-id 2 liveness-detection minimum-in
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability peer-id 2 liveness-detection multiplier
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability services-redundancy-group 0 peer-id 2
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability services-redundancy-group 1 deployment-
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability services-redundancy-group 1 peer-id 2
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability services-redundancy-group 1 activeness-
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability services-redundancy-group 1 activeness-
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability services-redundancy-group 1 monitor int
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability services-redundancy-group 1 monitor int
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability services-redundancy-group 1 active-sign
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability services-redundancy-group 1 active-sign
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability services-redundancy-group 1 backup-sign
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability services-redundancy-group 1 backup-sign
set groups MNHA-MNHA-BP-vSRX1 chassis high-availability services-redundancy-group 1 activeness-
```

7. If `commitConfig="true"`, ConnectorOps pushes the configurations to the SRX and you must manually reboot the SRX for the HA configs to take effect.

   ```
   request system reboot
   ```

6

CHAPTER

Custom Configlets

IN THIS CHAPTER

Custom Configlets

You can apply custom configurations on top of base configurations generated by ConnectorOps using Configlet objects. Configlet objects define custom configs.

Name configlets according to this naming convention:

- Per device: `custom-<blueprintname>-<srx-device-name>-<optional-suffix>`

  You can have multiple configlets with different suffixes for a single device.

  custom-static-cops-04-0-static-cops-04-0-srx1-suffix

- Global level (all devices): `custom-<blueprintname>`
- If the SRX name is included at the end, the configlet targets only that specific SRX.
- If not, the configlet is applied to all SRX devices in the Blueprint.

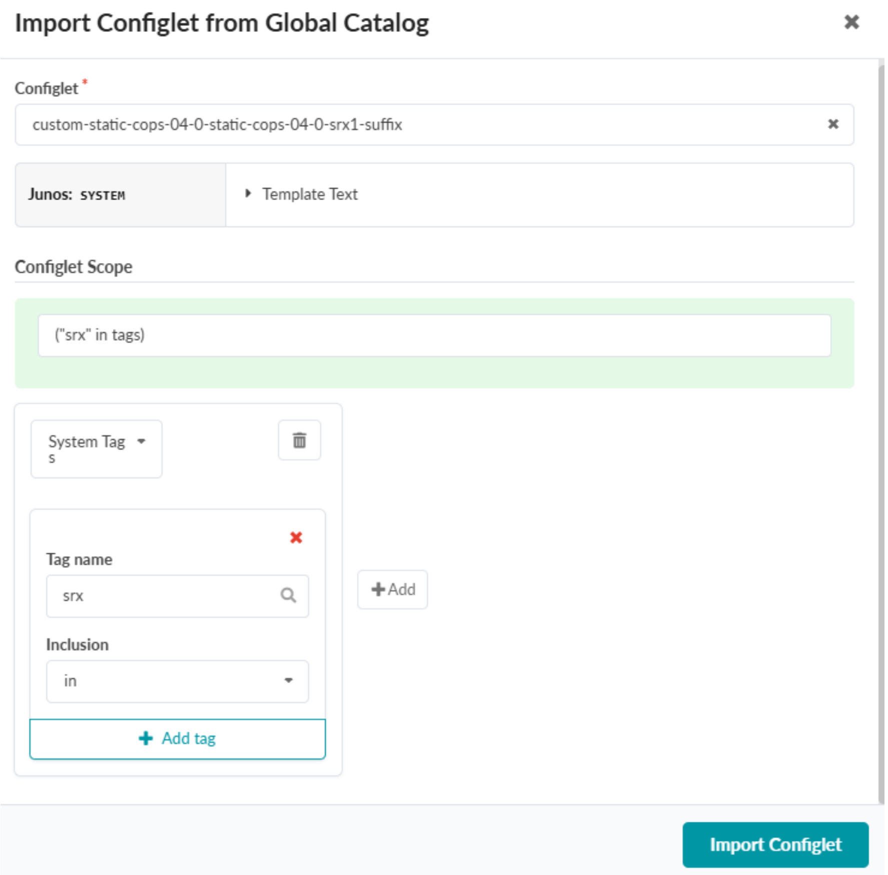
ConnectorOps supports both Hierarchical and Set commands during creation.

> **NOTE**: Custom configlets are only committed to the SRX if the `commitConfig` flag is "true".

If `commitConfig=false`, ConnectorOps does not save configurations to a file on the SRX. This is a "no action" for ConnectorOps.

After you create a custom Configlet at the Global scope, you must import it into your Blueprint at **Staged** > **Catalog** > **Configlets** > **Import Configlet**.

1. Select your custom Configlet from the dropdown.
2. Click the Role dropdown and select **System Tags**
3. Enter "**srx**" in the Tag name field.
4. Click **Import Configlet**.



The following is an example custom Configlet:

```
policy-options {
    route-filter-list BL1-pref {
        <example-subnet> exact;
    }
    route-filter-list BL2-pref {
        <example-subnet> exact;
    }
    policy-statement BL1-in {
        term 10 {
            from {
                route-filter-list BL1-pref;
            }
            then {
                local-preference 200;
                accept;
            }
        }
        term 20 {
            then {
                local-preference 100;
                accept;
            }
        }
    }
    policy-statement BL2-in {
        term 10 {
            from {
                route-filter-list BL2-pref;
            }
            then {
                local-preference 200;
                accept;
            }
        }
        term 20 {
            then {
                local-preference 100;
                accept;
            }
        }
    }
}
protocols {
    bgp {
        group EVPN-FABRIC {
            neighbor <example-IP> {
                import BL1-in;
            }
            neighbor <example-IP> {
                import BL2-in;
            }
        }
    }
}
```

7

CHAPTER

APIs

IN THIS CHAPTER

APIs

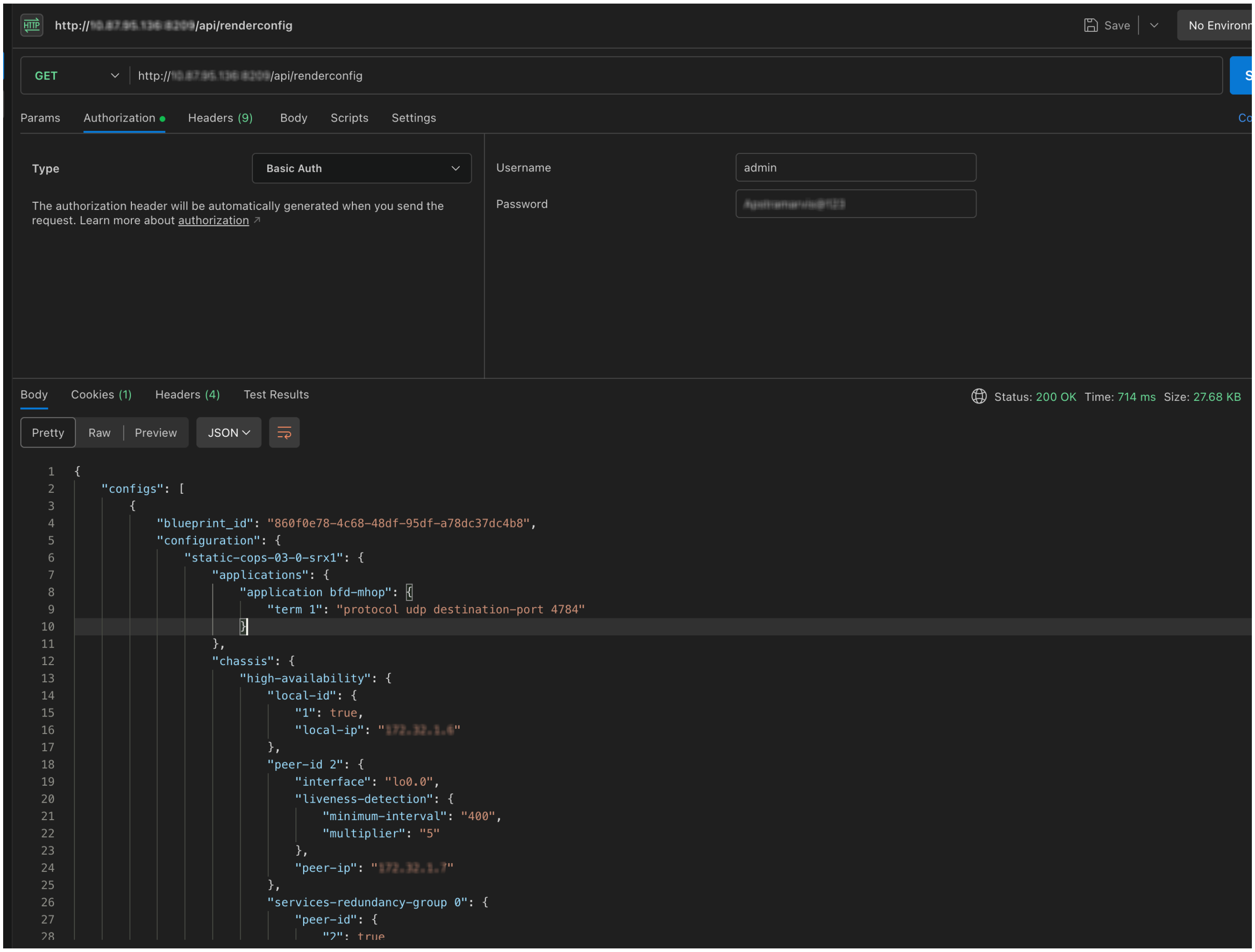
Apstra ConnectorOps exposes a REST API for pulling rendered SRX configurations and operational data. The API is exposed on the ConnectorOps VM/host/server at **http://<ConnectorOps-VM-IP:<VM-Port>/**. The VM port is the host-side port you specify in the `docker-compose.yml`. The following are the API endpoints:

NOTE: You must provide the username and password of your Apstra setup to authenticate ConnectorOps REST API calls.

- All Blueprints: **http://<ConnectorOps-VM-IP>:<VM-Port>/api/renderconfig**
- Per Blueprint: **http://<ConnectorOps-VM-IP:<VM-Port>/api/renderconfig/<BluePrintID>**
- Per device: **http://<ConnectorOps-VM-IP>:<VM-Port>/api/renderconfig/<BluePrintID>/device/<DeviceName>**

Response codes:

- 200: OK
- 401: Invalid username or password
- 500: For Invalid SRX Device Names and BluePrint ID



GET http://[redacted]/api/renderconfig

Type: Basic Auth

The authorization header will be automatically generated when you send the request. Learn more about authorization ↗

Username: admin

Password: [redacted]

Status: 200 OK Time: 714 ms Size: 27.68 KB

```
{
    "configs": [
        {
            "blueprint_id": "860f0e78-4c68-48df-95df-a78dc37dc4b8",
            "configuration": {
                "static-cops-03-0-srx1": {
                    "applications": {
                        "application bfd-mhop": {
                            "term 1": "protocol udp destination-port 4784"
                        }
                    },
                    "chassis": {
                        "high-availability": {
                            "local-id": {
                                "1": true,
                                "local-ip": "[redacted]"
                            },
                            "peer-id 2": {
                                "interface": "lo0.0",
                                "liveness-detection": {
                                    "minimum-interval": "400",
                                    "multiplier": "5"
                                },
                                "peer-ip": "[redacted]"
                            },
                            "services-redundancy-group 0": {
                                "peer-id": {
                                    "2": true
```

8

CHAPTER

Troubleshooting

IN THIS CHAPTER

Troubleshooting

Check the following to help with troubleshooting:

- Ensure that the SRX Generic Device is tagged as **"srx"** so ConnectorOps can process the blueprint and generate SRX configuration.
- Verify that all user-inputted Property Sets follow the correct naming conventions and are properly imported into the relevant blueprints.
- Confirm that both Apstra and the SRX devices are reachable from the VM/host/server where ConnectorOps is installed.
- Check that the ConnectorOps Docker process is up and running. Review logs for any errors using:

```
docker logs <container_name_or_id>
```

RELATED DOCUMENTATION

Apstra User Guide

9

CHAPTER

Compatibility

IN THIS CHAPTER

Compatibility

The following hardware and software versions are validated versions that Juniper has tested to work with Apstra ConnectorOps.

- SRX JUNOS version 23.4 R2 and higher
- Midrange SRX devices: SRX1500, SRX1600, SRX2300, SRX4100, SRX4120, SRX4200, SRX4300, SRX4600, and SRX4700
- High end SRX devices: SRX5000
- Juniper Apstra version 6.0 or later

10

CHAPTER

References

IN THIS CHAPTER

Set Up the Example ConnectorOps Topology

IN THIS SECTION

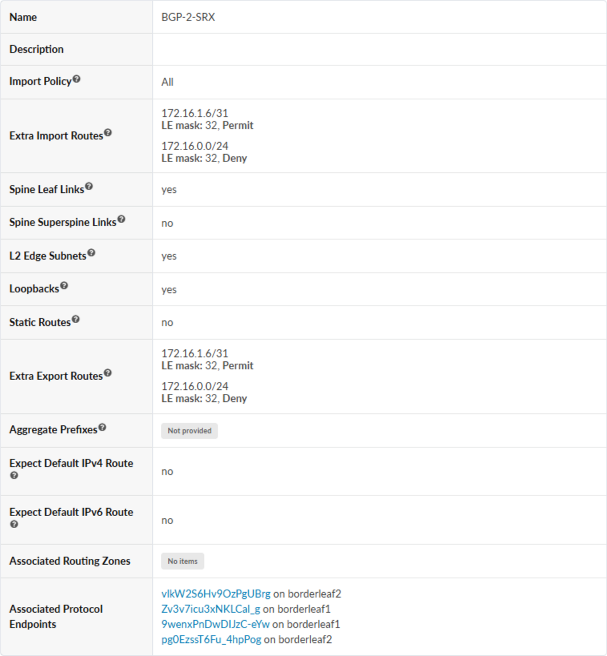
Use the following steps to set up the exact topology used in this Apstra ConnectorOps Setup Guide. There are many ways to configure the routing and design of your fabric, and this is just one example. All the configurations done in this process are specific to this topology.

If you are using your own topology, see the "Prerequisites" on page 4 for everything ConnectorOps needs to render SRX configurations in Apstra. High-level steps:

1. "Create a Routing Policy for Your Fabric" on page 29
2. "Create and Assign a Connectivity Template Between the SRXs and Border Leaves" on page 30
3. "Create External (Over the Top) Gateways in Apstra, and Designate them as Firewalls for Each SRX" on page 33
4. "Create VNs and Assign Routing Zones to Nodes" on page 34
5. "Configure Security Policies" on page 37

To Create a Routing Policy for Your Fabric

1. 1. Navigate to **Staged** > **Policies** > **Routing Policies** > **Create Routing Policy**.
   The Create Routing Policy window displays.
2. Configure your Routing Policy with the following parameters:

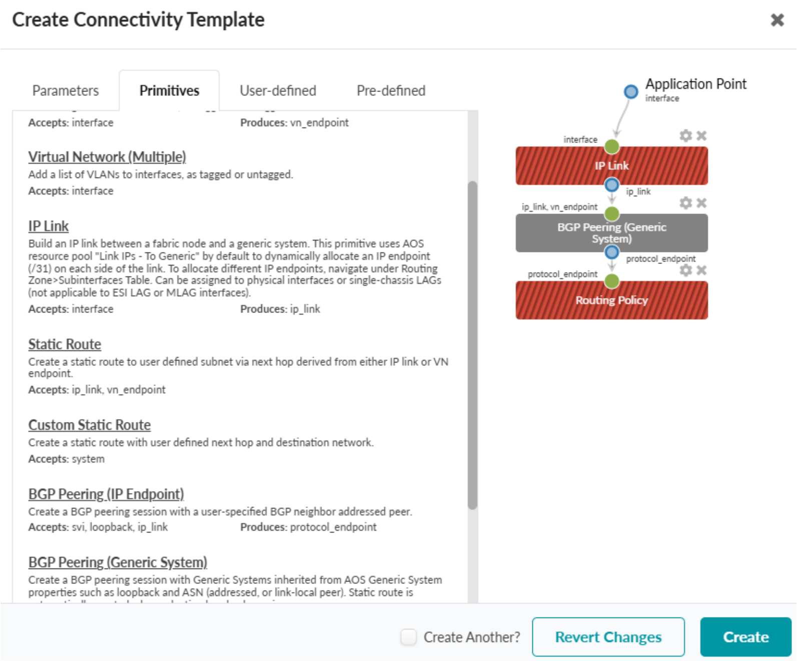


Name	BGP-2-SRX
Description	
Import Policy	All
Extra Import Routes	172.16.1.6/31 LE mask: 32, Permit 172.16.0.0/24 LE mask: 32, Deny
Spine Leaf Links	yes
Spine Superspine Links	no
L2 Edge Subnets	yes
Loopbacks	yes
Static Routes	no
Extra Export Routes	172.16.1.6/31 LE mask: 32, Permit 172.16.0.0/24 LE mask: 32, Deny
Aggregate Prefixes	Not provided
Expect Default IPv4 Route	no
Expect Default IPv6 Route	no
Associated Routing Zones	No items
Associated Protocol Endpoints	vlkW2S6Hv9OzPgUBrg on borderleaf2 Zv3v7icu3xNKLCal_g on borderleaf1 9wenxPnDwDIJzC-eYw on borderleaf1 pg0EzssT6Fu_4hpPog on borderleaf2

3. Click **Create**.

To Create and Assign a Connectivity Template Between the SRXs and Border Leaves

1. 1. From your blueprint, navigate to **Staged** > **Connectivity Templates** > **Add Template**.
   The Create Connectivity Template window displays.
2. Enter a name for your template, then select the **Primitives** tab.
3. Select **IP Link**, **BGP Peering (Generic System)**, and **Routing Policy**.



Create Connectivity Template

Parameters | **Primitives** | User-defined | Pre-defined

Accepts: interface **Produces**: vn_endpoint

Virtual Network (Multiple)
Add a list of VLANs to interfaces, as tagged or untagged.
Accepts: interface

IP Link
Build an IP link between a fabric node and a generic system. This primitive uses AOS resource pool "Link IPs - To Generic" by default to dynamically allocate an IP endpoint (/31) on each side of the link. To allocate different IP endpoints, navigate under Routing Zone>Subinterfaces Table. Can be assigned to physical interfaces or single-chassis LAGs (not applicable to ESI LAG or MLAG interfaces).
Accepts: interface **Produces**: ip_link

Static Route
Create a static route to user defined subnet via next hop derived from either IP link or VN endpoint.
Accepts: ip_link, vn_endpoint

Custom Static Route
Create a static route with user defined next hop and destination network.
Accepts: system

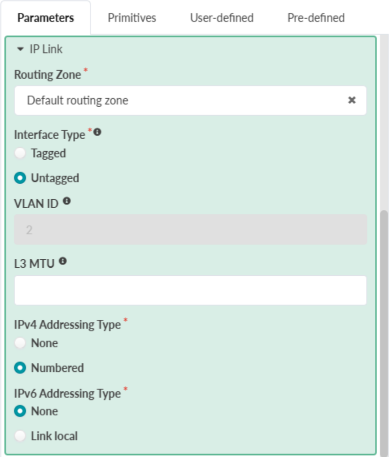
BGP Peering (IP Endpoint)
Create a BGP peering session with a user-specified BGP neighbor addressed peer.
Accepts: svi, loopback, ip_link **Produces**: protocol_endpoint

BGP Peering (Generic System)
Create a BGP peering session with Generic Systems inherited from AOS Generic System properties such as loopback and ASN (addressed, or link-local peer). Static route is

□ Create Another? Revert Changes Create

4. Select the **Parameters** tab and configure each primitive with the following parameters:

- IP Link



Parameters | Primitives | User-defined | Pre-defined

IP Link

Routing Zone *
Default routing zone

Interface Type *
○ Tagged
● Untagged

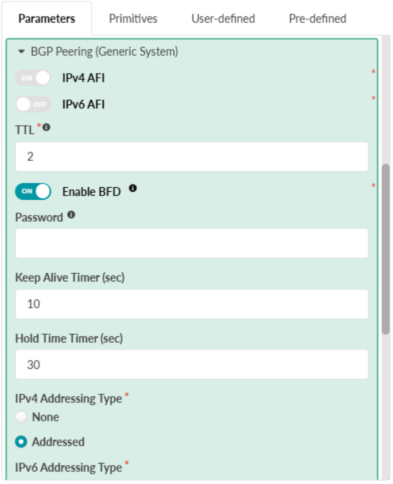
VLAN ID
2

L3 MTU

IPv4 Addressing Type *
○ None
● Numbered

IPv6 Addressing Type *
● None
○ Link local

- BGP Peering (Generic System)



▾ BGP Peering (Generic System)

ON IPv4 AFI *

OFF IPv6 AFI *

TTL * 2

ON Enable BFD *

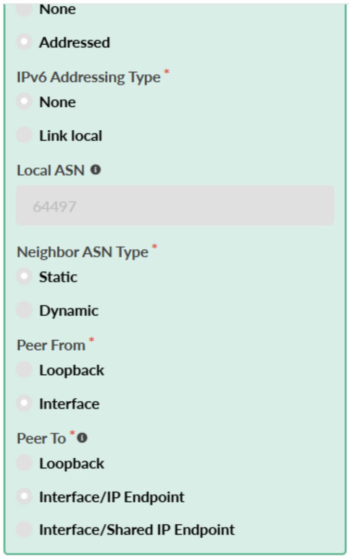
Password

Keep Alive Timer (sec) 10

Hold Time Timer (sec) 30

IPv4 Addressing Type *
○ None
● Addressed

IPv6 Addressing Type *



○ None
○ Addressed

IPv6 Addressing Type *
○ None
○ Link local

Local ASN 64497

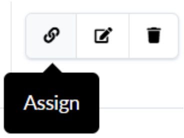
Neighbor ASN Type *
○ Static
○ Dynamic

Peer From *
○ Loopback
○ Interface

Peer To *
○ Loopback
○ Interface/IP Endpoint
○ Interface/Shared IP Endpoint

- Routing Policy: Your Routing Policy

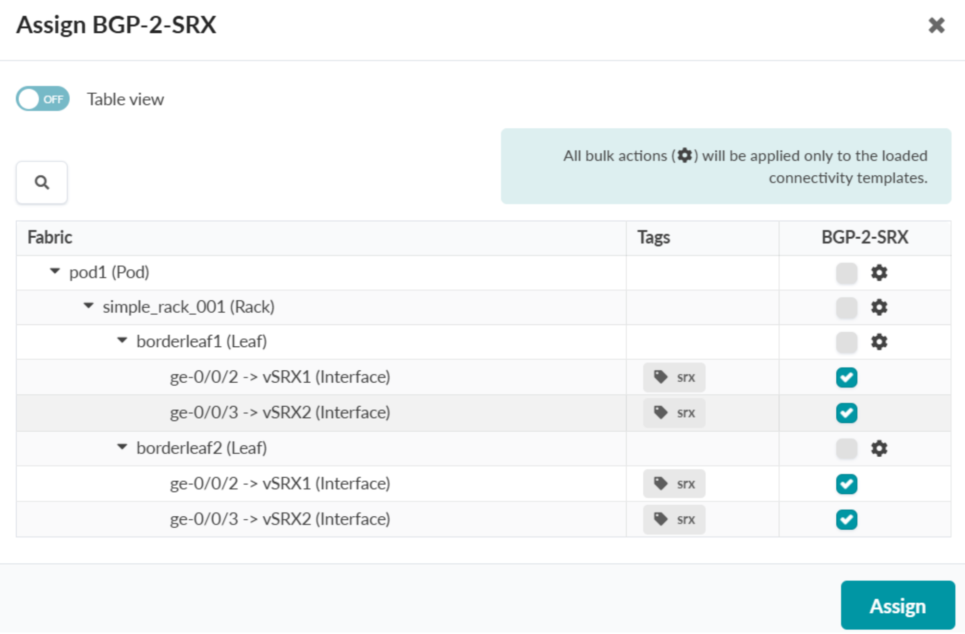
5. Click **Create**.
6. Select your Connectivity Template and click the **Assign** button.



7. Assign the template to each interface of your border leaves.

   This designates the desired BGP peers.
8. Click **Assign**.

   You can verify the new BGP peers by navigating to **Staged** > **Virtual** > **Routing Zones** > select default Routing Zone > **Interfaces** section.



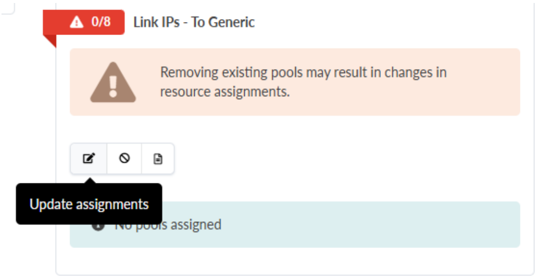
Assign BGP-2-SRX

Table view

All bulk actions (⚙) will be applied only to the loaded connectivity templates.

Fabric	Tags	BGP-2-SRX
pod1 (Pod)		☐ ⚙
simple_rack_001 (Rack)		☐ ⚙
borderleaf1 (Leaf)		☐ ⚙
ge-0/0/2 -> vSRX1 (Interface)	srx	☑
ge-0/0/3 -> vSRX2 (Interface)	srx	☑
borderleaf2 (Leaf)		☐ ⚙
ge-0/0/2 -> vSRX1 (Interface)	srx	☑
ge-0/0/3 -> vSRX2 (Interface)	srx	☑

9. Click **Assign**.
10. Assign Link IPs to Generic Systems



0/8 Link IPs - To Generic

Removing existing pools may result in changes in resource assignments.

No pools assigned

You can verify in **Staged** > **Virtual** > **Protocol Sessions**.

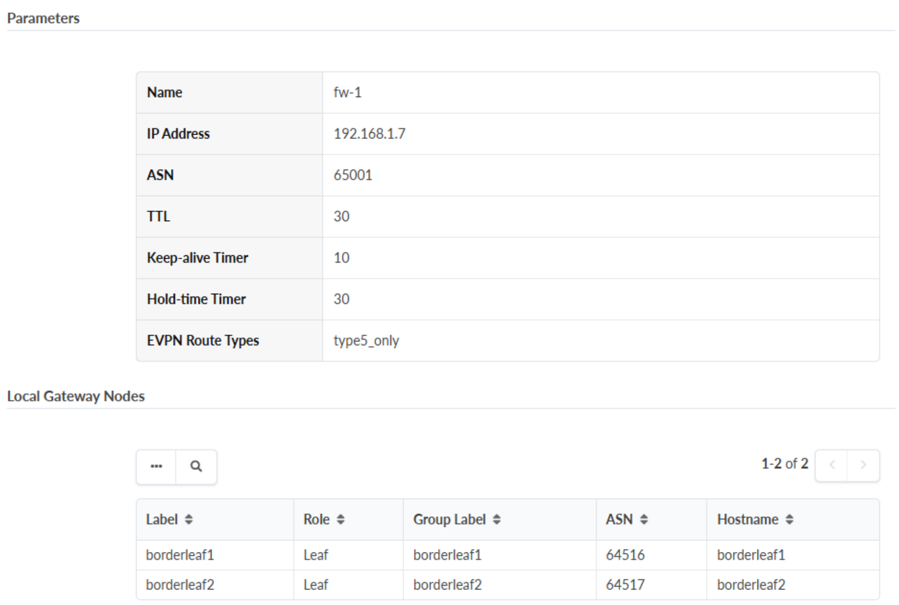
11. **Commit** your changes.

To Create External (Over the Top) Gateways in Apstra, and Designate them as Firewalls for Each SRX

1. Navigate to **Staged** > **DCI** > **Over the Top or External Gateways** > **Create Over the Top or External Gateway**.

   The Create Over the Top or External Gateway window displays.

2. 2. Configure an external gateway for each SRX, with the following parameters:



Parameters

Name	fw-1
IP Address	192.168.1.7
ASN	65001
TTL	30
Keep-alive Timer	10
Hold-time Timer	30
EVPN Route Types	type5_only

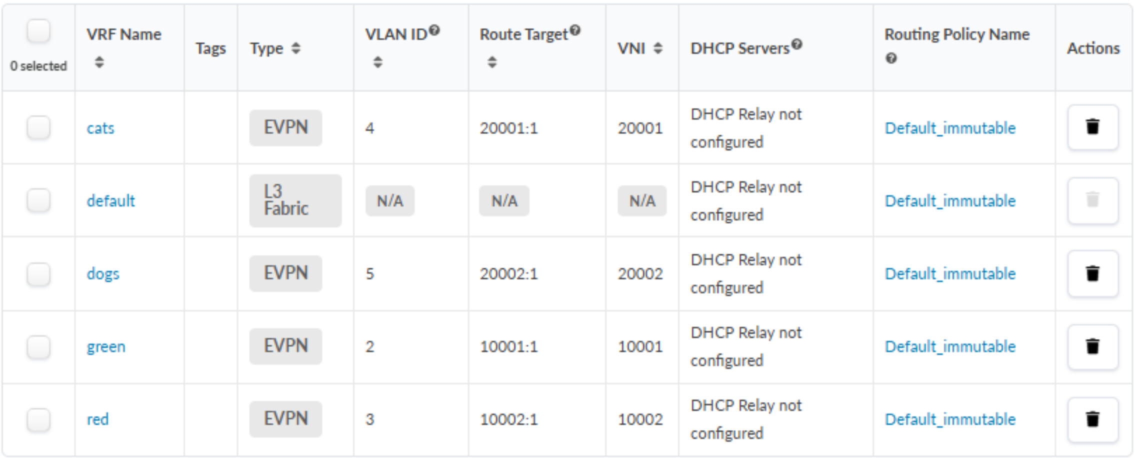
Local Gateway Nodes

Label ⇕	Role ⇕	Group Label ⇕	ASN ⇕	Hostname ⇕
borderleaf1	Leaf	borderleaf1	64516	borderleaf1
borderleaf2	Leaf	borderleaf2	64517	borderleaf2

3. Click **Create**.
4. **Commit** your changes.

To Create VNs and Assign Routing Zones to Nodes

1. Navigate to **Staged** > **Virtual** > **Routing Zones** > **Create Routing Zone**.
   The Create Routing Zone window displays.
2. Enter a name and **Virtual Network Interface** (VNI) number for your Routing Zone (RZ), then click **Create**.
   A Route Target and VLAN ID are automatically assigned.
3. Repeat this process until A list of Routing Zones displays. The following is an example list of Routing Zones.



□ 0 selected	VRF Name ⇕	Tags	Type ⇕	VLAN ID ⇕	Route Target ⇕	VNI ⇕	DHCP Servers	Routing Policy Name	Actions
□	cats		EVPN	4	20001:1	20001	DHCP Relay not configured	Default_immutable	🗑
□	default		L3 Fabric	N/A	N/A	N/A	DHCP Relay not configured	Default_immutable	🗑
□	dogs		EVPN	5	20002:1	20002	DHCP Relay not configured	Default_immutable	🗑
□	green		EVPN	2	10001:1	10001	DHCP Relay not configured	Default_immutable	🗑
□	red		EVPN	3	10002:1	10002	DHCP Relay not configured	Default_immutable	🗑

4. Select the **Virtual Networks** tab.

   Let's create VNs to assign to each RZ we created.

5. Click **Create Virtual Network**.

   The Create Virtual Networks window displays.

6. Enter the following information:

   - Name: Name of the VN

   - Routing Zone: Select an RZ to associate with the VN. In this example, **green-1** is associated with RZ **green** that we previously created.

   - VNI(s):

   - VLAN ID (on leafs):

7. Select the box for **Reserve across blueprint**.

8. Enter the following:

   - IPv4 Subnet: The subnet the VN uses for addressing.

   - Virtual Gateway IPv4: The IP address that is assigned to the Virtual Gateway.

9. Under Create Connectivity Templates for, select the box for **Untagged**.

10. In the Assigned To section, select each border leaf, and the rack containing the leaf pair.

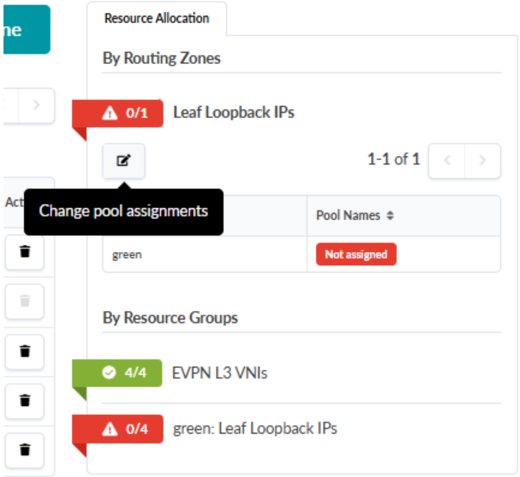
11. Click **Create**.

12. Repeat this process until each RZ has an associated VN.

    The list of VNs displays. Now, let's designate a Loopback IP pool for the VNs.

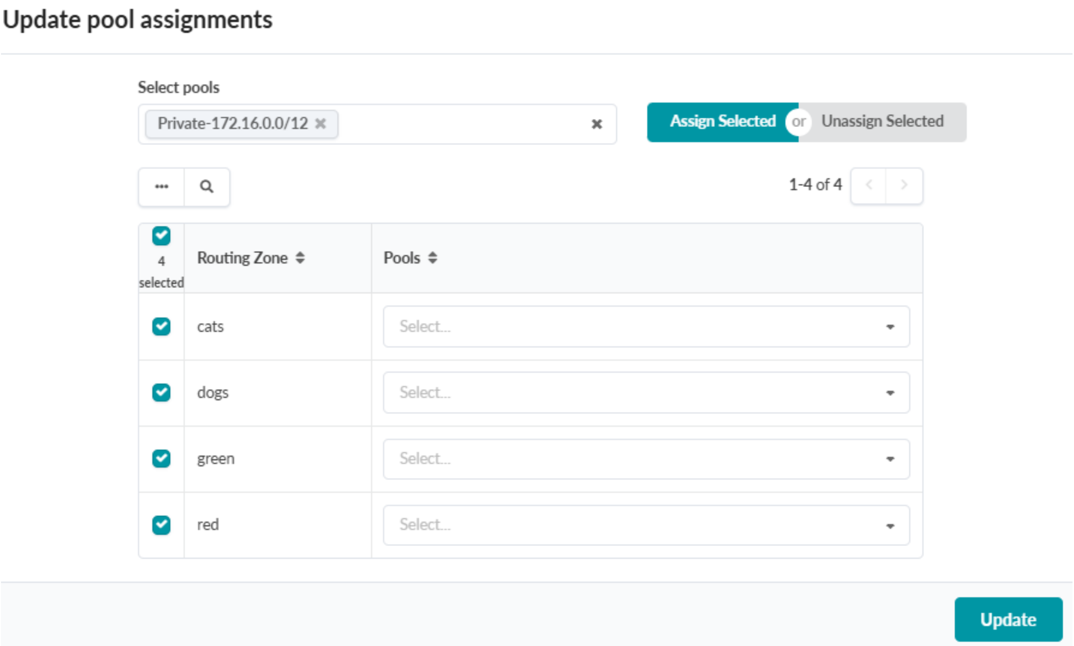
13. Navigate to **Virtual** > **Routing Zones**.

14. Click the **Assign** button to assign an IP pool for Loopback IP addressing.



15. Select all of the RZs, then select an IP pool from the dropdown at the top.

16. Select **Assign Selected**, then click **Update**.



Update pool assignments

Select pools

Private-172.16.0.0/12

☑ 4 selected	Routing Zone	Pools
☑	cats	Select...
☑	dogs	Select...
☑	green	Select...
☑	red	Select...

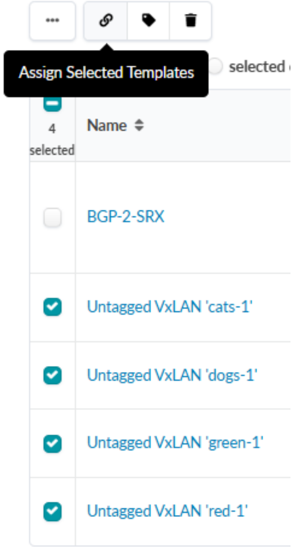
17. **Commit** your changes.

    Next, let's assign the RZs to the appropriate nodes.

18. Navigate to **Staged** > **Connectivity Templates**.

19. Select the new VNs from the list, and select the Assign Selected Templates button at the top.

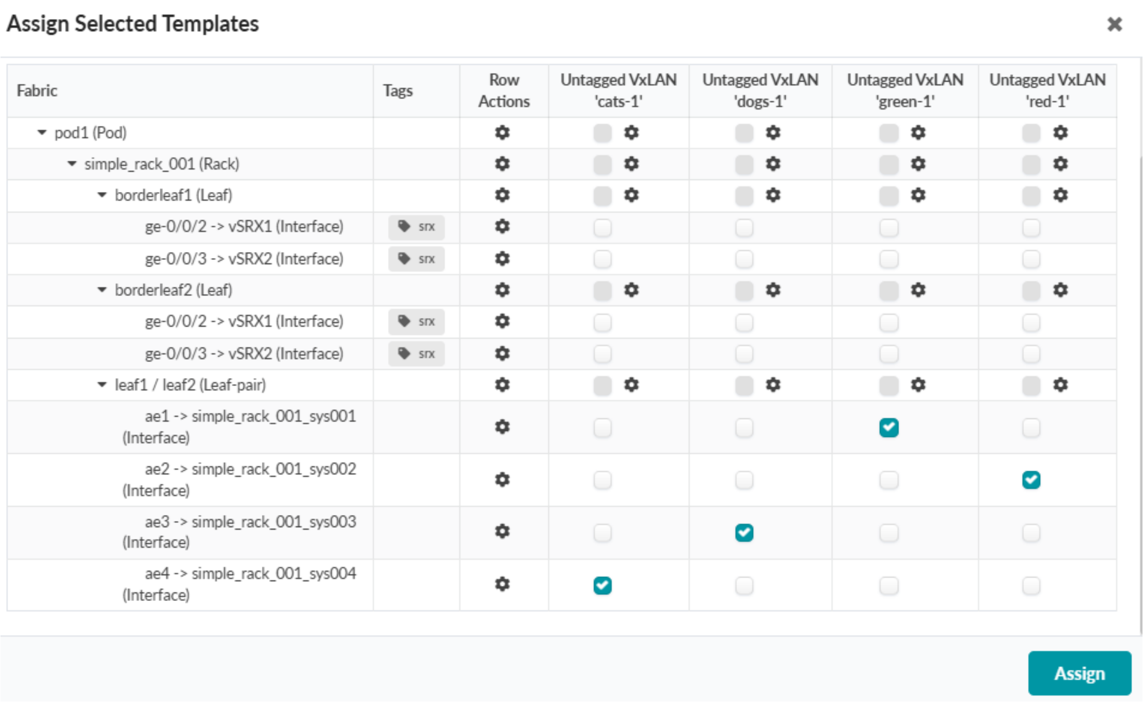
    The Assign Selected Templates window displays. Note the columns for each VN at the top of the table.



4 selected	Name
□	BGP-2-SRX
☑	Untagged VxLAN 'cats-1'
☑	Untagged VxLAN 'dogs-1'
☑	Untagged VxLAN 'green-1'
☑	Untagged VxLAN 'red-1'

20. Assign each VN to a corresponding interface (endpoint).

    The following example shows the designated VN assignments.



Fabric	Tags	Row Actions	Untagged VxLAN 'cats-1'	Untagged VxLAN 'dogs-1'	Untagged VxLAN 'green-1'	Untagged VxLAN 'red-1'
▾ pod1 (Pod)		⚙	□ ⚙	□ ⚙	□ ⚙	□ ⚙
▾ simple_rack_001 (Rack)		⚙	□ ⚙	□ ⚙	□ ⚙	□ ⚙
▾ borderleaf1 (Leaf)		⚙	□ ⚙	□ ⚙	□ ⚙	□ ⚙
ge-0/0/2 -> vSRX1 (Interface)	srx	⚙	□	□	□	□
ge-0/0/3 -> vSRX2 (Interface)	srx	⚙	□	□	□	□
▾ borderleaf2 (Leaf)		⚙	□ ⚙	□ ⚙	□ ⚙	□ ⚙
ge-0/0/2 -> vSRX1 (Interface)	srx	⚙	□	□	□	□
ge-0/0/3 -> vSRX2 (Interface)	srx	⚙	□	□	□	□
▾ leaf1 / leaf2 (Leaf-pair)		⚙	□ ⚙	□ ⚙	□ ⚙	□ ⚙
ae1 -> simple_rack_001_sys001 (Interface)		⚙	□	□	☑	□
ae2 -> simple_rack_001_sys002 (Interface)		⚙	□	□	□	☑
ae3 -> simple_rack_001_sys003 (Interface)		⚙	□	☑	□	□
ae4 -> simple_rack_001_sys004 (Interface)		⚙	☑	□	□	□

21. Navigate to **Staged** > **Fabric Settings** > **Fabric Policy** > **Modify Settings**.

    The Modify Fabric Policy Settings window displays.

22. Under Default IP Links to Generic Systems MTU, enter **9000**.

23. **Commit** your changes.

To Configure Security Policies

The following are the exact Security Policies applied on the SRX devices in this example topology for end-to-end routing.

```
set groups COMMON security address-book global address FW-1 172.16.1.6/32
set groups COMMON security address-book global address FW-2-lo 172.16.1.7/32
set groups COMMON security address-book global address FW-2-ge-0-0-0 192.168.0.19/32
set groups COMMON security address-book global address FW-2-ge-0-0-1 192.168.0.23/32
set groups COMMON security address-book global address BGP-0 172.16.0.4/32
set groups COMMON security address-book global address BGP-1 172.16.0.5/32
set groups COMMON security address-book global address green 210.210.0.0/16
set groups COMMON security address-book global address red 200.200.0.0/16
set groups COMMON security address-book global address cats 221.221.0.0/16
set groups COMMON security address-book global address dogs 220.220.0.0/16
set groups COMMON security address-book global address-set FW-2 address FW-2-lo
set groups COMMON security address-book global address-set FW-2 address FW-2-ge-0-0-0
set groups COMMON security address-book global address-set FW-2 address FW-2-ge-0-0-1
set groups COMMON security policies from-zone evpn to-zone evpn policy HA-ICL match source-address FW-2
set groups COMMON security policies from-zone evpn to-zone evpn policy HA-ICL match destination-address FW-1
set groups COMMON security policies from-zone evpn to-zone evpn policy HA-ICL match application any
set groups COMMON security policies from-zone evpn to-zone evpn policy HA-ICL then permit
set groups COMMON security policies from-zone evpn to-zone evpn policy HA-ICL then log session-init
set groups COMMON security policies from-zone evpn to-zone evpn policy HA-ICL then log session-close
set groups COMMON security policies from-zone evpn to-zone evpn policy G2R match source-address green
set groups COMMON security policies from-zone evpn to-zone evpn policy G2R match destination-address red
set groups COMMON security policies from-zone evpn to-zone evpn policy G2R match application junos-ping
set groups COMMON security policies from-zone evpn to-zone evpn policy G2R match application junos-ssh
set groups COMMON security policies from-zone evpn to-zone evpn policy G2R match source-l3vpn-vrf-group GREEN
set groups COMMON security policies from-zone evpn to-zone evpn policy G2R match destination-l3vpn-vrf-group RED
set groups COMMON security policies from-zone evpn to-zone evpn policy G2R then permit
set groups COMMON security policies from-zone evpn to-zone evpn policy G2R then log session-init
set groups COMMON security policies from-zone evpn to-zone evpn policy G2R then log session-close
set groups COMMON security policies from-zone evpn to-zone evpn policy R2G match source-address red
set groups COMMON security policies from-zone evpn to-zone evpn policy R2G match destination-address green
set groups COMMON security policies from-zone evpn to-zone evpn policy R2G match application junos-http
set groups COMMON security policies from-zone evpn to-zone evpn policy R2G match application junos-ping
set groups COMMON security policies from-zone evpn to-zone evpn policy R2G match source-l3vpn-vrf-group RED
set groups COMMON security policies from-zone evpn to-zone evpn policy R2G match destination-l3vpn-vrf-group GREEN
set groups COMMON security policies from-zone evpn to-zone evpn policy R2G then permit
set groups COMMON security policies from-zone evpn to-zone evpn policy R2G then log session-init
set groups COMMON security policies from-zone evpn to-zone evpn policy R2G then log session-close
set groups COMMON security policies from-zone evpn to-zone evpn policy D2C match source-address dogs
set groups COMMON security policies from-zone evpn to-zone evpn policy D2C match destination-address cats
set groups COMMON security policies from-zone evpn to-zone evpn policy D2C match application any
set groups COMMON security policies from-zone evpn to-zone evpn policy D2C match source-l3vpn-vrf-group DOGS
set groups COMMON security policies from-zone evpn to-zone evpn policy D2C match destination-l3vpn-vrf-group CATS
set groups COMMON security policies from-zone evpn to-zone evpn policy D2C then permit
set groups COMMON security policies from-zone evpn to-zone evpn policy C2D match source-address cats
set groups COMMON security policies from-zone evpn to-zone evpn policy C2D match destination-address dogs
set groups COMMON security policies from-zone evpn to-zone evpn policy C2D match application junos-ping
set groups COMMON security policies from-zone evpn to-zone evpn policy C2D match source-l3vpn-vrf-group CATS
set groups COMMON security policies from-zone evpn to-zone evpn policy C2D match destination-l3vpn-vrf-group DOGS
set groups COMMON security policies from-zone evpn to-zone evpn policy C2D then permit
set groups COMMON security policies from-zone evpn to-zone evpn policy FINAL-EVPN-DENY match source-address any
set groups COMMON security policies from-zone evpn to-zone evpn policy FINAL-EVPN-DENY match destination-address any
set groups COMMON security policies from-zone evpn to-zone evpn policy FINAL-EVPN-DENY match application junos-vxlan
set groups COMMON security policies from-zone evpn to-zone evpn policy FINAL-EVPN-DENY then deny
set groups COMMON security policies from-zone evpn to-zone evpn policy infra description "control plane permit"
set groups COMMON security policies from-zone evpn to-zone evpn policy infra match source-address any
set groups COMMON security policies from-zone evpn to-zone evpn policy infra match destination-address any
set groups COMMON security policies from-zone evpn to-zone evpn policy infra match application junos-bgp
set groups COMMON security policies from-zone evpn to-zone evpn policy infra match application junos-ping
set groups COMMON security policies from-zone evpn to-zone evpn policy infra match application bfd-mhop
set groups COMMON security policies from-zone evpn to-zone evpn policy infra then permit
set groups COMMON security policies from-zone evpn to-zone evpn policy infra then log session-init
set groups COMMON security policies from-zone evpn to-zone evpn policy infra then log session-close
set groups COMMON security policies from-zone evpn to-zone untrust policy C2EXT match source-address cats
set groups COMMON security policies from-zone evpn to-zone untrust policy C2EXT match destination-address any
set groups COMMON security policies from-zone evpn to-zone untrust policy C2EXT match application junos-http
set groups COMMON security policies from-zone evpn to-zone untrust policy C2EXT match application junos-http-ext
set groups COMMON security policies from-zone evpn to-zone untrust policy C2EXT match application junos-https
set groups COMMON security policies from-zone evpn to-zone untrust policy C2EXT match application junos-dns-udp
set groups COMMON security policies from-zone evpn to-zone untrust policy C2EXT match application junos-dns-tcp
set groups COMMON security policies from-zone evpn to-zone untrust policy C2EXT match application junos-ping
set groups COMMON security policies from-zone evpn to-zone untrust policy C2EXT match source-l3vpn-vrf-group CATS
set groups COMMON security policies from-zone evpn to-zone untrust policy C2EXT then permit
set groups COMMON security policies global policy infra-reject-log description "final touch"
set groups COMMON security policies global policy infra-reject-log match source-address any
set groups COMMON security policies global policy infra-reject-log match destination-address any
set groups COMMON security policies global policy infra-reject-log match application any
set groups COMMON security policies global policy infra-reject-log then reject
set groups COMMON security policies global policy infra-reject-log then log session-init
set groups COMMON security policies global policy infra-reject-log then log session-close
set groups COMMON security l3vpn vrf-group GREEN vrf GREEN
set groups COMMON security l3vpn vrf-group RED vrf RED
set groups COMMON security l3vpn vrf-group CATS vrf CATS
set groups COMMON security l3vpn vrf-group DOGS vrf DOGS
set security policies pre-id-default-policy then log session-close
set security zones security-zone trust tcp-rst
set security zones security-zone untrust screen untrust-screen
set security zones security-zone evpn host-inbound-traffic system-services all
set security zones security-zone evpn host-inbound-traffic protocols all
set security zones security-zone evpn interfaces ge-0/0/0.0
set security zones security-zone evpn interfaces ge-0/0/1.0
set security zones security-zone evpn interfaces lo0.0
set apply-groups COMMON
```

After setting up this topology, proceed to "Deploy ConnectorOps and Verify Connectivity" on page 9.

Create a Device Profile and Assign an Interface Map to the SRX

This step is a Blueprint prerequisite that you must complete to run ConnectorOps. For other prerequisites, see "Prerequisites" on page 4.

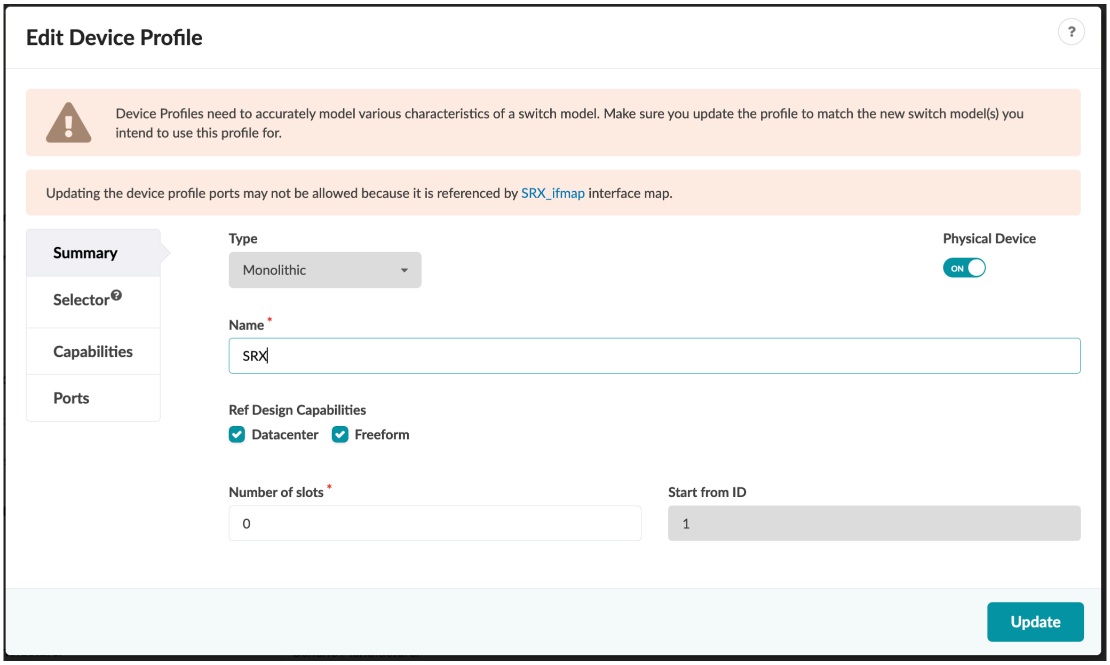
You must create a Device Profile for the Juniper SRX because it is onboarded as a Generic System in Apstra. A Device Profile specifies ports and hardware capabilities.

To create a Device Profile and Assign an Interface Map to the SRX:

1. Navigate to **Devices** > **Device Profiles** > **Create Device Profile**.

   The Create Device Profile window displays.

2. For the Summary tab, choose a **name** for your SRX and mirror the settings in the following screenshot.



Edit Device Profile

Device Profiles need to accurately model various characteristics of a switch model. Make sure you update the profile to match the new switch model(s) you intend to use this profile for.

Updating the device profile ports may not be allowed because it is referenced by SRX_ifmap interface map.

Type: Monolithic

Physical Device: ON

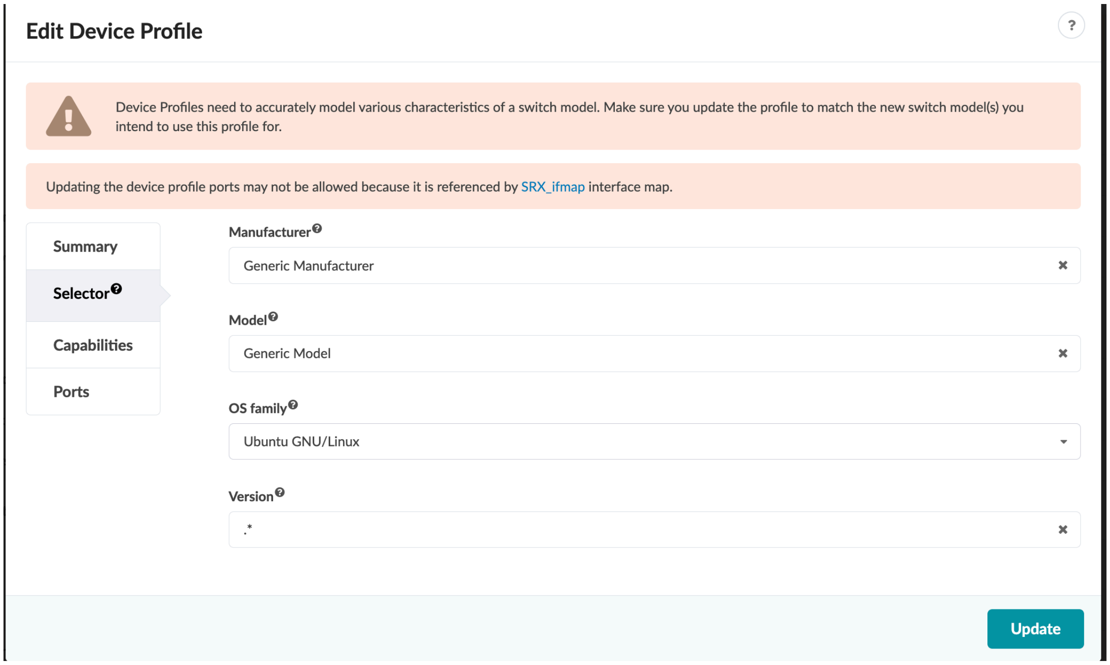
Name *: SRX

Ref Design Capabilities: ☑ Datacenter ☑ Freeform

Number of slots *: 0

Start from ID: 1

3. For the Selector tab, mirror the settings in the following screenshot.



Edit Device Profile

Device Profiles need to accurately model various characteristics of a switch model. Make sure you update the profile to match the new switch model(s) you intend to use this profile for.

Updating the device profile ports may not be allowed because it is referenced by SRX_ifmap interface map.

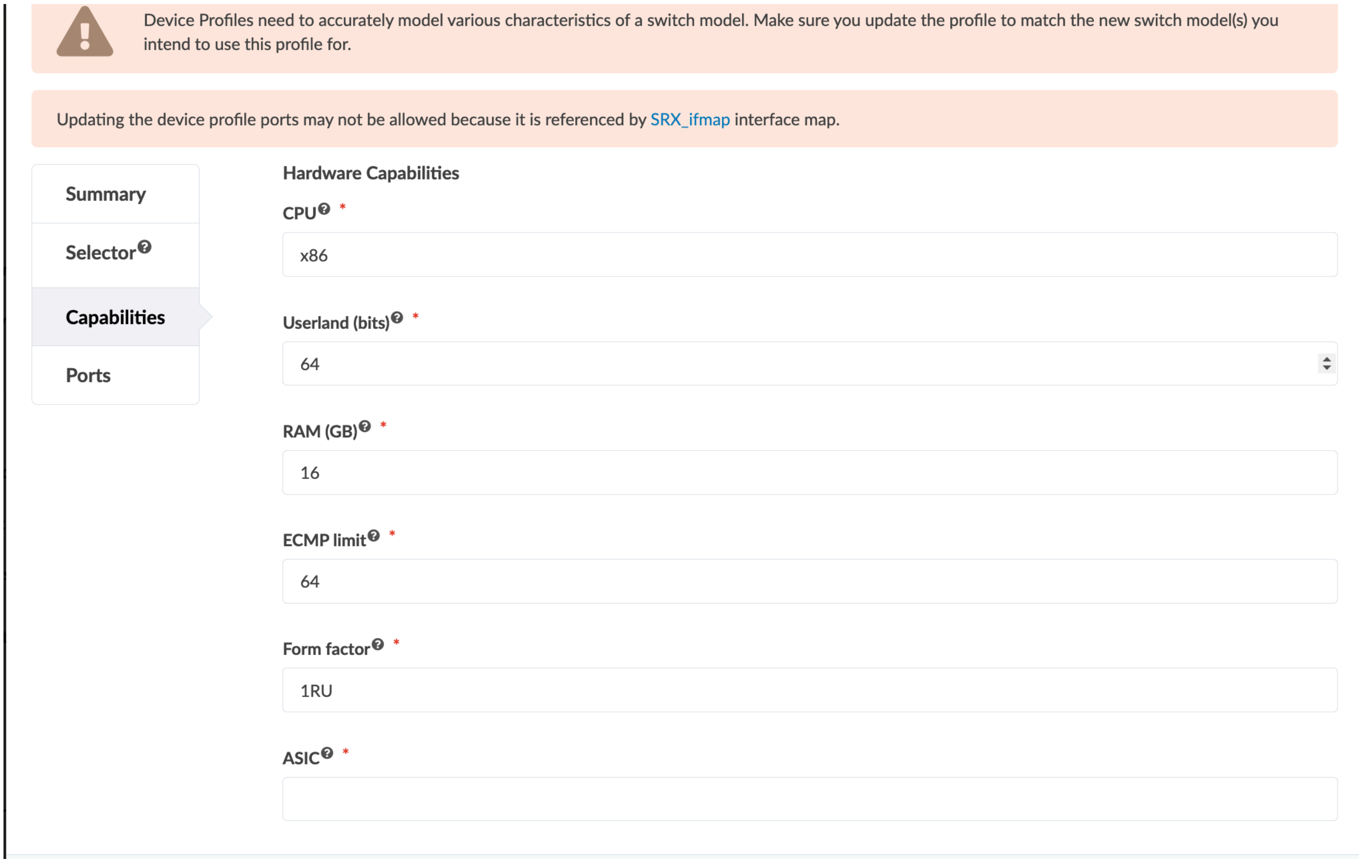
Manufacturer: Generic Manufacturer

Model: Generic Model

OS family: Ubuntu GNU/Linux

Version: .*

4. For the Capabilities tab, mirror the settings in the following screenshots.



Device Profiles need to accurately model various characteristics of a switch model. Make sure you update the profile to match the new switch model(s) you intend to use this profile for.

Updating the device profile ports may not be allowed because it is referenced by SRX_ifmap interface map.

Hardware Capabilities

CPU *: x86

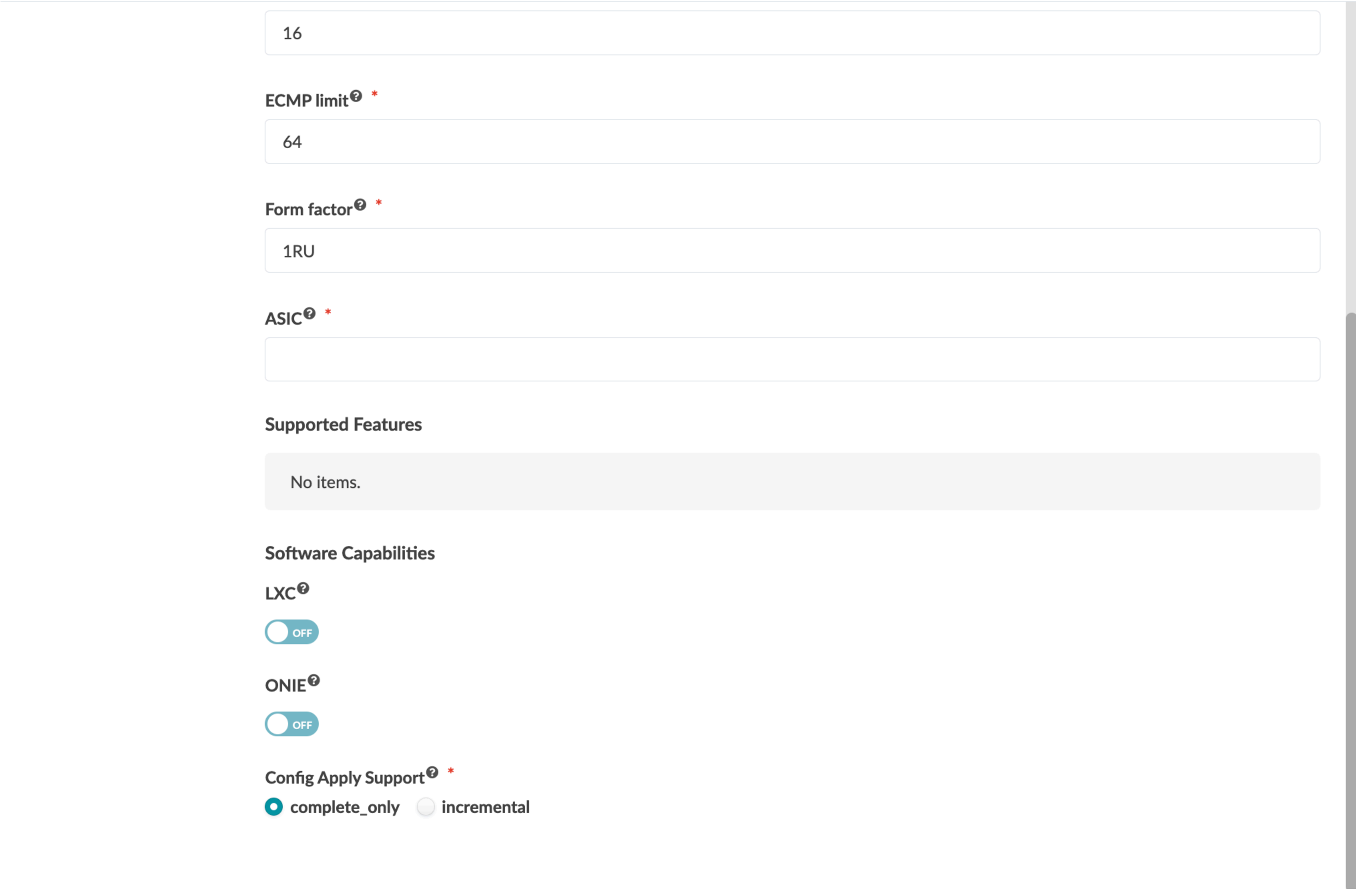
Userland (bits) *: 64

RAM (GB) *: 16

ECMP limit *: 64

Form factor *: 1RU

ASIC *:



16

ECMP limit *

64

Form factor *

1RU

ASIC *

Supported Features

No items.

Software Capabilities

LXC

OFF

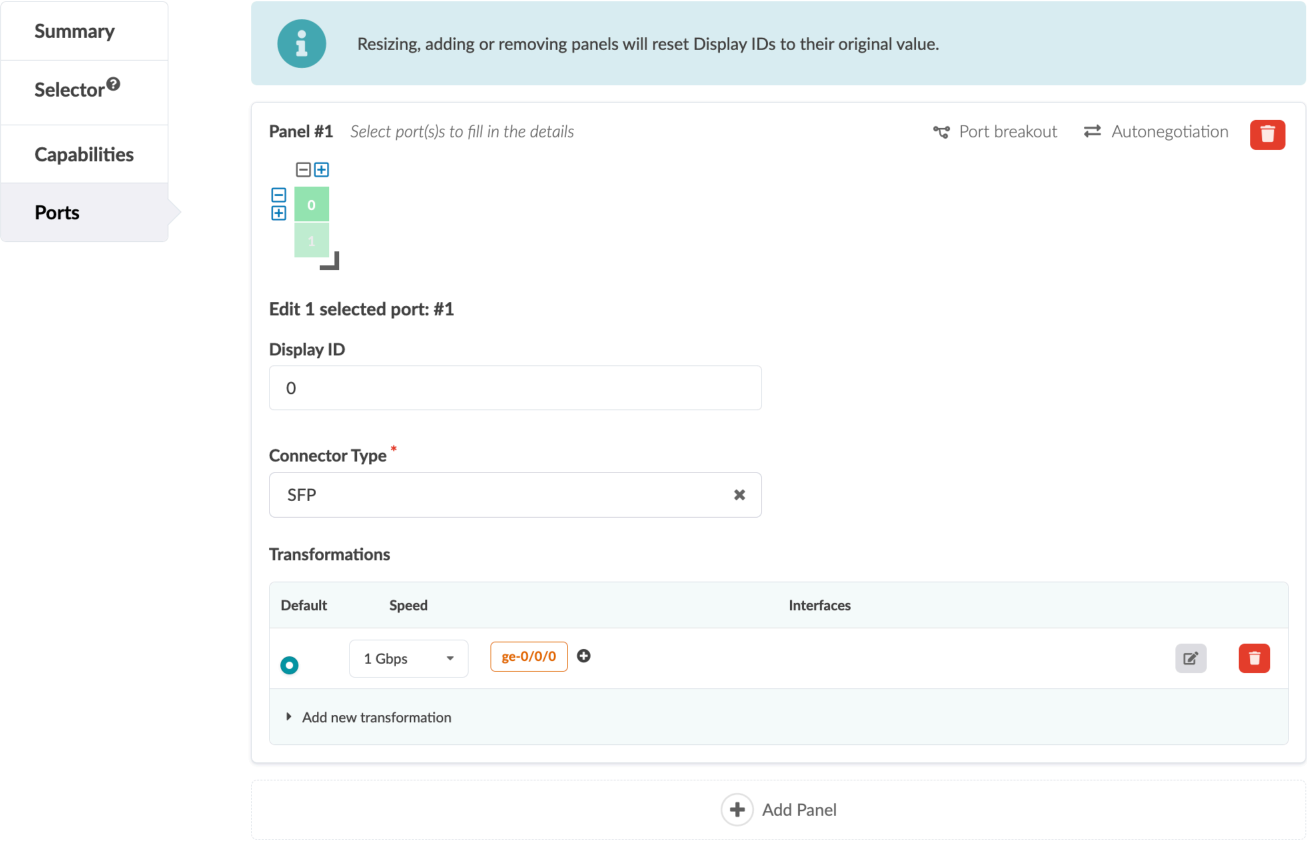
ONIE

OFF

Config Apply Support *

● complete_only ○ incremental

5. For the Ports tab, mirror the settings in the following screenshots.



Resizing, adding or removing panels will reset Display IDs to their original value.

Panel #1 *Select port(s)s to fill in the details*

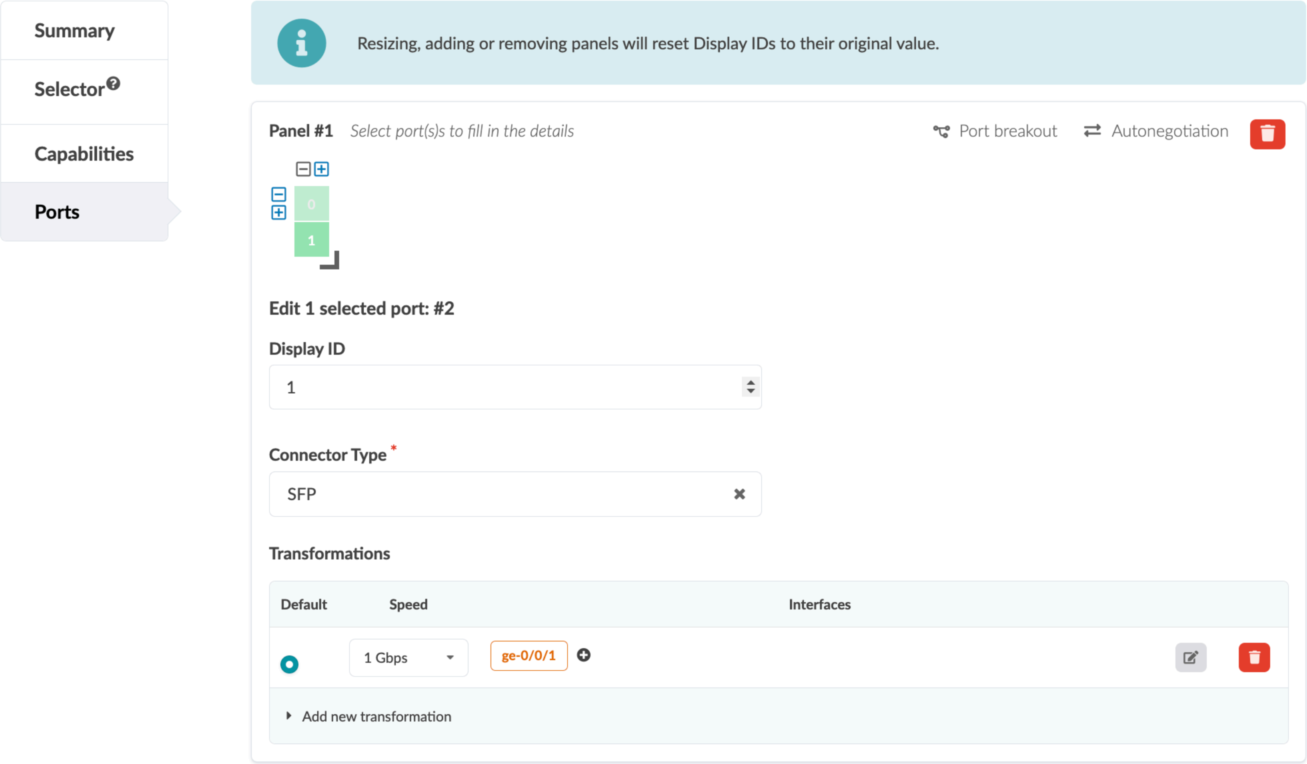
Edit 1 selected port: #1

Display ID: 0

Connector Type *: SFP

Transformations

Default	Speed	Interfaces
●	1 Gbps	ge-0/0/0



Resizing, adding or removing panels will reset Display IDs to their original value.

Panel #1 *Select port(s)s to fill in the details*

Edit 1 selected port: #2

Display ID: 1

Connector Type *: SFP

Transformations

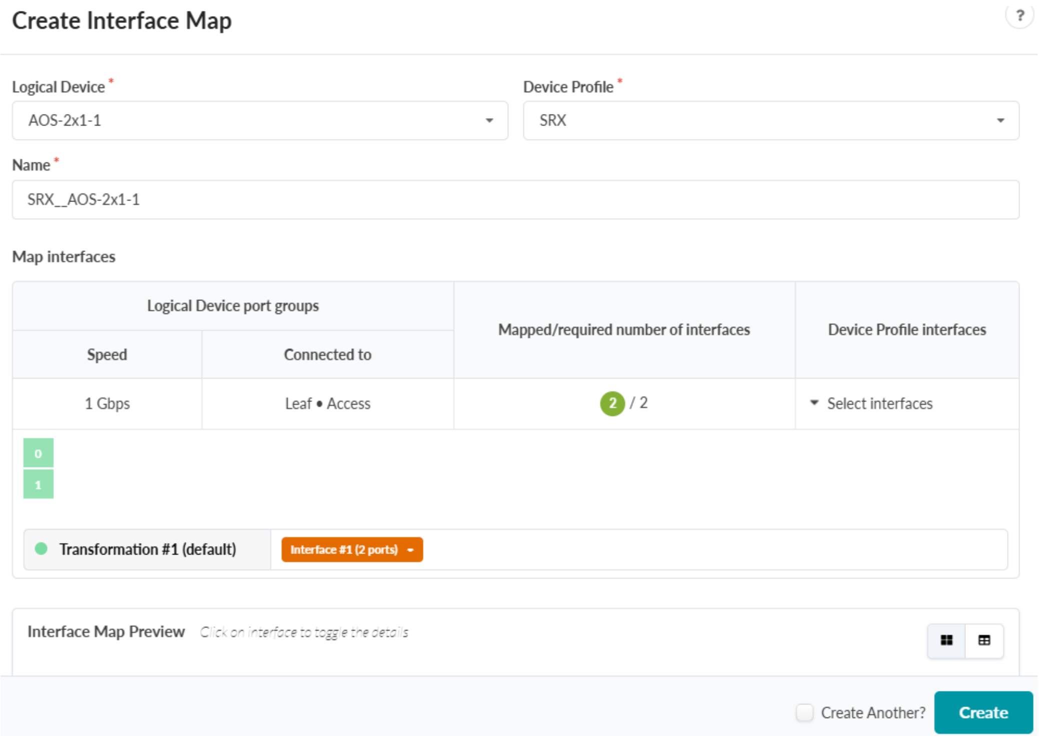
Default	Speed	Interfaces
●	1 Gbps	ge-0/0/1

6. Navigate to **Design** > **Interface Maps** > **Create Interface Map**.

The Create Interface Map window displays.

7. Select a **Logical Device** from the dropdown.
8. Select the **SRX Device Profile** from the dropdown.
9. Enter a **name** for the Interface Map.
10. Under the Map interfaces section, click **Select interfaces** in the Device Profile Interfaces column.

    The interfaces of your Logical Device display.
11. Select each interface to map the port groups to their respective interfaces.



Create Interface Map

Logical Device * AOS-2x1-1

Device Profile * SRX

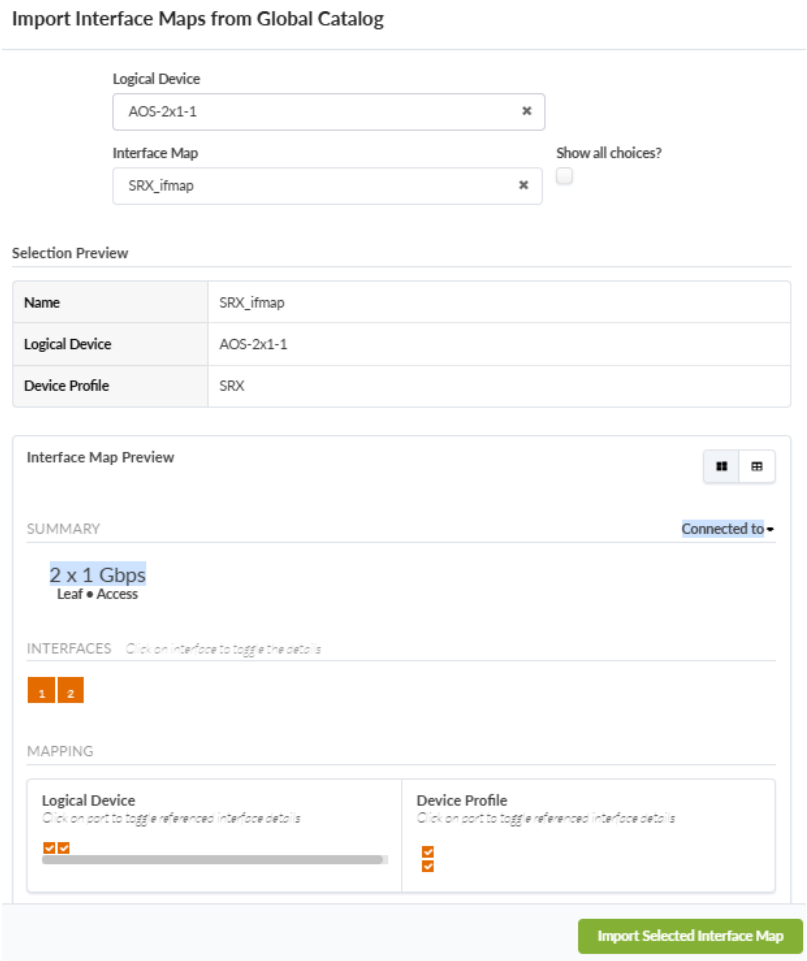
Name * SRX__AOS-2x1-1

Map interfaces

Logical Device port groups		Mapped/required number of interfaces	Device Profile interfaces
Speed	Connected to		
1 Gbps	Leaf • Access	2 / 2	▾ Select interfaces

12. Click **Create**.
13. From your Blueprint, navigate to **Staged** > **Catalog** > **Interface Maps** > **Import Interface Map**.

    The Import Interface Map from Global Catalog window displays.
14. Select the Logical Device and Interface Map from the dropdowns.
15. Ensure that the mapped interfaces are correct and click **Import Selected Interface Map**.



Import Interface Maps from Global Catalog

Logical Device: AOS-2x1-1

Interface Map: SRX_ifmap

Show all choices? □

Selection Preview

Name	SRX_ifmap
Logical Device	AOS-2x1-1
Device Profile	SRX

Interface Map Preview

SUMMARY — Connected to

2 x 1 Gbps
Leaf • Access

INTERFACES — Click on interface to toggle the details

1 2

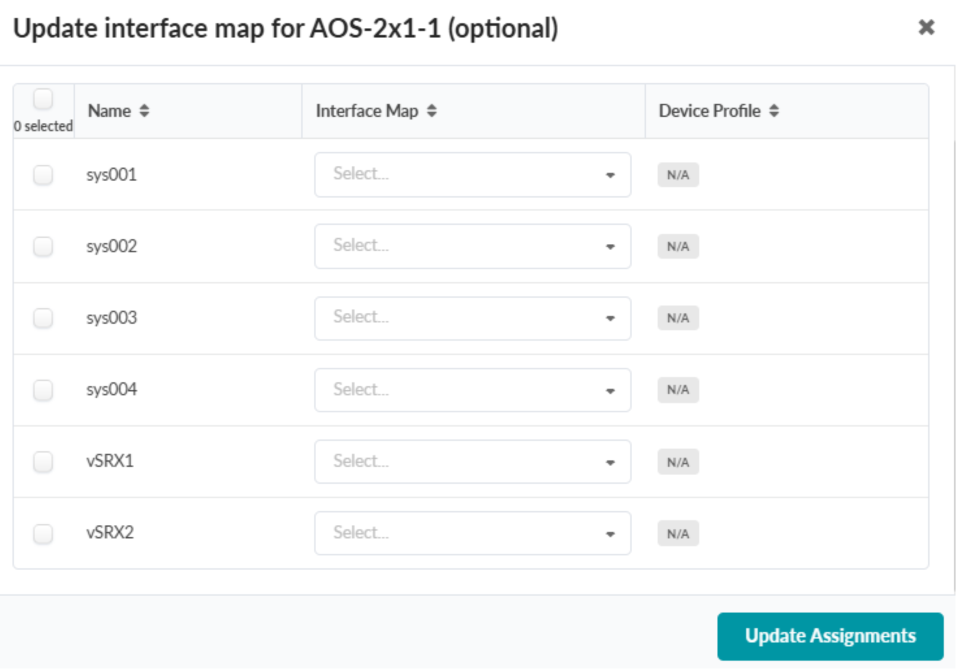
MAPPING

Logical Device — Click on port to toggle referenced interface details

Device Profile — Click on port to toggle referenced interface details

16. From the Blueprint, navigate to **Physical** > **Topology** view.

17. In the Build view, select **Device Profiles** > **Change interface map assignments** for the logical device you created.

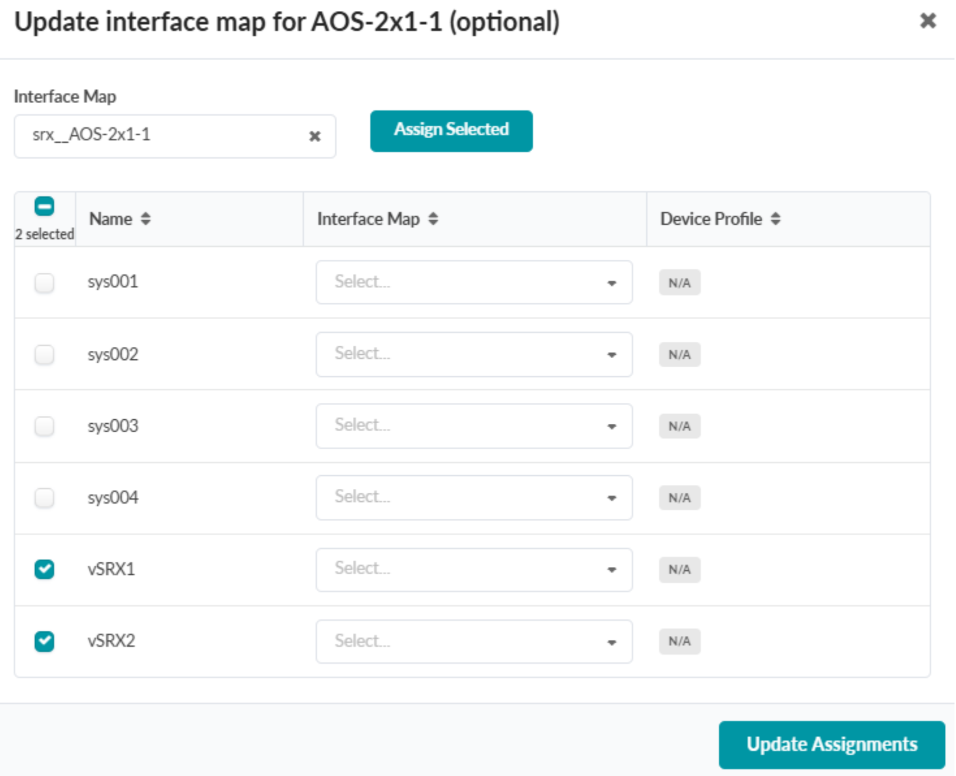


Update interface map for AOS-2x1-1 (optional) ✖

□ 0 selected	Name ⬍	Interface Map ⬍	Device Profile ⬍
□	sys001	Select...	N/A
□	sys002	Select...	N/A
□	sys003	Select...	N/A
□	sys004	Select...	N/A
□	vSRX1	Select...	N/A
□	vSRX2	Select...	N/A

18. Select the SRXs, then select the Interface Map from the dropdown and click **Assign Selected**.

19. Click **Update Assignments**.



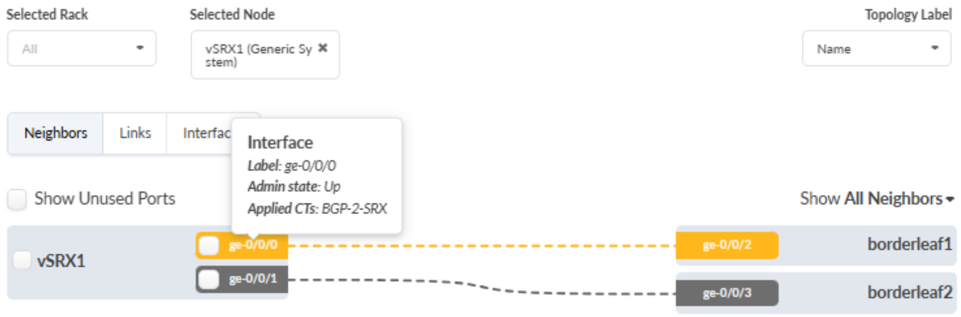
Update interface map for AOS-2x1-1 (optional) ✖

Interface Map

srx__AOS-2x1-1 ✖

⊟ 2 selected	Name ⬍	Interface Map ⬍	Device Profile ⬍
□	sys001	Select...	N/A
□	sys002	Select...	N/A
□	sys003	Select...	N/A
□	sys004	Select...	N/A
☑	vSRX1	Select...	N/A
☑	vSRX2	Select...	N/A

The SRX has an assigned Device Profile and Interface Map. You can view the links and interface names in the Topology view.



Selected Rack: All

Selected Node: vSRX1 (Generic System)

Topology Label: Name

Interface

Label: ge-0/0/0

Admin state: Up

Applied CTs: BGP-2-SRX

If you are setting up one of the Supported Topologies, ensure that the other Prerequisites are complete and proceed to "Deploy Apstra ConnectorOps and Verify Connectivity" on page 9.

For information about how to set up the example topology used in this guide, see "Set Up the Example ConnectorOps Topology" on page 29.