

# In Focus

## Juniper Apstra: How to Automate Your Data Center

### IN THIS GUIDE

- [About This In Focus Use Case | 1](#)
- [Workflow | 4](#)
- [Prepare | 5](#)
- [Design | 13](#)
- [Build | 22](#)
- [Deploy | 39](#)
- [Summary | 52](#)

## About This In Focus Use Case

Use Case	Use Juniper Apstra to design, build, and deploy your data center fabric.
Audience	Data center network administrator
Knowledge Level	General familiarity with EVPN-VXLAN data center network architectures and underlay and overlay routing. See the related topics section for background information on EVPN-VXLAN technology.

Benefits	<ul style="list-style-type: none"> <li>• Use intent-based networking and intent-based analytics to manage your fabric throughout its life cycle.</li> <li>• Increase efficiency and network awareness by managing your data center as a whole rather than as a collection of independent devices.</li> <li>• Reduce the risk of errors and misconfiguration through automation. Reduce service delivery times when compared to manual provisioning.</li> </ul>
Products Used	<ul style="list-style-type: none"> <li>• Apstra Release 4.0.0</li> <li>• QFX Series switches running Junos OS <ul style="list-style-type: none"> <li>• tested on QFX10002 running Junos OS 20.2R2-S1.3 as a spine device</li> <li>• tested on QFX5110 running Junos OS 20.2R2-S1.3 as a leaf device</li> </ul> </li> </ul> <p>For the full list of supported devices and OS versions, see <a href="#">Supported Juniper Devices</a>.</p>

**NOTE:** Are you interested in getting hands-on experience with the topics and operations covered in this guide? Visit [Juniper Networks Virtual Labs](#) and reserve your free sandbox today! You'll find the **Apstra** sandbox by scrolling down to the Switching category.

This use case demonstrates how you use Apstra to design, build, and deploy an EVPN-VXLAN data center network. Apstra is an intent-based networking and assurance solution that provides full life cycle management of your data center network, including:

- the initial design, build, and deployment of the fabric including the overlay networks
- the ongoing management of changes (intended or unplanned network events) throughout the fabric's life cycle
- the ongoing validation of your intent against the actual deployment throughout the fabric's life cycle

Juniper Apstra software is a horizontally-scalable, microservices-based application that brings intent-based networking to standard off-the-shelf switches. With Apstra, you design and deploy your data center network fabric using a declarative methodology that captures your intent. Apstra then translates this intent into commands that the fabric devices understand.

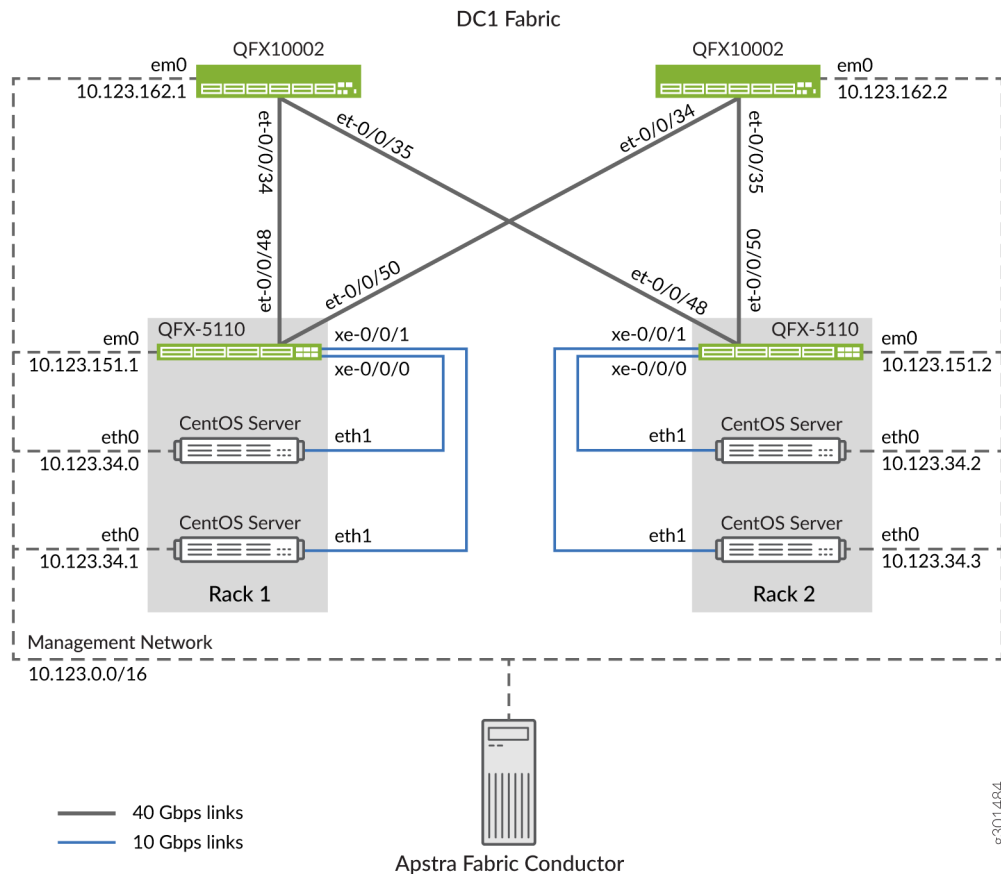
In this use case, you use Apstra to:

- design a simple 3-stage Clos network
- build a network blueprint
- deploy that blueprint to the fabric devices

Apstra checks your work every step of the way, first before you deploy to verify the soundness of your design, and again after deployment when Apstra verifies network operation and compliance with your design intent.

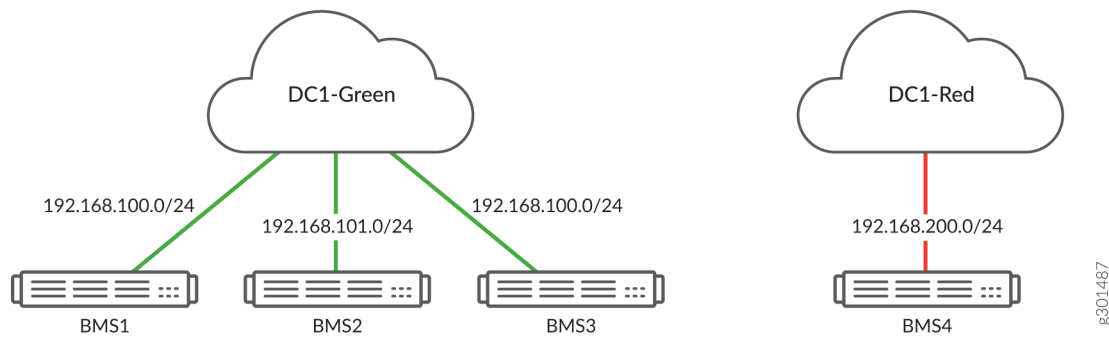
Figure 1 on page 3 shows the EVPN-VXLAN fabric you'll design, build, and deploy. The fabric consists of two QFX10002 switches acting as spine devices and two QFX5110 switches acting as leaf devices. Each leaf device is housed in a rack that also contains two servers running CentOS. The spine-to-leaf links are all 40 Gbps and the leaf-to-server links are all 10 Gbps. Apstra manages the fabric devices over the out-of-band management network.

Figure 1: DC1 Data Center Fabric



Once you set up the fabric, you'll use Apstra to create the overlay networks. Figure 2 on page 4 shows the overlay networks you'll create in this use case. DC1-Green and DC1-Red represent routing zones or segmented networks, which are implemented as VRFs on the QFX5110 switches. DC1-Green contains routes for two virtual networks (subnets 192.168.100.0/24 and 192.168.101.0/24). DC1-Red contains routes for one virtual network (subnet 192.168.200.0/24). Traffic cannot pass between the routing zones as these represent different tenants (or segmented networks) that share the same fabric. Attached to the overlay networks are the bare metal server endpoints (BMS1 through BMS4). These are the CentOS servers shown in Figure 1 on page 3. You use Apstra to configure the fabric down to the fabric edge, but you configure the actual servers themselves outside of Apstra.

Figure 2: DC1-Green and DC1-Red Overlay Virtual Networks



**NOTE:** An important Apstra feature is the ability to separate the site-specific details from the generic parts of a network design. To help you better understand this separation, we add a **DC1** prefix to the name of any resource or construct that is typically site-specific.

## Workflow

The general workflow takes you through three distinct stages: design, build, and deploy.

1. In the design stage, you specify the physical requirements for your fabric such as the types and roles of the fabric devices (for example, spines, leafs, superspines servers, etc.), the speeds and feeds of the fabric devices, how the devices are organized in racks, and how the racks connect together. The output of this stage is one or more templates that you can use as building blocks for your data center. Each template represents a standalone part of the fabric infrastructure. It can represent a row of racks, a pod, or any other construct including an entire data center.

The design stage is device-agnostic and site-agnostic. A typical use case is for you to create a template that represents a unique row or pod in your data center. You then take this single template to create blueprints for all similar rows or pods in all your data center sites.

2. The build stage consists of two main tasks: realizing your physical design and creating your overlay networks.
  - You realize your physical design by assigning site-specific resource pools and devices to your template.
  - You create your overlay networks by creating your routing zones and then adding virtual networks (subnets) to those routing zones. A routing zone translates to a VRF on the QFX Series switch.

The output of this stage is a blueprint that captures your intent for the underlay and overlay networks for a specific site.

3. In the deploy stage, Apstra translates your blueprint into configuration commands that the devices understand and pushes those commands to the devices in your fabric. Apstra then verifies the deployment against the blueprint to detect cabling errors and misbehaving devices. In effect, the blueprint acts as a reference design that Apstra uses to

constantly measure the behavior and performance of the actual network against. Any deviations from the reference design display as anomalies.

**NOTE:** To keep this use case short, we only cover the configuration of mainline parameters. You should use the default settings for parameters not detailed in this document.

## Prepare

### SUMMARY

Create agents for the fabric devices, ensure device profiles exist for those devices, and define the resource pools you want to use in your deployment.

### IN THIS SECTION

- [Explanation of Procedure | 5](#)
- [Create the Device Agents | 6](#)
- [View the Device Profiles | 8](#)
- [Define Resources | 10](#)

## Explanation of Procedure

With Apstra, you can design and build your fabric before you install the actual fabric devices in your data center. This is because you don't need to assign physical devices to your blueprint until you're ready to deploy. This decoupling of the design and build from the actual deployment gives you the freedom to work within the timelines and workflows suitable for your business. For example, you can use Apstra to design the fabric in all your data center sites before you even roll out equipment.

For expediency in this use case, however, you'll install the fabric devices before you start your fabric design. Once you install the devices, you'll create agents for these devices. An agent is an intermediary software service that translates API calls and responses between Apstra and the device. An agent can reside on the device (where supported) or on the Apstra server. There is one agent per device.

If you run with ZTP, Apstra automatically creates the agents for you. If you don't deploy with ZTP, you'll need to create these agents manually, which is what you'll do in this use case.

When you create an agent for a device, the agent reaches out to the device to determine what kind of device it is. Apstra uses device profiles for this purpose. Apstra ships with a set of profiles for all devices that are known to work with Apstra. A device profile contains information about a supported device including hardware model and supported software versions. When Apstra connects to a device for the first time, it gathers information from the device and looks for a matching device profile. Apstra allows you to create your own device profile, but this is beyond the scope of this document. For the purposes of this use case, your task is to familiarize yourself with the pre-existing device profiles that you'll use.

Next, you'll define the resource pools that you'll use for your fabric. These are the IP address ranges, the BGP autonomous system (AS) number ranges, the VNI ranges, and so forth. These resources are typically specific to a data center location.

## Create the Device Agents

1. Prepare your fabric devices. Connect to the console port on each of your fabric devices. Log in as a user with super-user access, and do the following:
  - a. Create the login credentials that you want Apstra to use to log in to the device. If you want Apstra to use an existing set of credentials, then you can skip this step. The login credentials that Apstra uses must have super-user capabilities on the device.
  - b. Configure the management IP address and subnet and connect the device to the management network over the out-of-band management interface (for example, **em0**). In this use case, the management IP addresses of the leaf devices are **10.123.151.1/16** and **10.123.151.2/16**. The management IP addresses of the spine devices are **10.123.162.1/16** and **10.123.162.2/16**.
  - c. Enable SSH and NETCONF on the device. Apstra uses NETCONF to configure the device. If you plan on having Apstra log in to the device as the root user, then be sure to enable root login over ssh on the device.

For complete information on prerequisite device configuration including how to configure a management routing instance, see [Juniper Device Agent](#).

**NOTE:** Since you haven't designed your fabric yet, you don't generally wire up your fabric at this time (other than to connect the devices to the management network). Once you finish your design and build later, you'll use Apstra to generate a cabling map for your cabling installers to follow.

2. Create an agent for each fabric device.

An agent acts as a proxy for the fabric device, translating commands from Apstra to commands that the device understands, and translating responses from the device to responses that Apstra understands.

There is one agent per fabric device. An agent can be onbox or offbox. An onbox agent resides on the device. An offbox agent resides on the Apstra server. For Junos devices, the agents are offbox.

**NOTE:** To see what each icon in the left-nav bar represents, click the square at the bottom of the left-nav bar.

- a. Log in to the Apstra UI. The default username is **admin**, the default password is **admin**.
- b. In the left-nav bar, select **Devices>System Agents>Agents** to bring up the Agents page.
- c. Select the **OFFBOX** tab at the top of the page.
- d. Click **Create Offbox Agent(s)** to bring up the Create System Agent(s) window.

- e. Fill in the required fields as shown in [Table 1 on page 7](#) and click **Create**.

**Table 1: Offbox Agent Parameters**

Agent Parameters	Description	Settings in this Use Case
Device Addresses	The management IP addresses of the devices in your fabric. Addresses can be a comma-separated list or a range.	10.123.151.1 10.123.151.2 10.123.162.1 10.123.162.2
Operation Mode	FULL CONTROL - deploys configuration and collects telemetry  TELEMETRY ONLY - collects telemetry only	FULL CONTROL
Platform	The platform that these devices belong to.	junos
Username	The username to log in to the devices.	<username>
Password	The password to log in to the devices.	<password>

**NOTE:** If you have different login credentials for all your devices, then you must create your offbox agents one at a time.

The Agents page now lists the agents that you've just created. The Connection State indicates whether the agent can connect to the device. The Job State shows the progression as the agent starts up, logs in to the device, and retrieves system information to allow Apstra to identify the device hardware and software version. Once Apstra identifies the device, it assigns a device profile to use for that device. When successful, the Job State shows SUCCESS and the Connection State shows CONNECTED. The page also displays the device software version and the host name (if configured).

**NOTE:** This step succeeds even if you have not configured NETCONF on the device. If you have not configured NETCONF properly, you'll see errors during deployment.

### 3. Acknowledge the devices.

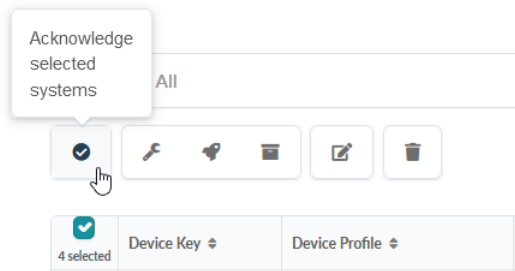
At this point, the agents are up and running and connected to the devices they represent. However, Apstra is not actively managing the devices yet. You must acknowledge the devices to bring them under Apstra management.

- a. Select **Devices>Managed Devices** in the left-nav bar to bring up the Managed Devices page.

This page shows the devices in OOS-QUARANTINED state, which means that the devices are out-of-service and not yet under Apstra management.

- b. Select all the devices using the checkboxes on the left. A toolbar appears above the table ([Figure 3 on page 8](#)).

**Figure 3: Acknowledge Systems**



- c. Click the **Acknowledge selected systems** icon. The devices all change state to OOS-READY, which means that the devices are still out-of-service but are now ready to be managed.

## View the Device Profiles

Apstra ships with a set of predefined profiles for supported devices. In most situations, you should be able to find a predefined profile for your device.


Apstra also allows you to define your own device profile in the event that a predefined profile for your device does not exist. Creating a device profile is beyond the scope of this document. For information on creating device profiles, see [Creating a Device Profile](#).

To view the predefined device profiles used in this use case:

1. Select **Devices>Device Profiles** to bring up the Device Profiles page.
2. Scroll through the list to confirm that the devices deployed in this use case are in the list. You may want to use the **Query** pull down and enter "Juniper" in the **Manufacturer** field to narrow down the list:
  - Juniper\_QFX5110-48S for the leaf devices
  - Juniper\_QFX10002-36Q for the spine devices
3. Click a device profile to see how each device is defined. [Figure 4 on page 9](#) shows the device profile for the QFX10002-36Q switch, and [Table 2 on page 9](#) provides a high level summary of the different sections in the profile.

Figure 4: Device Profile for QFX10002-36Q

## Summary

Name	Juniper_QFX10002-36Q
Modular?	no
Slot count	0
Ports preview	

Selector<sup>®</sup>

Manufacturer <sup>®</sup>	Juniper
Model <sup>®</sup>	QFX10002-36Q
OS family <sup>®</sup>	Junos
Version <sup>®</sup>	{1[8-9]}[20].*

## Capabilities

## Hardware Capabilities

CPU <sup>®</sup>	x86
Userland (bits) <sup>®</sup>	64
RAM (GB) <sup>®</sup>	32

## Supported Features

No items.

## Software Capabilities

LXC <sup>®</sup>	yes
ONIE <sup>®</sup>	no
Config Apply Support <sup>®</sup>	complete_only

Table 2: Device Profile

Section	Description
Summary	A summary of the device.
Selector	<p>The set of matching criteria that Apstra uses to determine whether a device matches this device profile. The criteria include both hardware and software matching criteria.</p> <p>As part of agent creation, Apstra reads system information from the device and looks for a matching device profile based on the matching criteria in this section.</p>
Capabilities	A summary of the hardware and software capabilities and characteristics.

4. Scroll down to the Ports section. Click a port in the port pictogram to see details, including how the interfaces are named on this device.

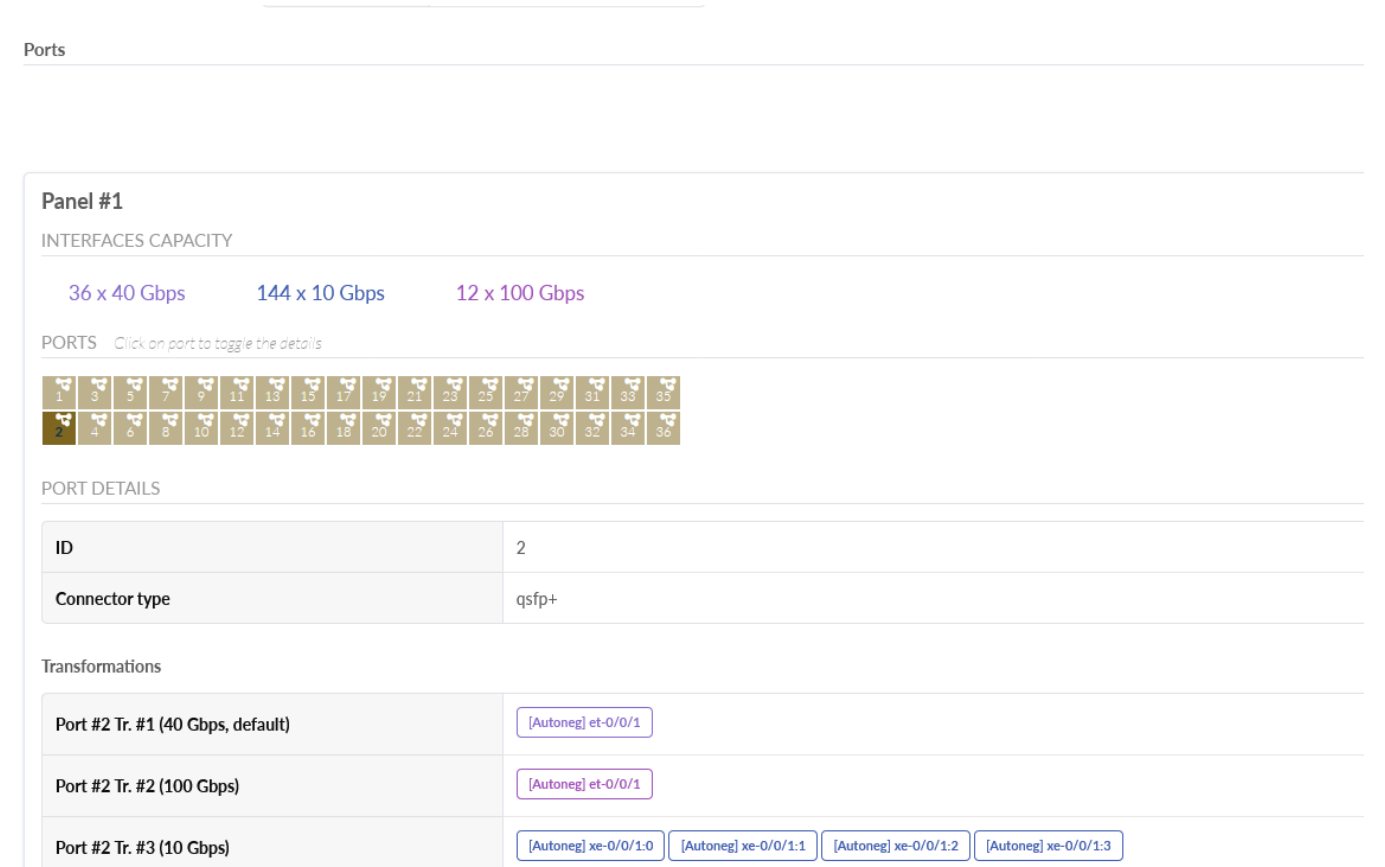
Since the interface name and speed may change depending on the capabilities of the transceiver or interface module, Apstra allows you to define mappings or transformations that capture the different personalities a port can assume.

The QFX10002-36Q has a port panel consisting of 6 cages with 6 ports per cage. Depending on the transceivers you install, you can configure any port in any cage for 10 Gbps (breakout) or 40 Gbps. Within each cage, you can install

QSFP28 transceivers in the second and sixth ports to support 100 Gbps. Apstra uses transformations to capture all this information in the device profile.

Figure 5 on page 10 shows the port panel with port 2 selected. The Transformation table confirms the port supports 10 Gbs, 40 Gbps, and 100 Gbps. If you click on port 1 you'll see that only 10 Gbps and 40 Gbps are supported.

Figure 5: QFX10002-36Q Port Transformations



## Define Resources

Define the resources specific to your data center site. In general, each data center site has a different resource definition.

1. Specify the IP pools to use for your fabric.
  - a. Select **Resources>IP Pools** in the left-nav bar to bring up the IP Pools page.
  - b. Click **Create IP Pool** to create the IP pools for your network.

The Create IP Pool window appears.

- c. Fill in the required fields to create the IP pool for the fabric loopback interfaces ([Table 3 on page 11](#)) and click **Create**. These are the loopback IP addresses used for establishing BGP peering in the overlay.

**Table 3: IP Pool for Fabric Loopback Interfaces**

Parameter	Description	Setting in this Use Case
Name	A unique name for this IP pool.	DC1-Loopback-IP
Subnets	The IP subnet to use.	10.255.0.0/24
Create Another	Select to create another IP address pool.	Checked

- d. Fill in the required fields to create the IP pool for the fabric interfaces interconnecting the switches ([Table 4 on page 11](#)) and click **Create**. These are the IP addresses for the *et* interfaces in [Figure 1 on page 3](#). These addresses are used for BGP peering in the underlay.

**Table 4: IP Pool for Fabric Interfaces**

Parameter	Description	Setting in this Use Case
Name	A unique name for this IP pool.	DC1-Intra-Fabric-IP
Subnets	The IP subnet to use.	10.10.0.0/24
Create Another	Select to create another IP address pool.	Checked

2. Specify the IP pools to use for your per-overlay network loopback addresses. In this use case, you'll create routing zones (or VRFs) called DC1-Green and DC1-Red representing two overlay networks. In this step, you're defining the loopback IP address pools for the two VRFs.

- a. Fill in the required fields to create the IP pool for the DC1-Green overlay loopback interfaces ([Table 5 on page 11](#)) and click **Create**.

**Table 5: IP Pool for Overlay Network (DC1-Green)**

Parameter	Description	Setting in this Use Case
Name	A unique name for this IP pool.	DC1-Green-Loopback-IP
Subnets	The IP subnet to use.	10.192.0.0/24

**Table 5: IP Pool for Overlay Network (DC1-Green) (Continued)**

Parameter	Description	Setting in this Use Case
Create Another	Select to create another IP address pool.	Checked

- b. Fill in the required fields to create the IP pool for the DC1-Red overlay loopback interfaces ([Table 6 on page 12](#)) and click **Create**.

**Table 6: IP Pool for Overlay Network (DC1-Red)**

Parameter	Description	Setting in this Use Case
Name	A unique name for this IP pool.	DC1-Red-Loopback-IP
Subnets	The IP subnet to use.	10.193.0.0/24
Create Another	Select to create another IP address pool.	—

3. Specify the AS number pools to use for your fabric. Apstra assigns each switch its own AS number. The assigned AS number is used to support the BGP peering used in the underlay. For the example network you only need four AS numbers. There is no harm in specifying a pool that is larger than needed in the event you expand the fabric at a later time.
- Click **Resources>ASN Pools** in the left-nav bar to bring up the ASN Pools page.
  - Click **Create ASN Pool** to create the ASN pool for your fabric.  
The Create ASN Pool window appears.
  - Fill in the required fields ([Table 7 on page 12](#)) and click **Create**.

**Table 7: ASN Pool**

Parameter	Description	Setting in this Use Case
Name	A unique name for this ASN pool.	DC1-ASN
Ranges	The range of ASNs to use.	65000 - 65099

4. Specify the virtual network identifier (VNI) pools to use for your fabric. VNIs are used in VXLAN encapsulation to provide layer 2 separation for the overlay traffic in your fabric.
- Select **Resources>VNI Pools** in the left-nav bar to bring up the VNI Pools page.
  - Click **Create VNI Pool** to create the VNI pool for your fabric.

The Create VNI Pool window appears.

- c. Fill in the required fields ([Table 8 on page 13](#)) and click **Create**.

**Table 8: VNI Pool**

Parameter	Description	Setting in this Use Case
Name	A unique name for this VNI pool.	DC1-VNI
Ranges	The range of VNIs to use.	5000 - 9999

## Design

### SUMMARY

Design your fabric using generic building blocks that are free of site-specific details and site-specific hardware. The output of this stage is a template that you later use in the Build stage to create blueprints for all your data center locations.

### IN THIS SECTION

- [Explanation of Procedure | 13](#)
- [Define the Logical Devices | 14](#)
- [Define Interface Maps | 15](#)
- [Define Racks | 17](#)
- [Define Templates | 19](#)

## Explanation of Procedure

In the design stage, you use progressively larger building blocks to construct your network. You start by defining the devices, then the racks that hold these devices, and finally the templates that describe how these racks interconnect. The template definition is very flexible. A template represents a standalone part in your data center such as a row of racks, a pod, or the entire data center network. In this use case, the template represents your entire data center network.

A template is not specific to a site. It does not contain site-specific details such as IP addresses or AS numbers, nor does it contain any hardware-specific information. The idea is that you can use this same template to create blueprints for all your data center locations, regardless of what IP addresses and physical equipment you use at the different locations.

## Define the Logical Devices

Create the logical devices that represent the leaf and the spine switches and servers. Logical devices are abstractions of physical devices. In effect, by creating logical devices, you're specifying the minimum set of requirements for your hardware. At a later stage when you select the actual devices for your fabric, Apstra will only allow you to select the physical switches and servers that meet these requirements.

With this approach, your data center design is not dependent on the deployment of any specific physical device. This abstraction reduces device dependency, facilitates equipment inventory management, and promotes design reuse. For example, if a particular device fails in your network, you don't have to replace the failed device with the exact same model. You can replace it with a different model as long as the new model meets the specified requirements. You simply adjust your blueprint to use that model. Your network design remains unchanged.

Apstra ships with a predefined set of logical devices that you can use in many situations. In fact, you won't need to create your own logical devices most of the time. In this use case, however, for illustration purposes, we'll show you how to create your own logical devices for the leaf and the spine switches. You'll use a predefined server definition for the server endpoints (AOS-1x10-1).

**NOTE:** You're creating logical device types and not logical device instances. In other words, if your data center has one type of leaf switch deployed across numerous racks, then you're creating a single logical device representing that type of leaf switch. You're not creating each instance of that leaf switch yet. You create instances when you define the rack and template.

1. Select **Design>Logical Devices** to open the Logical Devices page.
2. Create the logical device representing the leaf switch. In this use case, the leaf device has 2 x 10 Gbps ports connecting to servers, and 2 x 40 Gbps ports connecting to spine devices.
  - a. Click **Create Logical Device**. The Create Logical Device window appears.
  - b. Specify a meaningful **Name** for this logical device (for example, **AOS-2x10+2x40**).
  - c. Configure the 2 x 10 Gbps ports on this device to connect to the servers.
    - **PANEL #1** shows a 24-port pictogram. Drag the corner of the pictogram to form 2 ports. By default, the ports operate at 10 Gbps, which is what you want.
    - Specify that these ports connect to L2 servers by selecting **Access** and **Generic** in the Connected To section. In this context, access indicates ports that attach to a workload and generic indicates a device not managed by Apstra.
    - Click **Create Port Group** to save the access port definition.
  - d. Configure the 2 x 40 Gbps ports on this device to connect to the spine devices.
    - Click **Add Panel** to add **PANEL #2**.
    - In **PANEL #2**, drag the corner of the 24-port pictogram to form 2 ports.
    - Use the drop-down list to set the **Speed** to 40 Gbps.

- Specify that these ports connect to spine devices by selecting **Spine** in the Connected To section.
  - Click **Create Port Group**.
- e. Click **Create**.
3. Create the logical device representing the spine switch. The spine devices have 2 x 40 Gbps ports connecting to leaf devices.
- a. Click **Create Logical Device**. The Create Logical Device window appears.
  - b. Specify a meaningful **Name** for this logical device (for example, **AOS-2x40**).
  - c. Configure the 2 x 40 Gbps ports on this device to connect to the leaf devices.
    - In PANEL #1, drag the corner of the 24-port pictogram to form 2 ports.
    - Use the drop-down list to set the **Speed** to 40 Gbps.
    - Specify that these ports connect to leaf devices by selecting **Leaf** in the Connected To section.
    - Click **Create Port Group**.
  - d. Click **Create**.

**NOTE:** At this stage, no correlation exist between the port numbers in the pictograms and the port numbers on the actual physical devices. All you've specified so far is that the leaf device has two 10 Gbps ports connecting to servers and two 40 Gbps ports connecting to spine devices, and that the spine devices have two 40 Gbps ports connecting to the leaf devices.

## Define Interface Maps

An interface map associates a physical device with a logical device and includes the mapping of ports from one to the other. In this part of the design process, you're pairing the physical devices you want to deploy in your fabric with the logical devices in your design. You're not actually selecting your physical devices in this step. You're just creating a library of pairings that you'll pick from later on.

If you have different candidates for the actual device hardware (for example, different QFX Series switch models that you want to use for your leaf devices), then you'll create interface maps that pair each candidate with the logical device in your design. Later, when you build your blueprint, you'll select the QFX Series switch model you want to deploy and then Apstra picks the corresponding interface map.

Apstra ships with a predefined set of interface maps that match the predefined set of logical devices. In many situations, you should be able to pick from this predefined list instead of creating your own. In this use case, however, since you've created your own logical devices, you'll need to create your own interface maps for those devices. You'll create one interface map for the spine device and one interface map for the leaf device.

1. Select **Design>Interface Maps** to open the Interface Maps page.

2. Create the interface map for the leaf device. The interface map you create in this step maps the physical leaf switch you want to use in your fabric (QFX5110-48S) to the logical device representing the leaf switch in your design (AOS-2x10+2x40).
  - a. Click **Create Interface Map**. The Create Interface Map window appears.
  - b. Specify a meaningful **Name** for this interface map (for example, **Juniper\_QFX5110-48S\_\_\_\_AOS-2x10+2x40**). The suggested name clearly indicates the association between the physical device you want to use and the logical device in your design.
  - c. Use the **Logical device** drop-down list to select the logical leaf device you created earlier (**AOS-2x10+2x40**). The Create Interface Map page appears.
  - d. Use the **Device profile** drop-down list to select the model of the actual physical leaf switch (**Juniper\_QFX5110-48S**). You can filter entries in the drop-down list based on name by typing any characters in the name of the device profile (for example, "Juniper"). Apstra only allows you to select device profiles that meet the logical device requirements. For example, if your logical device has two 40 Gbps ports, Apstra only allows you to select device profiles that have at least two 40 Gbps ports.

- e. Specify which ports connect to the servers.

In the 10 Gbps row in the **Map interfaces** table, click **Select interfaces**. A port pictogram representing the QFX5110-48S appears.

Recall from [Figure 1 on page 3](#) that you're using the xe-0/0/0 and xe-0/0/1 interfaces to connect to the servers. Therefore, select port 1 and port 2 in the pictogram.

**NOTE:** Apstra begins indexing interfaces at 1 while the QFX Series switches begin their numbering at 0. The result is that port 1 and port 2 map to xe-0/0/0 and xe-0/0/1 respectively.

- f. Specify which ports connect to the spine devices.

In the 40 Gbps row in the Map interfaces table, click **Select interfaces**. A port pictogram representing the QFX5110-48S appears.

Recall from [Figure 1 on page 3](#) that you're using xe-0/0/48 and xe-0/0/50 to connect to the spine devices. Select port 49 and port 51, representing xe-0/0/48 and xe-0/0/50 interfaces, respectively.

- g. Click **Create** to save the changes to your interface map.

3. Create the interface map for the spine device. The interface map you are creating in this step maps the physical spine switch you want to use in your fabric (QFX10002-36Q) to the logical device representing the spine switch in your design (AOS-2x40).
  - a. Click **Create Interface Map**. The Create Interface Map window appears.
  - b. Specify a meaningful **Name** for this interface map (for example, **Juniper\_QFX10002-36Q\_\_\_\_AOS-2x40**).
  - c. Use the **Logical device** drop-down list to select the logical spine device you created earlier (**AOS-2x40**).
  - d. Use the **Device profile** drop-down list to select the model of the physical spine switch (**Juniper\_QFX10002-36Q**).
  - e. Specify which ports connect to the leaf devices.

In the 40 Gbps row in the Map interfaces table, click **Select interfaces**. A port pictogram representing the QFX10002-36Q appears.

Recall from [Figure 1 on page 3](#) that you're using et-0/0/34 and et-0/0/35 to connect to the leaf devices. Select port 35 and port 36, representing the et-0/0/34 and et-0/0/35 interfaces respectively.

- f. Click **Create** to save the changes to your interface map.

## Define Racks

You've defined the first set of building blocks for your data center design, namely the logical devices that represent your hardware. You're now ready to define the next building block, which is the rack that houses these logical devices.

A rack represents a physical rack in the data center. It consists of one or more Top-of-Rack (ToR) switches and a bank of servers. A ToR switch and a leaf switch are different names for the same device.

Apstra ships with a set of predefined racks that cover many common deployments, but these rack types are too elaborate for this basic use case. In this example, you'll define a rack that consists of one leaf (ToR) device and two servers.

**NOTE:** You're creating rack types and not rack instances. In other words, if your data center has one type of rack deployed, then you're creating a single rack type. You're not creating each instance of that rack yet. You create instances when you define the template.

1. Select **Design>Rack Types** to open the Rack Types page.
2. Create the rack you'll use in your design.
  - a. Click **Create Rack Type**. The Create Rack Type window appears.
  - b. Specify a meaningful name for the rack (for example, **L2 1L2S**).
  - c. Specify a description for the rack (for example, **Layer 2, one leaf switch and two servers**).
  - d. Set the **Fabric connectivity design** to **L3 Clos**. This setting indicates a conventional leaf and spine EVPN-VXLAN fabric.
3. Configure the leaf device.
 

The Configuration section contains 3 tabs: **Leafs**, **Access Switches**, and **Generic Systems**.

  - a. Select the **Leafs** tab.
  - b. Fill in the required fields ([Table 9 on page 18](#)).

**Table 9: Leaf Switch Configuration for the Rack**

Parameter	Description	Setting in this Use Case
Name	<p>The name of this leaf switch.</p> <p>The name you want to give to the leaf switch in the design. This name is <b>not</b> the hostname of any physical switch in your fabric. In the build stage, you will assign the actual physical switch to represent this leaf switch.</p>	1L2S-Leaf
Leaf Logical Device	<p>The logical device that you want to use as the leaf switch.</p> <p>Based on this selection, Apstra limits the selectable values in the remaining fields to what this logical device supports.</p>	AOS-2x10+2x40
Links per spine	The number of links connecting this leaf switch to any single spine switch.	1
Link speed	The speed of the link connecting this leaf switch to any single spine switch.	40 Gbps

#### 4. Configure the servers.

- a. In the Configuration section, select the **Generic Systems** tab.
- b. Click **Add new generic system group**.
- c. Fill in the required fields ([Table 10 on page 18](#)).

**Table 10: Server Configuration for the Rack**

Parameter	Description	Setting in this Use Case
Name	<p>The name of the servers.</p> <p>The name you want to give to the servers in the design. This name is not the hostname of any physical server.</p>	1L2S-Server
Generic system count	The number of generic systems (servers) in the rack.	2
Logical Device	<p>The logical device that you want to use as the server.</p> <p>Based on this selection, Apstra limits the selectable values in the remaining fields to what this logical device supports.</p>	AOS-1x10-1

Table 10: Server Configuration for the Rack *(Continued)*

Parameter		Description	Setting in this Use Case
Link (click <b>Add logical link</b> )	Name	The name of the link.	1L2S-Leaf-to-Server
	Switch	The leaf switch connected to this server.  If you've added only one leaf switch to this rack, as in this use case, then Apstra pre-populates this field with the name of that leaf switch.	1L2S-Leaf
	Physical link count per leaf	The number of links connecting this server to the leaf switch.	1
	Link speed	The link speed between this server and the leaf switch.	10 Gbps

**NOTE:** As you enter information, Apstra draws a preview of the rack based on the information you enter. The diagram allows you to better visualize the rack you're creating.

- d. Click **Create**.

## Define Templates

You've now created your logical devices and the racks that hold these logical devices. You're now ready to define the next building block, which is the template that describes how the racks connect together.

A template can represent a row or a Point of Delivery (POD) in your data center. In this use case, the template represents the entire data center, which consists of two racks.

1. Select **Design>Templates** to open the Templates page.
2. Click **Create Template**. The Create Template window appears.
3. Complete the required fields according to [Table 11 on page 19](#).

Table 11: Create Template

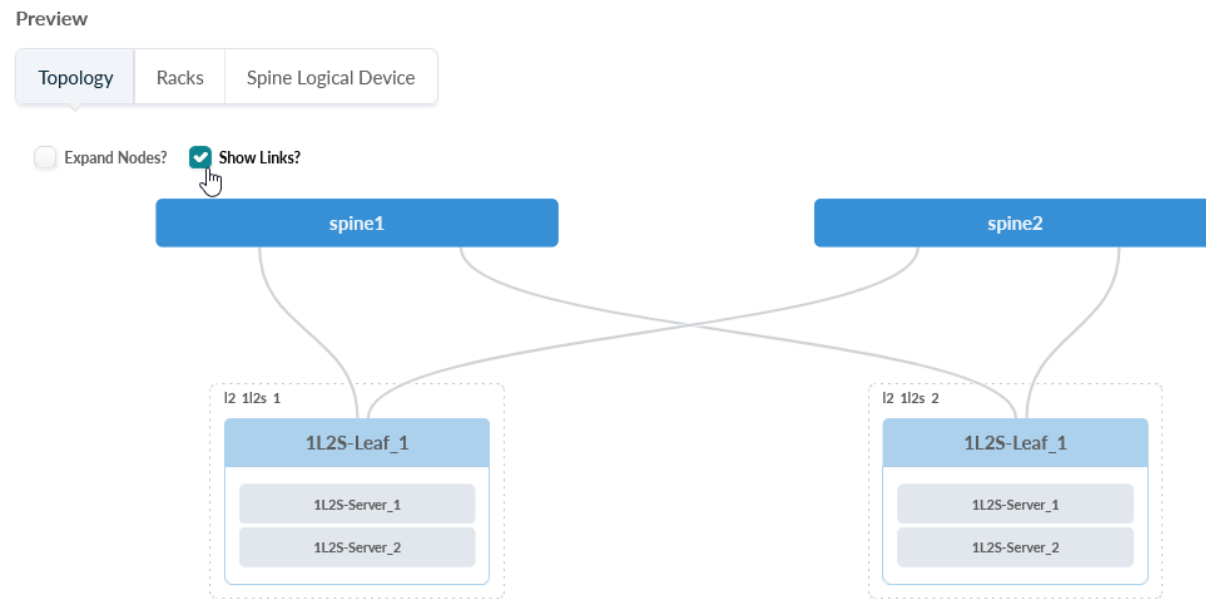
Parameter	Description	Setting in this Use Case
Common Parameters>Name	The name you want to call this template.	L2 Pod 2x2

Table 11: Create Template (*Continued*)

Parameter	Description	Setting in this Use Case
Common Parameters>Type	RACK BASED for creating a set of racks (for a 3-stage Clos).  POD BASED for creating a set of pods (for a 5-stage Clos).	RACK BASED
Policies>Overlay Control Protocol	Static VXLAN for static tunnels.  MP-EBGP EVPN for dynamic VXLAN tunnels.	MP-EBGP EVPN
Structure>Rack Types	Select the rack type and the number of instances from the drop-down list. This use case consists of two instances of the same rack type.  If your design has multiple different rack types, then simply click <b>Add racks</b> to add them.	L2 1L2S  2 instances
Structure>Spines>Spine Logical Device	Select the logical device for the spine from the drop-down list. The drop-down list shows all logical spine devices. If you select a logical spine device that is incompatible with your design, Apstra will display an error when you try to create the template.	AOS-2x40
Structure>Spines>Count	The number of spine switches.	2

**NOTE:** Apstra draws a preview of the design based on the information you enter. This allows you to better visualize the template you're creating. Click **Expand Nodes** and/or **Show Links** to show more detail as desired. See [Figure 6 on page 21](#).

Figure 6: Template Preview



Apstra automatically appends an instance number to the names of the devices as follows:

- spines - Apstra automatically names the spine device as *spine* and appends an instance value to represent the instances. The format is *spine<instance>* (for example, **spine1**).
- racks - Apstra appends a rack instance value to the rack type you specified. The format is *<rack type> <instance>* (for example, **l2 1l2s 1**).
- leafs - Apstra appends a leaf instance value to the name of the leaf device in this rack type. The format is *<leaf name>\_<instance>* (for example, **1L2S-Leaf\_1**). The instance value represents the instance of the leaf device within the rack.
- servers - Apstra appends a server instance value to the name of the server in this rack type. The format is *<server name>\_<instance>* (for example, **1L2S-Server\_1**). The instance value represents the instance of the server within the rack.

4. When finished click **Create** to save your template.

Your design is now complete! You've created the logical devices, the racks that hold these logical devices, and finally the template that describes how these racks interconnect. You're now ready to enter the build phase.

# Build

## SUMMARY

Create and build a site-specific blueprint from your generic template. The blueprint contains site details such as the hardware to deploy and the resource pools to use.

## IN THIS SECTION

- [Explanation of Procedure | 22](#)
- [Create the Blueprint | 22](#)
- [Build the Blueprint - Physical | 23](#)
- [Build the Blueprint - Virtual | 29](#)

## Explanation of Procedure

In the build stage, you add site-specific details to your template to create a blueprint. While your template is generic and reusable across multiple data centers, a blueprint is specific to a given data center site.

The blueprint contains site-specific details for both your physical fabric and your virtual overlay networks.

In the blueprint, you assign IP addresses and other network resources like AS numbers, and you specify the actual hardware to use in your physical underlay network. You also create your overlay networks, as defined by routing zones and the virtual networks running in those zones. Since overlay networks are part of the blueprint, it's expected that you'll regularly change the virtual part of your blueprint as your overlay connectivity requirements change.

## Create the Blueprint

1. Select **Blueprints** in the left-nav bar to open the Blueprints page.
2. Click **Create Blueprint**. The Create Blueprint window appears.
3. Type in the **Name** you want to call this blueprint. The name should distinguish this data center site from your other data center sites (for example **DC1-Fabric**).
4. Use the **Template** drop-down list to select the template you created earlier (for example, **L2 Pod 2x2**). Once you select a template, Apstra provides an Intent preview to help you visualize the data center fabric you are building.
5. Click **Create** to save the blueprint.

The new blueprint displays on the Blueprints page.

## Build the Blueprint - Physical

This part of the blueprint covers the underlay network. It is at this stage that you specify the actual physical devices you want to use in your site. This portion of the blueprint also specifies the loopback IP address assignments, the IP address assignments for fabric ports, and the BGP AS numbers used for peering in both the underlay and overlay.

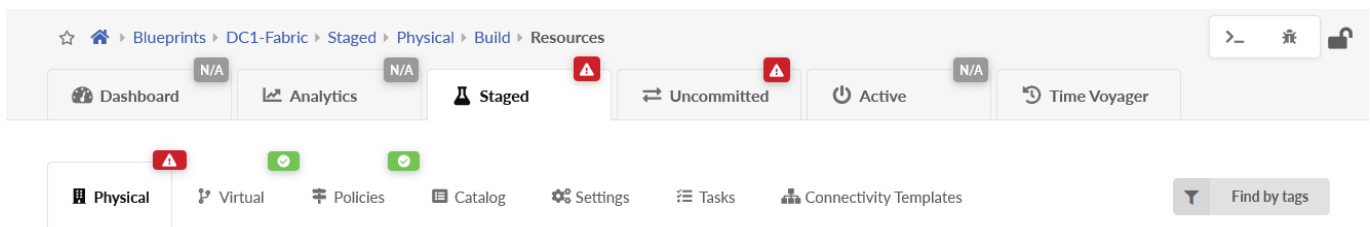
1. On the Blueprints page, click the blueprint you want to build (for example, **DC1-Fabric**).

The blueprint Dashboard page opens. Since you haven't deployed your blueprint yet, the Dashboard page is empty.

2. Click the **Staged** tab. This is where you stage your blueprint with site-specific details.

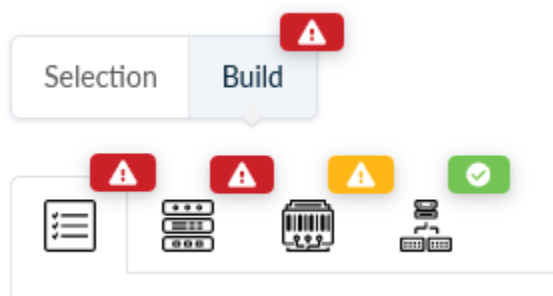
The Staged page consists of a row of tabs at the top (Figure 7 on page 23). Once you select a tab, the rest of the page is placed in context with that selection. By default, the **Physical** tab is selected.

Figure 7: Blueprint - Staged



When the **Physical** tab is selected, the right-most pane shows various icons with color-coded warnings under the **Build** tab (Figure 8 on page 23).

Figure 8: Physical Build



Hover over each icon to see what each represents (from left to right):

- Resources
- Device Profiles
- Devices
- Configlets (not used in this use case).

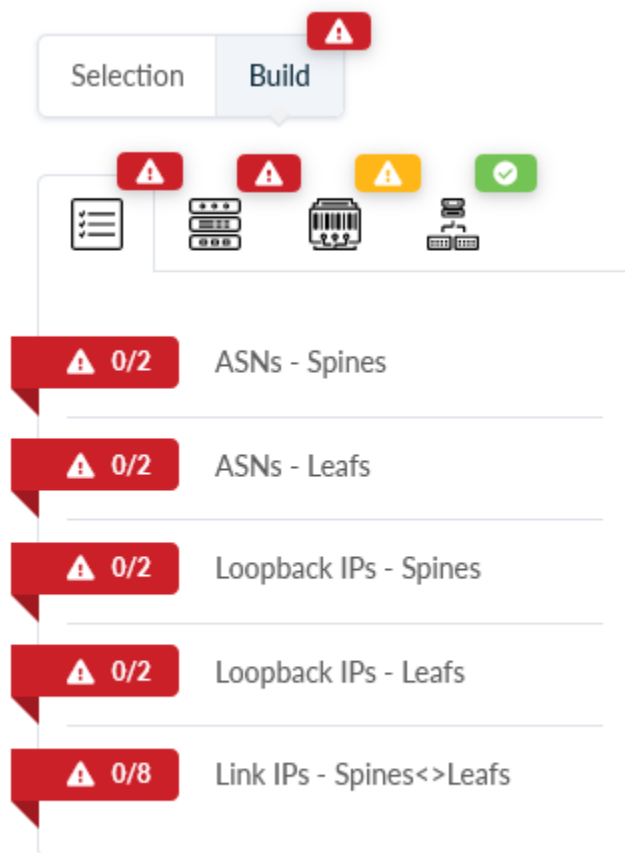
**NOTE:** Configlets allow you to append Junos configuration statements to the configuration pushed by Apstra. A typical use is to add a custom login banner that warns the device is under Apstra management.

When you select an icon, the area below the icons are placed in context with that selection. By default, the **Resources** icon is selected.

3. With the **Physical** tab and **Resources** icon selected (Figure 9 on page 24), specify the resource pools to use.

As you progress through this step, the red warning icons turn green to indicate that you've successfully assigned the respective resource.

Figure 9: Physical Build - Resources



**NOTE:** In the steps below, when you click the **Update assignments** icon, you might need to use the pagination chevrons to find your selection. Each page displays a maximum of 5 entries.

- a. Click the red warning icon to the left of **ASNs - Spines** to expand the selection and click the **Update assignments** icon in the expanded section.

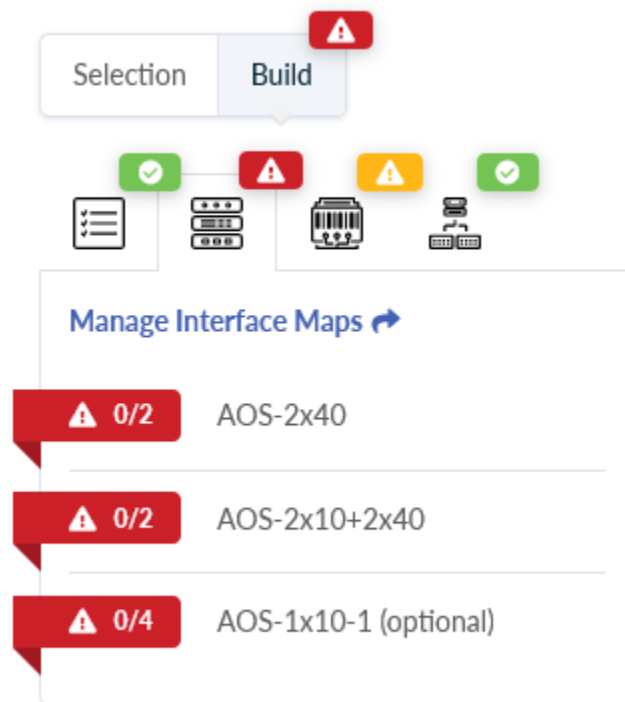
- b. Select the **DC1-ASN** pool you created earlier and click the **Save** icon. This icon looks like a floppy computer disk.
- c. Click the red warning icon to the left of **ASNs - Leafs** to expand the selection and click the **Update assignments** icon in the expanded section.
- d. Select the **DC1-ASN** pool again and click the **Save** icon.
- e. Click the red warning icon to the left of **Loopback IPs - Spines** to expand the selection and click the **Update assignments** icon in the expanded section.
- f. Select the **DC1-Loopback-IP** pool you created earlier and click the **Save** icon.
- g. Click the red warning icon to the left of **Loopback IPs - Leafs** to expand the selection and click the **Update assignments** icon in the expanded section.
- h. Select the **DC1-Loopback-IP** pool again and click the **Save** icon.
- i. Click the red warning icon to the left of **Link IPs - Spines<>Leafs** to expand the selection and click the **Update assignments** icon in the expanded section.
- j. Select the **DC1-Intra-Fabric-IP** pool you created earlier and click the **Save** icon.

At this time all the physical resource tab icons should be green to indicate successful resource assignments.

4. Click the **Device Profiles** icon to select the device profiles to use ([Figure 10 on page 26](#)).

This is where you tell Apstra about the device hardware models and device OS versions you want to deploy. You do this by telling Apstra what interface maps you want to use for the logical devices in your template. Recall that the interface map associates a device profile (hardware model and allowed OS versions) to a logical device. By specifying the interface map, you're selecting the device hardware model and allowed OS versions to use for the selected logical device.

Figure 10: Physical Build - Device Profiles



- Click the red warning icon to the left of **AOS-2x40** to expand the selection and click the **Change interface maps assignments** icon in the expanded section. The Update interface map for AOS-2x40 window appears.
- For both spine1 and spine2, use the drop-down lists to select **Juniper\_QFX10002-36Q\_\_\_AOS-2x40**. The drop-down lists shows all the defined interface maps for the AOS-2x40 logical device. In this use case, you created only one interface map for this logical device, so that's the only selection available. By selecting this interface map, you're indicating that you want to use the QFX10002-36Q switch for the spine devices in this fabric.
- Click **Update Assignments**.
- Click the red warning icon to the left of **AOS-2x10+2x40** to expand the selection and click the **Change interface maps assignments** icon in the expanded section. The Update interface map for AOS-2x10+2x40 window appears.
- For both I2\_1I2s\_001\_leaf1 and I2\_1I2s\_002\_leaf1, use the drop-down lists to select **Juniper\_QFX5110-48S\_\_\_AOS-2x10+2x40**. The drop-down list shows all the defined interface maps for the AOS-2x10+2x40 logical device. In this use case, you created only one interface map for this logical device, so that's the only selection available. By selecting this interface map, you're indicating that you want to use the QFX5110-48S switch for the leaf devices in this fabric.

**NOTE:** In the blueprint, the names of the leaf devices differ from the template. The main difference is that Apstra does not use the name you gave to the leaf device when you defined the rack. Instead, the format is `<rack name>_<rack instance>_leaf<leaf instance>` (see [Table 12 on page 30](#)).

- Click **Update Assignments**.

- g. Click the red warning icon to the left of **AOS-1x10-1** to expand the selection and click the **Change interface maps assignments** icon in the expanded section. The Update interface map for AOS-1x10-1 window appears.
- h. For all servers, use the drop-down lists to select **Generic\_Server\_1RU\_1x10G\_Centos\_AOS-1x10-1**. Recall that you're using the predefined AOS-1x10-1 logical device for your servers. This logical device has multiple predefined interface maps. The Generic\_Server\_1RU\_1x10G\_Centos\_AOS-1x10-1 interface map associates a CentOS server to this device.

**NOTE:** In the blueprint, the names of the servers differ from the template. The main difference is that Apstra does not use the name you gave to the server when you defined the rack. Instead, the format is `<rack name>_<rack instance>_sys<server instance>` (see [Table 12 on page 30](#)).

- i. Click **Update Assignments**.
- j. (Optional) Click the **Links** tab to display the links that Apstra is creating based on your interface map assignments ([Figure 11 on page 27](#)). Recall that the interface maps contain information on how the ports are used in your fabric. When you created the interface maps earlier, you specified that the leaf devices use ports et-0/0/48 and et-0/0/50 to connect to your spine devices, and your spine devices use ports et-0/0/34 and et-0/0/35 to connect to your leaf devices. Note that you did not specify which specific port connects to which specific port. Apstra makes those assignments automatically when you assign an interface map to a device.

**Figure 11: Physical Links Table**

Topology

Nodes

Links

Racks

Layer

Uncommitted Changes

Has Uncommitted Changes

Selected Rack

All

1-8 of 8

<<

<

1

>

>>

Columns (12/16)

Page Size: 25

<div><input type="checkbox"/></div> <div>0 selected</div>	Name	Role	Speed	Tags	Endpoint 1				Endpoint 2			
					Name	Role	Interface	IPv4	Name	Role	Interface	IPv4
<input type="checkbox"/>	spine1<->l2_1l2s_001_leaf1[1]	Spine to Leaf	40G		spine1	Spine	et-0/0/34	10.10.0.0/31	l2_1l2s_001_leaf1	Leaf	et-0/0/48	10.10.0.1/31
<input type="checkbox"/>	spine1<->l2_1l2s_002_leaf1[1]	Spine to Leaf	40G		spine1	Spine	et-0/0/35	10.10.0.2/31	l2_1l2s_002_leaf1	Leaf	et-0/0/48	10.10.0.3/31
<input type="checkbox"/>	spine2<->l2_1l2s_001_leaf1[1]	Spine to Leaf	40G		spine2	Spine	et-0/0/34	10.10.0.4/31	l2_1l2s_001_leaf1	Leaf	et-0/0/50	10.10.0.5/31
<input type="checkbox"/>	spine2<->l2_1l2s_002_leaf1[1]	Spine to Leaf	40G		spine2	Spine	et-0/0/35	10.10.0.6/31	l2_1l2s_002_leaf1	Leaf	et-0/0/50	10.10.0.7/31
<input type="checkbox"/>	l2_1l2s_001_leaf1<->l2_1l2s_001_sys001(1l2s-Leaf-to-Server)[1]	To Generic System	10G		l2_1l2s_001_leaf1	Leaf	xe-0/0/0	N/A	l2_1l2s_001_sys001	Generic System	eth0	N/A
<input type="checkbox"/>	l2_1l2s_001_leaf1<->l2_1l2s_001_sys002(1l2s-Leaf-to-Server)[1]	To Generic System	10G		l2_1l2s_001_leaf1	Leaf	xe-0/0/1	N/A	l2_1l2s_001_sys002	Generic System	eth0	N/A
<input type="checkbox"/>	l2_1l2s_002_leaf1<->l2_1l2s_002_sys001(1l2s-Leaf-to-Server)[1]	To Generic System	10G		l2_1l2s_002_leaf1	Leaf	xe-0/0/0	N/A	l2_1l2s_002_sys001	Generic System	eth0	N/A

In this example, you can see that Apstra has assigned port et-0/0/48 on each leaf device to connect to the spine1 device. In like fashion Apstra has assigned port et-0/0/50 on each leaf device to connect to the spine2 device.

These port assignments are purely arbitrary. The table also confirms the server to leaf attachments. For example, the xe-0/0/0 interface on Leaf1 attaches to the first BMS named "I2\_1I2s\_001\_sys001".

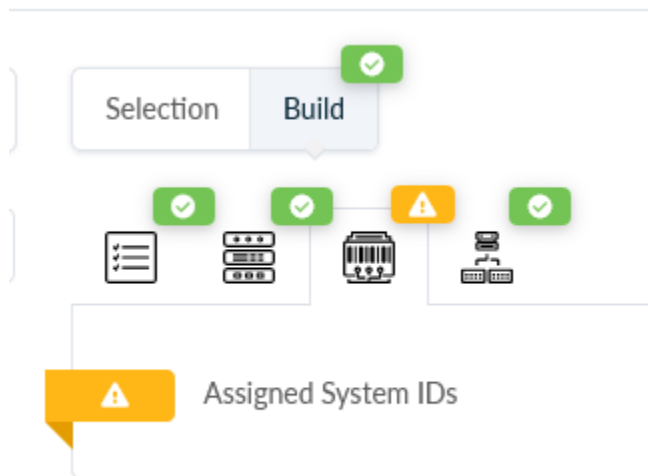
If you've already attached cables to your switches, then there's a chance that these arbitrary port assignments do not match your actual wiring. Later in the Deploy stage, we'll show you how you can override these assignments to match an existing cabling.

As you progress through this step, the red warning icons turn green to indicate that you've successfully assigned the respective device profile.

5. Click the **Devices** icon to assign the actual devices you want to use to your blueprint (Figure 12 on page 28).

These are the devices you specified when you created the device agents. Apstra only allows you to assign devices that match the interface maps you picked in the previous step. Once you assign a device, that device is no longer available for assignment in this blueprint or any other blueprint. In this way, Apstra prevents you from double booking a device by mistake.

**Figure 12: Physical Build - Devices**



**NOTE:** You can defer this step if you haven't installed your physical devices yet. You only need to assign devices when you are ready to deploy the blueprint. The ability to stage a data center deployment, before you have a data center to deploy to, is a key characteristic of the Apstra solution.

- a. Click the amber warning icon to the left of **Assigned System IDs** to expand the entry and click the **Change System IDs assignments** icon in the expanded section. The Assign Systems window appears.
- b. For spine1, use the drop-down list to select **10.123.162.1** and ensure you set the Deploy Mode to **Deploy**.
- c. For spine2, use the drop-down list to select **10.123.162.2** and ensure you set the Deploy Mode to **Deploy**.
- d. For I2\_1I2s\_001\_leaf1, use the drop-down list to select **10.123.151.1** and set the Deploy Mode to **Deploy**.
- e. For I2\_1I2s\_002\_leaf1, use the drop-down list to select **10.123.151.2** and set the Deploy Mode to **Deploy**.

- f. Click **Update Assignments**.
- g. (Optional) Click the **Nodes** tab to display the nodes that Apstra is creating based on your device assignments (Figure 13 on page 29). You can see the device names, serial numbers, loopback IP addresses, AS numbers, and other information here. The table shown here is truncated to show only one of the four servers.

**Figure 13: Physical Nodes Table**

	Name	Tags	Role	Deploy Mode	Device Profile	Hostname	ASN	Loopback IPv4
<input type="checkbox"/>	spine1		Spine	Deploy	Juniper_QFX10002-36Q	spine1	65000	10.255.0.0/32
<input type="checkbox"/>	spine2		Spine	Deploy	Juniper_QFX10002-36Q	spine2	65001	10.255.0.1/32
<input type="checkbox"/>	I2-1l2s-001-leaf1		Leaf	Deploy	Juniper_QFX5110-48S	I2-1l2s-001-leaf1	65002	10.255.0.2/32
<input type="checkbox"/>	I2-1l2s-002-leaf1		Leaf	Deploy	Juniper_QFX5110-48S	I2-1l2s-002-leaf1	65003	10.255.0.3/32
<input type="checkbox"/>	I2-1l2s-001-sys001		Generic System	Not assigned	Generic_Server_1RU_1x10G_Centos	I2-1l2s-001-sys001	Not assigned	Not assigned

**NOTE:** You don't need to update assignments for the servers when you're connecting the servers to a leaf device using layer 2. When you connect a server to a leaf device using layer 2, you manage the server outside of Apstra. Because the servers don't have an ID, the **Assigned Systems IDs** warning icon remains amber. You can safely ignore this warning.

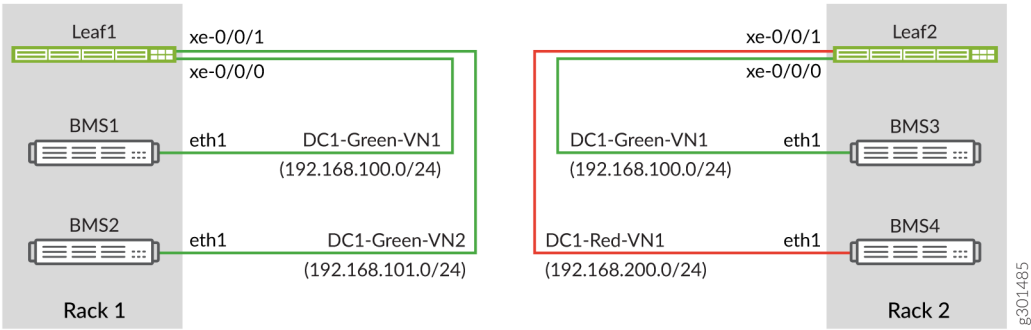
At this point, Apstra has sufficient information to configure your underlay network. You can now proceed to define your overlay networks.

## Build the Blueprint - Virtual

This part of the blueprint covers the overlay networks that run on top of the underlay. Recall from Figure 2 on page 4 that you'll be creating two routing zones (or VRFs), DC1-Green and DC1-Red. DC1-Green contains routes for subnets 192.168.100.0/24 and 192.168.101.0/24. DC1-Red contains routes for subnet 192.168.200.0/24. In Apstra, each subnet is represented by a virtual network, which you'll configure as DC1-Green-VN1, DC1-Green-VN2, and DC1-Red-VN1.

Figure 14 on page 30 shows these subnets in the context of the physical leaf devices and servers.

Figure 14: Virtual Networks and Subnets



As in the design stage, Apstra automatically names devices in a particular format. This format allows you to determine the rack and the device types and instances solely from the name, but might be difficult to parse. Use [Table 12 on page 30](#) to correlate between the devices in the figure above with the device names that Apstra uses.

Table 12: Device Names

Device	Apstra Device Name
Leaf1	l2_1l2s_001_leaf1
Leaf2	l2_1l2s_002_leaf1
BMS1	l2_1l2s_001_sys001
BMS2	l2_1l2s_001_sys002
BMS3	l2_1l2s_002_sys001
BMS4	l2_1l2s_002_sys002

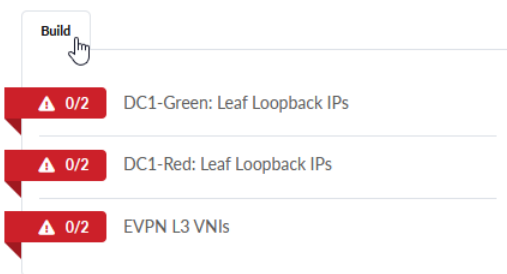
**NOTE:** You can't change the device names in Apstra, but you can change the hostname that Apstra configures on the device. Changing the hostname on the device is easy to do in Apstra but is outside the scope of this document.

1. Click the **Virtual** tab. A new row of tabs appears immediately below: **Virtual Networks**, **Routing Zones**, **Floating IPs**, **Static Routes**, **Protocol Sessions**, **Remote EVPN Gateways**, **Virtual Infra**, and **Endpoints**.
2. Create the DC1-Green routing zone (VRF).
  - a. Select the **Routing Zones** tab.
  - b. Click **Create Routing Zone**.

The Create Routing Zone window appears.

- c. Enter a **VRF Name** (for example, **DC1-Green**) and click **Create**.
3. Create the DC1-Red routing zone (VRF).
  - a. Click **Create Routing Zone**.  
The Create Routing Zone window appears.
  - b. Enter a **VRF Name** (for example, **DC1-Red**) and click **Create**.
4. Select the resource pools for your routing zones.
  - a. In the right-most pane ( [Figure 15 on page 31](#), click the red warning icon to the left of **DC1-Green:Leaf Loopback IPs** to expand the selection and click the **Update assignments** icon in the expanded section.

**Figure 15: Virtual Routing Zones - Build**



- b. Select the **DC1-Green-Loopback IP** pool and click the **Save** icon.
- c. Click the red warning icon to the left of **DC1-Red:Leaf Loopback IPs** to expand the selection and click the **Update assignments** icon in the expanded section.
- d. Select the **DC1-Red-Loopback IP** pool and click the **Save** icon.
- e. Click the red warning icon to the left of **EVPN L3 VNIs** to expand the selection and click the **Update assignments** icon in the expanded section.
- f. Select **DC1-VNI** and click the **Save** icon.

As you progress through this step, the red warning icons turn green to indicate that you've successfully assigned the respective resources.

5. Create the first DC1-Green virtual network.
  - a. Click the **Virtual Networks** tab.
  - b. Select **Create Virtual Networks**.  
The Create Virtual Network window appears.
  - c. Complete the required fields ([Table 13 on page 32](#)) and click **Create**.

Table 13: Create Virtual Network - DC1-Green-VN1

Parameter	Description	Setting in this Use Case
Type	VLAN for single-rack scope.  VXLAN for fabric-wide scope.	VXLAN
Name	The name you want to call this virtual network.	DC1-Green-VN1
Routing Zone	The routing zone that this virtual network belongs to.	DC1-Green
Set same VLAN ID on all leafs?	This setting determines if VLAN IDs have local or fabric-wide significance.	Unchecked
IPv4 Connectivity	Specify whether to enable IPv4 capability on the switch interface. This setting creates an IRB interface on the switch and assigns the <b>Virtual Gateway IPv4</b> address to the IRB interface.	Enabled
IPv4 Subnet	The subnet for this virtual network.	192.168.100.0/24
Virtual Gateway IPv4	The IP address of the gateway for this virtual network. Apstra assigns this IP address to the IRB interface on the switch.	192.168.100.1
Create Connectivity Templates for	Indicate whether the virtual network uses VLAN tagging.  If you want the server to connect to the virtual network using VLAN tagging, select Tagged. If you want the server to connect without VLAN tagging, select Untagged. If you want some servers to connect using VLAN tagging and others to connect without VLAN tagging, then select both Tagged and Untagged.  If you don't select a template option when defining a virtual network, you must create the connectivity templates manually.	Untagged
Assigned To	Select the leaf switches that are part of this virtual network.  From <a href="#">Figure 14 on page 30</a> , both Leaf1 and Leaf2 are part of the DC1-Green-VN1 virtual network.	I2_1I2s_001_leaf1  I2_1I2s_002_leaf1

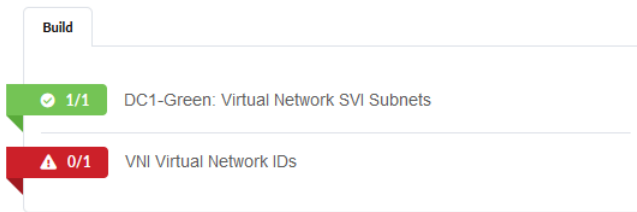
6. Create the second DC1-Green virtual network.
  - a. Select **Create Virtual Networks**.  
The Create Virtual Network window appears.
  - b. Complete the required fields ([Table 14 on page 33](#)) and click **Create**.

Table 14: Create Virtual Network - DC1-Green-VN2

Parameter	Description	Setting in this Use Case
Type	VLAN for single-rack scope.  VXLAN for fabric-wide scope.	VXLAN
Name	The name you want to call this virtual network.	DC1-Green-VN2
Routing Zone	The routing zone that this virtual network belongs to.	DC1-Green
Set same VLAN ID on all leafs?	This setting determines if VLAN IDs have local or fabric-wide significance.	Unchecked/Default
IPv4 Connectivity	Specify whether to enable IPv4 capability on the switch interface. This setting creates an IRB interface on the switch and assigns the <b>Virtual Gateway IPv4</b> address to the interface.	Enabled
IPv4 Subnet	The subnet for this virtual network.	192.168.101.0/24
Virtual Gateway IPv4	The IP address of the gateway for this virtual network. Apstra assigns this IP address to the IRB interface on the switch.	192.168.101.1
Create Connectivity Templates for	Indicate whether the virtual network uses VLAN tagging.  If you want the server to connect to the virtual network using VLAN tagging, select Tagged. If you want the server to connect without VLAN tagging, select Untagged.  If you don't select a template option when defining the virtual network you must create the connectivity templates manually.	Untagged
Assigned To	Select the leaf switches that are part of this virtual network.  From <a href="#">Figure 14 on page 30</a> , only Leaf1 is part of the DC1-Green-VN2 virtual network.	l2_1l2s_001_leaf1

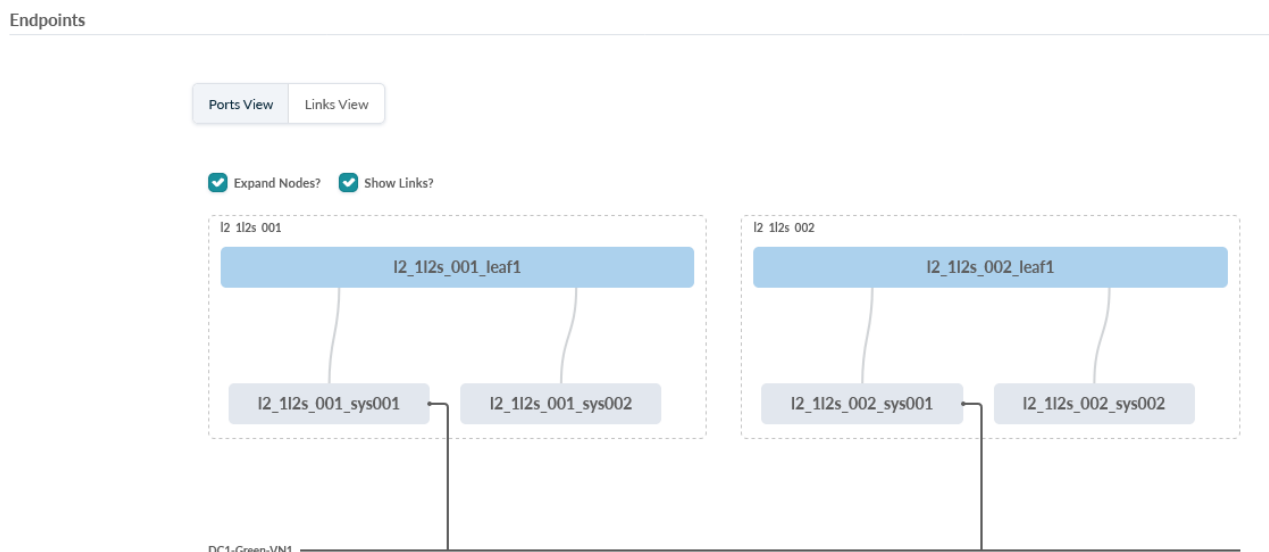
7. Select the resource pools to use for your virtual networks.
  - a. In the right-most pane ([Figure 16 on page 34](#)), click the red warning icon to the left of **VNI Virtual Network IDs** to expand the selection and click the **Update assignments** icon in the expanded section.

Figure 16: Virtual Networks - Build



- b. Select **DC1-VNI** and click the **Save** icon.
8. Configure the server-facing ports on the leaf device for the first DC1-Green virtual network.
  - a. Click the **Connectivity Templates** tab to open the Connectivity Templates page.
  - b. Click the **Assign** icon (it looks like a chain) for the DC1-Green-VN1 network. The Assign Untagged VxLAN 'DC1-Green-VN1' window opens.
  - c. Select the leaf devices and ports associated with this virtual network. In this example the first DC1-Green virtual network is associated with the xe-0/0/0 interfaces on both leaf devices. Select the xe-0/0/0 interface on both leaf devices and click **Assign**. The Status column turns green and indicates the first DC1-Green virtual network has two endpoints.
9. Configure the server-facing ports on the leaf device for the second DC1-Green virtual network.
  - a. While still on the **Connectivity Templates** tab, click the **Assign** icon for the DC1-Green-VN2 network. The Assign Untagged VxLAN 'DC2-Green-VN2' window opens.
  - b. Select the leaf devices and ports associated with this virtual network. In this example the second DC1-Green virtual network is associated with the xe-0/0/1 interface on the Leaf1 (l2\_1l2s\_001\_leaf1) device. Select the xe-0/0/1 interface and click **Assign**. The Status column turns green and indicates the second DC1-Green virtual network has one endpoint.
10. Confirm connectivity for the DC1-Green virtual networks.
  - a. Click the **Virtual Networks** tab.
  - b. In the virtual networks table, click **DC1-Green-VN1**.
  - c. Scroll down to Endpoints>Ports View. This section shows a virtual network topology and port pictogram of both leaf devices (Figure 17 on page 35), showing how the DC1-Green-VN1 virtual network is connected. The display confirms that the DC1-Green virtual network is associated with the first server (sys001) on both leaf devices.

Figure 17: DC1-Green-VN1 Preview



- d. Scroll down to the Port Maps: section. The port pictogram ([Figure 18 on page 35](#)) confirms that port 1 is assigned to the DC1-Green-VN1 virtual network on both leaf devices, and that the port is untagged. Recall you defined this virtual network as being untagged in the Connectivity Template in a previous step. Click on port 1 in the port pictogram to confirm this is the xe-0/0/0 interface on both leaf devices.

Figure 18: DC1-Green-VN1 Port Assignments

Port Maps:

Query: All

Ports: ■ Unassigned ■ Untagged ■ VLAN Tagged ■ Partial

1-2 of 2 < > Page Size: 5

**I2\_112s\_001\_leaf1**

1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39	41	43	45	47	49	51
2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38	40	42	44	46	48	50	52

Port #1 Tr. #1 (10 Gbps, default) xe-0/0/0

**I2\_112s\_002\_leaf1**

1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39	41	43	45	47	49	51
2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38	40	42	44	46	48	50	52

Port #1 Tr. #1 (10 Gbps, default) xe-0/0/0

- e. Repeat this step for the DC1-Green-VN2 network. The topology confirms the second DC1- Green virtual network attaches to Leaf1 only. The Port Map confirms this VN is untagged and maps to the xe-0/0/1 interface on Leaf1.

**11. Create the DC1-Red-VN1 virtual network.**

- a. Scroll back up and click the **Virtual Networks** tab.

- b. Select **Create Virtual Networks**.

The Create Virtual Network window appears.

- c. Complete the required fields ([Table 15 on page 36](#)) and click **Create**.

**Table 15: Create Virtual Network - DC1-Red-VN1**

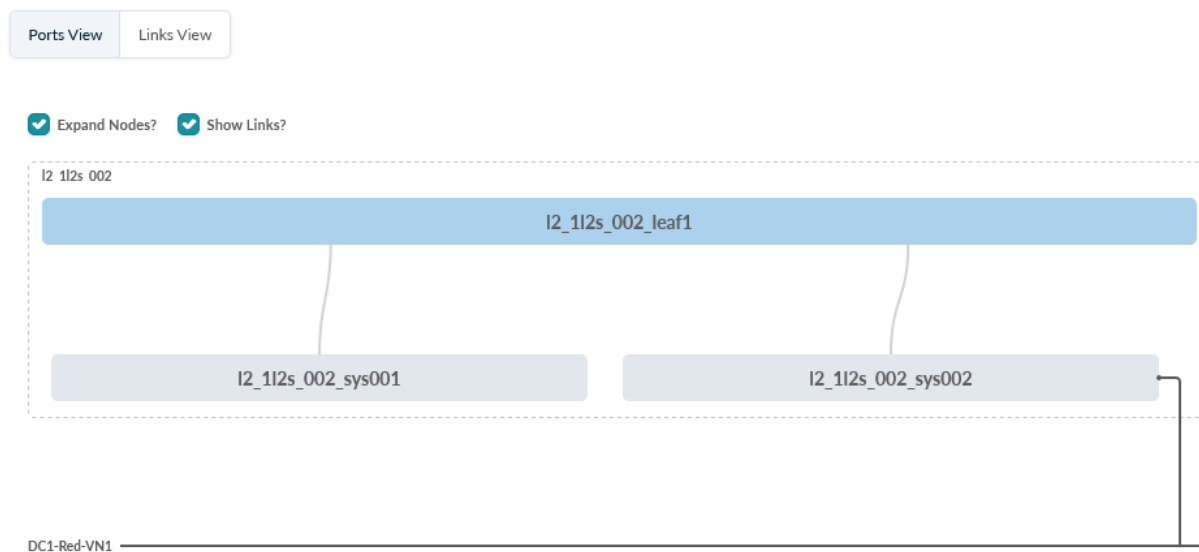
Parameter	Description	Setting in this Use Case
Type	VLAN for single-rack scope.  VXLAN for fabric-wide scope.	VXLAN
Name	The name you want to call this virtual network.	DC1-Red-VN1
Routing Zone	The routing zone that this virtual network belongs to.	DC1-Red
Set same VLAN ID on all leafs?	This setting determines if VLAN IDs have local or fabric-wide significance.	Unchecked
IPv4 Connectivity	Specify whether to enable IPv4 capability on the switch interface. This setting creates an IRB interface on the switch and assigns the <b>Virtual Gateway IPv4</b> address to the interface.	Enabled
IPv4 Subnet	The subnet for this virtual network.	192.168.200.0/24
Virtual Gateway IPv4	The IP address of the gateway for this virtual network. Apstra assigns this IP address to the IRB interface on the switch.	192.168.200.1
Create Connectivity Templates for	Indicate whether the virtual network uses VLAN tagging.  If you want the server to connect to the virtual network using VLAN tagging, select Tagged. If you want the server to connect without VLAN tagging, select Untagged.  If you don't select a template option when defining the virtual network you must create the connectivity templates manually.	Untagged

Table 15: Create Virtual Network - DC1-Red-VN1 (*Continued*)

Parameter	Description	Setting in this Use Case
Assigned To	<p>Select the leaf switches that are part of this virtual network.</p> <p>From <a href="#">Figure 14 on page 30</a>, only Leaf2 is part of the DC1-Red-VN1 virtual network.</p>	I2_1I2s_002_leaf1

12. Configure the server-facing ports on the leaf device for the DC1-Red virtual network.
  - a. Click **Connectivity Templates** tab to open the Connectivity Templates page.
  - b. Click the **Assign** icon (it looks like a chain) for the DC1-Red-VN1 network. The Assign Untagged VxLAN 'DC1-Red' page opens.
  - c. Select the leaf devices and ports associated with this virtual network. In this example the DC1-Red virtual network is associated with the xe-0/0/1 interface on the Leaf2 (I2\_1I2s\_002\_leaf1) device. Select the xe-0/0/1 interface on Leaf2 and click **Assign**. The Status column turns green and indicates the DC1-Red virtual network has a single endpoint.
13. Confirm connectivity for the DC1-Red-VN1 virtual network.
  - a. Click the **Virtual Networks** tab.
  - b. In the virtual networks table, click **DC1-Red-VN1**.
  - c. Scroll down to Endpoints>Ports View. This section shows a virtual network topology and port pictogram of both leaf devices ([Figure 19 on page 37](#)) showing how the DC1-Red-VN1 virtual network is connected. The display confirms that the DC1-Red-VN1 virtual network is associated with the second server (sys002) on the on the Leaf2 (I2\_1I2s\_002\_leaf1) device.

Figure 19: DC1-Red-VN1 Preview



- d. Scroll down to the Port Maps: section. The pictogram ([Figure 20 on page 38](#)) confirms that port 2 is assigned to the DC1-Red-VN1 virtual network on Leaf2, and that the port is untagged. Recall you defined this virtual network as being untagged in the Connectivity Template in a previous step. Click on port 2 in the pictogram to confirm this is the xe-0/0/1 interface on Leaf2.

**Figure 20: DC1-Red-VN1 Port Assignment**

Port Maps:

Query: All

Ports: ■ Unassigned ■ Untagged ■ VLAN Tagged ■ Partial

1 of 1 < > Page Size: 5

**I2\_1I2s\_002\_leaf1**

1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39	41	43	45	47	49	51
2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38	40	42	44	46	48	50	52

Port #2 Tr. #1 (10 Gbps, default) xe-0/0/1

**14. Double check your virtual networks.**

- a. Scroll back up and click the **Virtual Networks** tab.
- b. In the virtual networks table, check that the Routing Zone, Assigned to, and IPv4 Subnet settings are as shown in [Figure 21 on page 38](#).

**Figure 21: Virtual Networks**

<input type="checkbox"/>	Name	Routing Zone	Type	VN ID	Assigned to	IPv4 Connectivity	IPv4 Subnet	IPv6 Connectivity	IPv6 Subnet	Actions
0 selected										
<input type="checkbox"/>	DC1-Green-VN1	DC1-Green	VXLAN	5002	2 nodes	Enabled	192.168.100.0/24	Disabled	N/A	
<input type="checkbox"/>	DC1-Green-VN2	DC1-Green	VXLAN	5003	1 nodes	Enabled	192.168.101.0/24	Disabled	N/A	
<input type="checkbox"/>	DC1-Red-VN1	DC1-Red	VXLAN	5004	1 nodes	Enabled	192.168.200.0/24	Disabled	N/A	

- c. Correct any mistakes before going to the Deploy stage.

You've now finished creating and building your site-specific blueprint. You're ready to deploy a new data center network.

# Deploy

## SUMMARY

Cable your data center fabric and deploy your blueprint.

## IN THIS SECTION

- [Explanation of Procedure | 39](#)
- [Connect the Devices | 39](#)
- [Deploy the Blueprint | 41](#)
- [Case Study: Cabling Anomaly | 42](#)
- [View the Overlay Routing on the Switch \(Optional\) | 43](#)
- [Test BMS Connectivity \(Optional\) | 47](#)
- [Case Study: Configuration Anomaly | 49](#)
- [Case Study: Device Software Upgrade | 51](#)

## Explanation of Procedure

In the deploy stage, you install and cable your devices, and then push an EVPN-VXLAN configuration to the fabric devices.

If you haven't pre-wired your fabric, then you can generate a cabling map for your installers to follow.

If you have pre-wired your fabric, Apstra discovers the existing connectivity and updates the reference design. Recall that Apstra assigns port connections arbitrarily, so there's a chance these arbitrary port assignments differ from the pre-existing wiring. By importing the actual connections back to your blueprint, you're ensuring that Apstra has an accurate view of the connectivity.

Regardless of which approach you take, when you deploy the configuration, Apstra checks the actual cabling and the behavior of the network with the reference design and flags any anomalies.

## Connect the Devices

1. Click the **Staged** tab in your blueprint and select the **Physical** tab.
2. Select the **Links** tab.
3. If you haven't pre-wired your fabric, then export a cabling map and connect your devices according to the map.
  - a. Click the **Export cabling map** icon ([Figure 22 on page 40](#)). You can save the cabling map as a JSON or CSV file.

Figure 22: Export Cabling Map

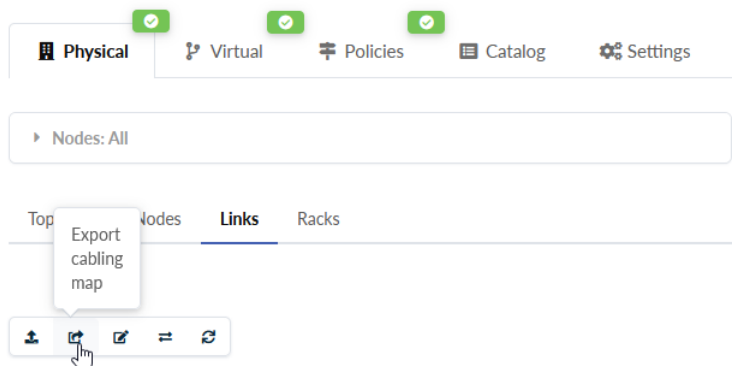


Figure 23 on page 40 shows an example of the cabling map in JSON format.

Figure 23: Cabling Map

### Export Cabling Map

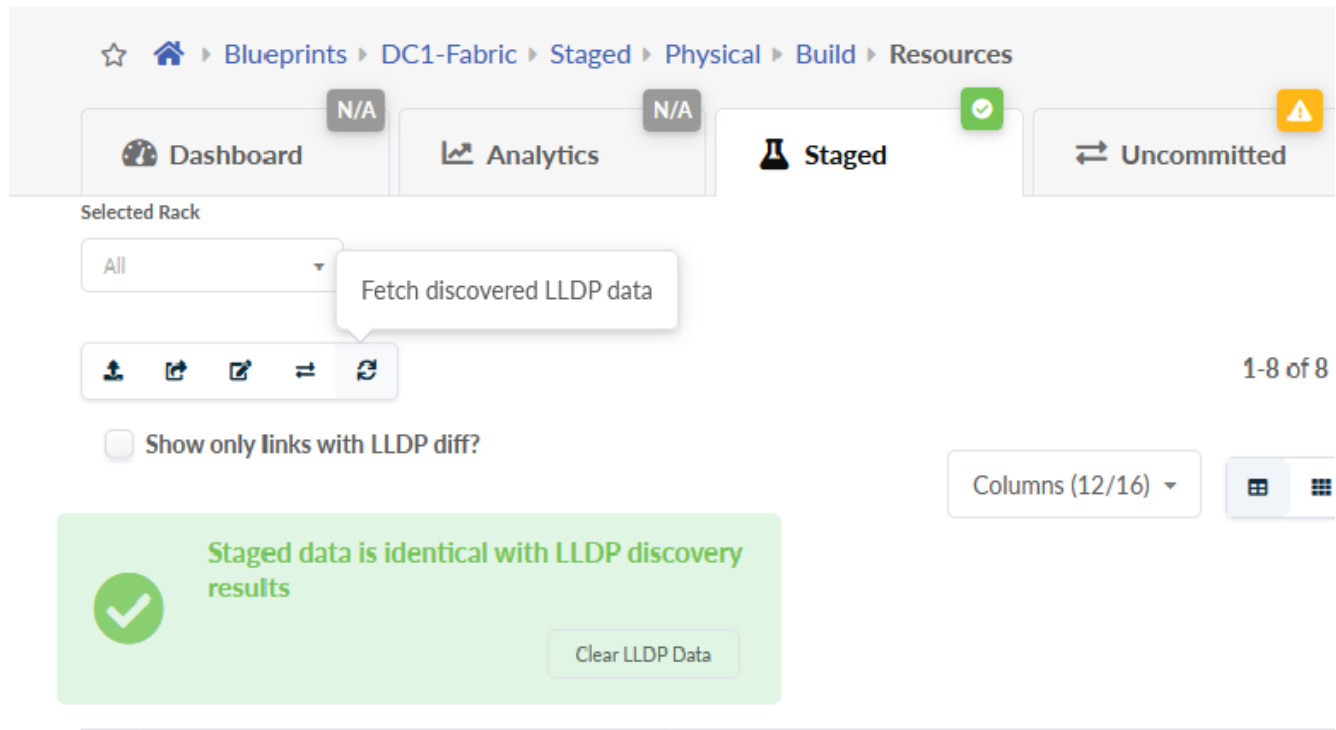
Mode  
☒ JSON ☐ CSV

```
{
  "version": 5,
  "links": [
    {
      "type": "ethernet",
      "aggregate_link_id": null,
      "group_label": "1L2S-Leaf-to-Server",
      "label": "12_1l2s_001_leaf1<->12_1l2s_001_server002 (1L2S-Leaf-to-Server) [1]",
      "role": "leaf_l2_server",
      "endpoints": [
        {
          "interface": {
            "ipv6_addr": null,
            "if_name": "xe-0/0/1",
            "port_channel_id": null,
            "ipv4_addr": null,
            "if_type": "ethernet",
            "id": "d04a83fb-36dd-4a74-87c8-43ec4c36f4bf",
            "lag_mode": null
          },
          "system": {
```

Copy Save As File

- b. Physically connect your devices based on the information in the cabling map.
4. If you have pre-wired your fabric and you want that wiring to remain in effect, import the links back to your blueprint.
  - a. Click the **Fetch discovered LLDP data** icon (Figure 24 on page 41).

Figure 24: Fetch Discovered LLDP Data



- b. If the staged data is different from the LLDP discovery results, then click **Update Cabling Map from LLDP** to open the Cabling Map Editor where you can update the blueprint to match the connectivity discovered by LLDP. In the example shown above, the discovered LLDP data matches the connectivity in the existing cabling map.

**NOTE:** Updating the cabling map from LLDP data is only successful if the existing wiring does not violate the interface map specifications in the blueprint. In this use case, ports et-0/0/48 and et-0/0/50 on the leaf devices must connect to the spine devices. Likewise, ports et-0/0/34 and et-0/0/35 on the spine devices must connect to the leaf devices.

## Deploy the Blueprint

You're now ready to deploy your design! Apstra pushes the configuration to all your fabric devices. Apstra then checks to make sure the fabric is behaving in accordance with the reference design.

1. Select the **Uncommitted** tab to open the Uncommitted page. This page shows a summary of the differences between the staged configuration and the actual configuration on the devices. Scroll through this page to see the details of the configuration changes needed on the fabric devices to instantiate your blueprint.
2. Click **Commit** to deploy the blueprint. The Commit changes from Staged to Active? confirmation window appears. If desired, enter a comment and click the **Commit** button to activate the changes.

3. Select the **Dashboard** tab to watch as the configuration is downloaded and committed. The anomalies eventually clear as the underlay and overlay protocol sessions establish.

This step can take several minutes in this use case. If one or more anomalies fail to clear, click the related anomaly gauge to see details on the anomaly.

## Case Study: Cabling Anomaly

One of the features of Apstra is its ability to detect mismatches between your blueprint and the actual physical connectivity of the fabric devices. [Figure 25 on page 42](#) shows an example of BGP, cabling, and routing anomalies. It's always good practice to address any physical layer anomalies first, which is what you'll do in this troubleshooting example.

Figure 25: Deployment Anomalies

### Anomalies



1. Click the **Cabling** anomalies gauge.

This brings up the Anomalies page filtered for the anomaly type of cabling ([Figure 26 on page 43](#)).

Figure 26: Cabling Anomalies

Node	Hostname	Service	Anomaly Type	Role	Anomaly Extra Details	Expected	Actual	Time Updated
spine2	spine2	IP Fabric	cabling	Spine to Leaf	Property: Value Interface: "et-0/0/35"	Property: Value neighbor interface: "et-0/0/50" neighbor name: "12-112a-002-leaf1"	Property: Value neighbor interface: "522" neighbor name: "e1:5d:37:e9:d5:02"	a few seconds ago
spine1	spine1	IP Fabric	cabling	Spine to Leaf	Property: Value Interface: "et-0/0/35"	Property: Value neighbor interface: "et-0/0/48" neighbor name: "12-112a-002-leaf1"	Property: Value neighbor interface: "521" neighbor name: "e1:5d:37:e9:d5:02"	a few seconds ago
Q_12a_002-leaf1	Q-12a-002-leaf1	IP Fabric	cabling	Spine to Leaf	Property: Value Interface: "et-0/0/50"	Property: Value neighbor interface: "et-0/0/35" neighbor name: "spine2"	Property: Value neighbor interface: "" neighbor name: ""	a few seconds ago
Q_12a_002-leaf1	Q-12a-002-leaf1	IP Fabric	cabling	Spine to Leaf	Property: Value Interface: "et-0/0/48"	Property: Value neighbor interface: "et-0/0/35" neighbor name: "spine1"	Property: Value neighbor interface: "" neighbor name: ""	a few seconds ago

## 2. Interpret the findings on this page.

- The first row indicates that the Spine2 device is expecting to connect to the et-0/0/50 interface on the Leaf2 device in your design, but instead connects to a device with the displayed MAC address.
- The second row indicates that the Spine1 device is expecting to connect to the et-0/0/48 interface on the Leaf2 device in your design, but instead connects to a device with the displayed MAC address.
- The third row indicates that the Leaf2 device is expecting to connect to the et-0/0/35 interface on the spine2 device in your design, but instead is unconnected.
- The fourth row indicates that the Leaf2 device is expecting to connect to the et-0/0/35 interface on the spine1 device in your design, but instead is unconnected.

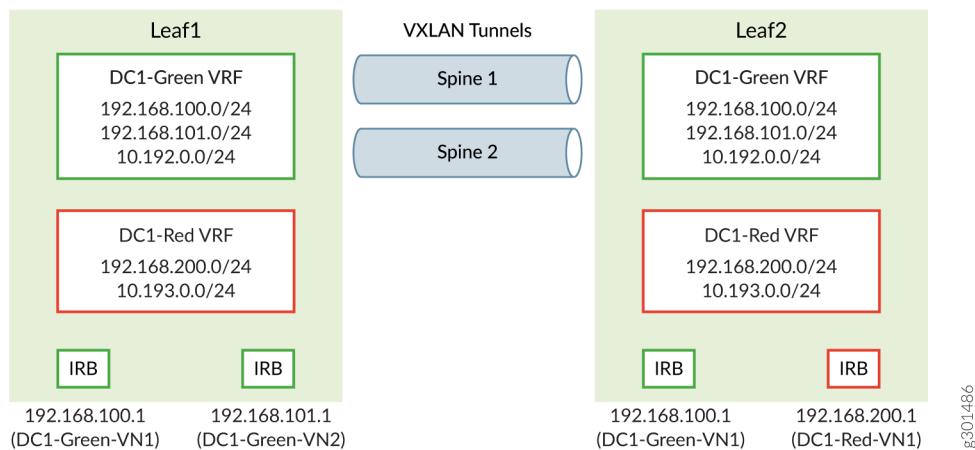
These anomalies all indicate that the problem is with the connections between the spine devices and the Leaf2 device.

Once you fix the cabling, the cabling anomalies will disappear. If the cabling fault is the root cause of the BGP and the route anomalies, then these anomalies also clear when BGP sessions establish and the routing tables converge.

## View the Overlay Routing on the Switch (Optional)

You might be curious on how Apstra configures the switches for the overlay networks. [Figure 27 on page 44](#) shows a simplified view of the routing tables in the leaf switches. You can see more details as you follow this procedure.

Figure 27: Overlay Routing (Simplified)



Both leaf switches contain VRFs for the DC1-Green and DC1-Red overlay networks. Each VRF contains routes to the BMS endpoints, loopback addresses, and local IRB interfaces that are part of that VRF's networks. Leaf1 has IRB interfaces for the DC1-Green-VN1 and DC1-Green-VN2 virtual networks while Leaf2 has IRB interfaces for the DC1-Green-VN1 and DC1-Red-VN1 virtual networks.

These correspond to how you attached the virtual networks to each leaf switch earlier. Refer to (Figure 14 on page 30, Figure 17 on page 35, and Figure 19 on page 37 ) for details on the virtual networks used in this example.

The spine switches do not perform any overlay routing in this Edge-routed bridging (ERB) architecture. The spine switches merely provide underlay routing to support the VXLAN tunnels between the leaf switches.

1. See how Apstra has configured the overlay IP addresses and routing tables on the Leaf1 switch.
  - a. Log in to the Leaf1 switch.
  - b. Show the IRB interface IP addresses.

```
user@l2-112s-001-leaf1> show interfaces terse | match irb
irb                up    up
irb.4              up    up   inet    192.168.100.1/24
irb.5              up    up   inet    192.168.101.1/24
```

- c. Show the loopback addresses.

```
user@l2-112s-001-leaf1> show interfaces terse | match lo0
lo0                up    up
lo0.0              up    up   inet    10.255.0.2        --> 0/0
lo0.2              up    up   inet    10.192.0.0        --> 0/0
```

```

lo0.3          up    up    inet    10.193.0.0    --> 0/0
lo0.16385      up    up    inet

```

d. Show the routing table for the DC1-Green VRF.

```

user@l2-112s-001-leaf1> show route table DC1-Green.inet.0

DC1-Green.inet.0: 6 destinations, 7 routes (6 active, 0 holddown, 0 hidden)
@ = Routing Use Only, # = Forwarding Use Only
+ = Active Route, - = Last Active, * = Both

10.192.0.0/32    *[Direct/0] 00:39:46
                 > via lo0.2
10.192.0.1/32    *[EVPN/170] 00:38:15
                 to 10.10.0.0 via et-0/0/48.0
                 > to 10.10.0.4 via et-0/0/50.0
192.168.100.0/24 *[Direct/0] 00:38:22
                 > via irb.4
                 [EVPN/170] 00:38:15
                 to 10.10.0.0 via et-0/0/48.0
                 > to 10.10.0.4 via et-0/0/50.0
192.168.100.1/32 *[Local/0] 00:38:22
                 Local via irb.4
192.168.101.0/24 *[Direct/0] 00:38:22
                 > via irb.5
192.168.101.1/32 *[Local/0] 00:38:22
                 Local via irb.5

```

e. Show the routing table for the DC1-Red VRF.

```

user@l2-112s-001-leaf1> show route table DC1-Red.inet.0

DC1-Red.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
@ = Routing Use Only, # = Forwarding Use Only
+ = Active Route, - = Last Active, * = Both

10.193.0.0/32    *[Direct/0] 00:41:55
                 > via lo0.3
10.193.0.1/32    *[EVPN/170] 00:40:24
                 to 10.10.0.0 via et-0/0/48.0
                 > to 10.10.0.4 via et-0/0/50.0
192.168.200.0/24 *[EVPN/170] 00:40:24
                 > to 10.10.0.0 via et-0/0/48.0
                 to 10.10.0.4 via et-0/0/50.0

```

## 2. See how Apstra has configured the overlay IP addresses and routing tables on the Leaf2 switch.

- a. Log in to the Leaf2 switch.
- b. Show the IRB interface IP addresses.

```
user@l2-112s-002-leaf1> show interfaces terse | match irb
irb                up    up
irb.4              up    up   inet    192.168.100.1/24
irb.5              up    up   inet    192.168.200.1/24
```

- c. Show the loopback addresses.

```
user@l2-112s-002-leaf1> show interfaces terse | match lo0
lo0                up    up
lo0.0              up    up   inet    10.255.0.3        --> 0/0
lo0.2              up    up   inet    10.192.0.1        --> 0/0
lo0.3              up    up   inet    10.193.0.1        --> 0/0
lo0.16385          up    up   inet
```

- d. Show the routing table for the DC1-Green VRF.

```
user@l2-112s-002-leaf1> show route table DC1-Green.inet.0

DC1-Green.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
@ = Routing Use Only, # = Forwarding Use Only
+ = Active Route, - = Last Active, * = Both

10.192.0.0/32      *[EVPN/170] 00:44:58
                   to 10.10.0.2 via et-0/0/48.0
                   > to 10.10.0.6 via et-0/0/50.0
10.192.0.1/32      *[Direct/0] 00:46:29
                   > via lo0.2
192.168.100.0/24   *[Direct/0] 00:45:04
                   > via irb.4
                   [EVPN/170] 00:44:58
                   to 10.10.0.2 via et-0/0/48.0
                   > to 10.10.0.6 via et-0/0/50.0
192.168.100.1/32   *[Local/0] 00:45:04
                   Local via irb.4
192.168.101.0/24   *[EVPN/170] 00:44:58
                   to 10.10.0.2 via et-0/0/48.0
                   > to 10.10.0.6 via et-0/0/50.0
```

e. Show the routing table for the DC1-Red VRF.

```
user@l2-112s-002-leaf1> show route table DC1-Red.inet.0

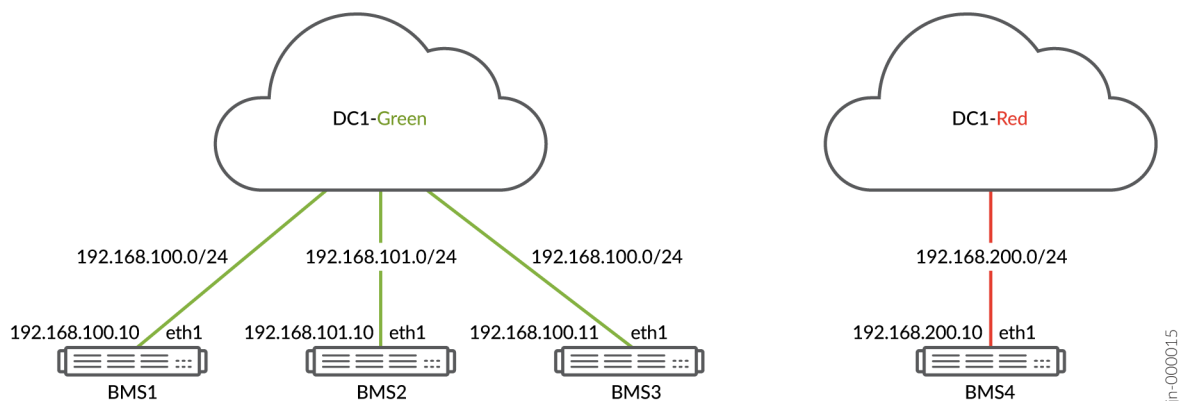
DC1-Red.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
@ = Routing Use Only, # = Forwarding Use Only
+ = Active Route, - = Last Active, * = Both

10.193.0.0/32      *[EVPN/170] 00:46:10
                   to 10.10.0.2 via et-0/0/48.0
                   > to 10.10.0.6 via et-0/0/50.0
10.193.0.1/32      *[Direct/0] 00:47:41
                   > via lo0.3
192.168.200.0/24   *[Direct/0] 00:47:41
                   > via irb.5
192.168.200.1/32  *[Local/0] 00:47:41
                   Local via irb.5
```

## Test BMS Connectivity (Optional)

In this section you test the connectivity between the servers in the DC1-Green virtual networks. [Figure 28 on page 47](#) summarizes the DC1-Green and DC1-Red virtual networks including the IP address assignments for the server endpoints.

Figure 28: DC1-Green and DC1-Red Overlay Virtual Networks



Refer to [Figure 14 on page 30](#) for details on how the four BMS devices attach to the leaf devices in the underlay. In summary, Leaf1 supports both the DC1-Green-VN1 and the DC1-Green-VN2 virtual networks, while Leaf2 supports the DC1-Green-VN1 and the DC1-Red-VN1 virtual networks.

Display the IP addressing and routing configured on BMS1. You configure the BMS network settings outside of Apstra using standard CentOS commands. Using CentOS to configure network parameters is outside the scope of this document. The information provided in this section is for completeness.

- a. Log in to the BMS1 server.
- b. Confirm the IP addresses assigned to the eth1 interface. Also confirm the static route needed to reach the DC1-Green-VN2 virtual network.

```
[user@bms1 ~]$ ip a show dev eth1
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether 00:50:56:a2:c0:a1 brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.10/24 brd 192.168.100.255 scope global eth1
        valid_lft forever preferred_lft forever
    inet6 fe80::9662:5fb2:617e:fd1d/64 scope link
        valid_lft forever preferred_lft forever

[user@bms1 ~]$ ip route get 192.168.101.0/24
192.168.101.0 via 192.168.100.1 dev eth1 src 192.168.100.10
```

- c. Verify that BMS1 can ping its default gateway. This address belongs to the IRB interface associated with the DC1-Green-VN1 VRF.

```
[user@bms1 ~]$ ping 192.168.100.1 -c 2
PING 192.168.100.1 (192.168.100.1) 56(84) bytes of data.
64 bytes from 192.168.100.1: icmp_seq=1 ttl=64 time=0.767 ms
64 bytes from 192.168.100.1: icmp_seq=2 ttl=64 time=0.742 ms

--- 192.168.100.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.742/0.754/0.767/0.030 ms
```

- d. Verify that BMS1 can ping BMS3. Recall that these two servers attach to different leaf devices, but both servers reside in the DC1-Green-VN1 virtual network. As a result, these machines require only Layer 2 connectivity to communicate. You confirm Layer 2 connectivity by tracing the path between the servers. The result shows a single-hop that is the destination endpoint.

```
[user@bms1 ~]$ ping 192.168.100.11 -c 2
PING 192.168.100.11 (192.168.100.11) 56(84) bytes of data.
64 bytes from 192.168.100.11: icmp_seq=1 ttl=64 time=0.569 ms
64 bytes from 192.168.100.11: icmp_seq=2 ttl=64 time=0.310 ms

--- 192.168.100.11 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.310/0.439/0.569/0.131 ms
```

```
[user@bms1 ~]$ tracepath -n 192.168.100.11
1?: [LOCALHOST] pmtu 1500
1: 192.168.100.11 0.503ms !H
1: 192.168.100.11 0.228ms !H
Resume: pmtu 1500
```

- e. Verify that BMS1 can ping BMS2. Recall that these two servers attach to the same leaf device, but the servers reside in different DC1-Green virtual networks. As a result, these machines communicate using Layer 3 connectivity. You confirm Layer 3 connectivity by tracing the path between the servers and noting that the IRB interface address, and the resulting extra hop, appear in the path.

```
[user@bms1 ~]$ ping 192.168.101.10 -c 2
PING 192.168.101.10 (192.168.101.10) 56(84) bytes of data.
64 bytes from 192.168.101.10: icmp_seq=1 ttl=63 time=0.573 ms
64 bytes from 192.168.101.10: icmp_seq=2 ttl=63 time=0.440 ms

--- 192.168.101.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.440/0.506/0.573/0.070 ms
```

```
[user@bms1 ~]$ tracepath -n 192.168.101.10
1?: [LOCALHOST] pmtu 1500
1: 192.168.100.1 81.981ms
1: 192.168.100.1 0.778ms
2: 192.168.101.10 0.496ms !H
Resume: pmtu 1500
```

The ping results confirm the expected connectivity for the servers in the DC1-Green virtual networks. You can follow similar steps for BMS4 in the DC1-Red-VN1 virtual network. Because only one BMS is in this virtual network, you must limit ping testing at BMS4 to the IRB gateway address (192.168.200.1) associated with the DC1-Red virtual network.

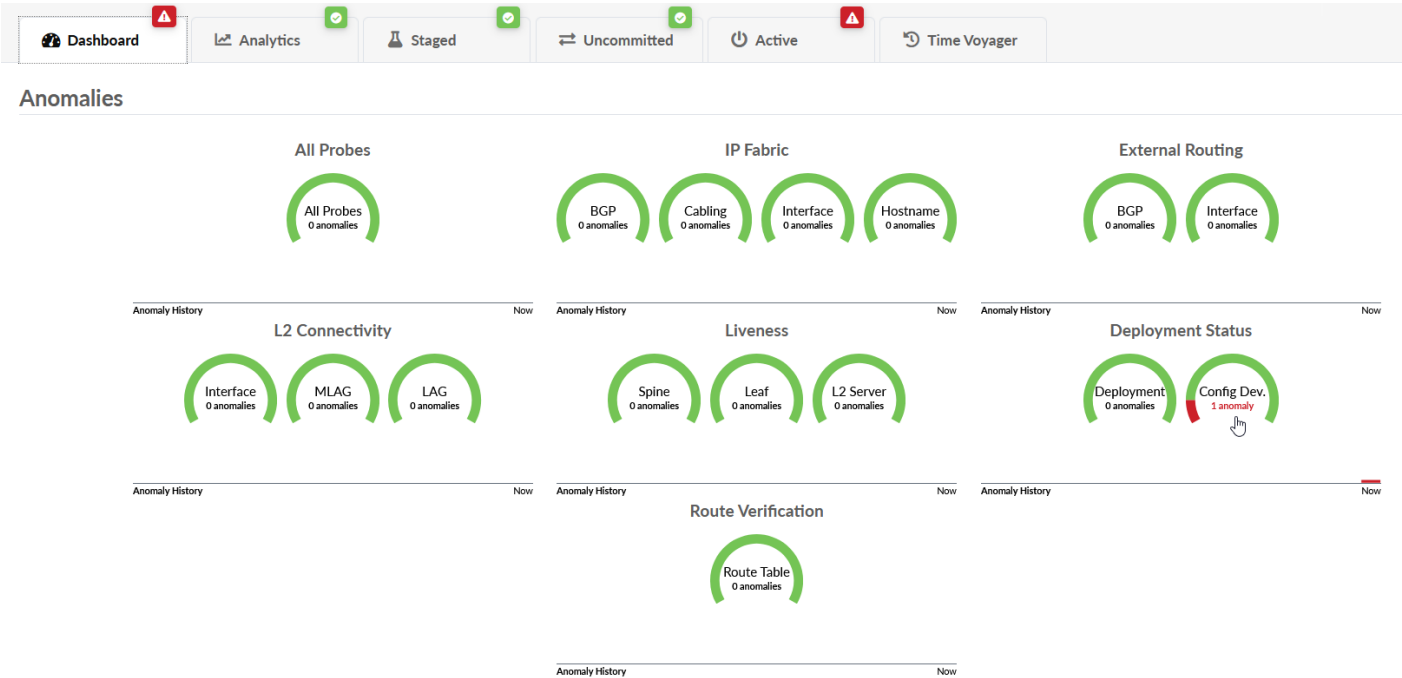
Pings between servers in a green virtual network to a device in the red virtual network are expected to fail. This is because the green and red virtual networks are housed in different VRFs (routing zones).

## Case Study: Configuration Anomaly

If a fabric device reports a configuration change, Apstra compares that change with the reference design and flags any differences as anomalies. In this case study, a rogue operator has logged in to the Leaf2 switch CLI and manually changed the configuration on the device.

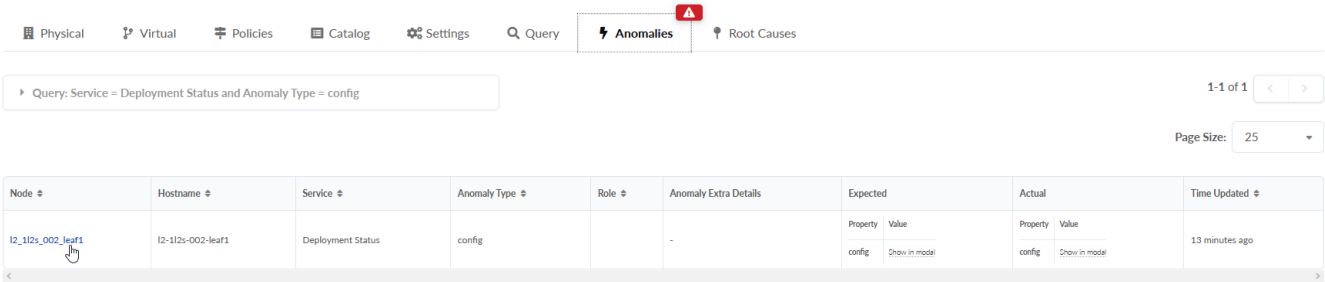
Apstra detects a configuration difference between the reference design and the configuration on the device and flags the anomaly in the dashboard (Figure 29 on page 50).

Figure 29: Dashboard with Configuration Anomaly



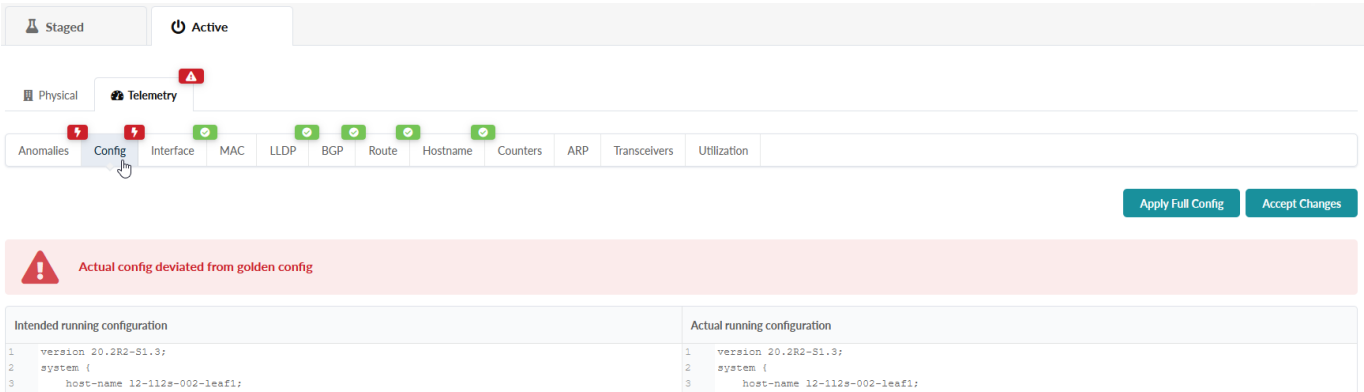
1. Click the **Config Dev** anomaly gauge to bring up the Anomalies page with a filter automatically applied to show only configuration-related anomalies (Figure 30 on page 50).

Figure 30: Anomalies



2. Click the **I2\_1I2s\_002\_leaf1** node in the table to open the Anomalies page for that node.
3. Click the **Config** tab to display the differences between the intended and the actual running configuration (Figure 31 on page 51).

Figure 31: Configuration Differences Between Intended and Actual



4. Scroll through this page to look for highlighted discrepancies (Figure 32 on page 51).

Figure 32: Highlighted Differences



In this example, you see that someone has changed the route target for the DC1-Red VRF on Leaf2 to a value that coincidentally matches the route target for the DC1-Green VRF. This means that DC1-Red routes get imported into DC1-Green routing tables and vice versa. The result is unintended connectivity between the DC1-Green and DC1-Red routing zones!

- Assuming you don't want to this connectivity, click **Apply Full Config** to reapply the intended configuration. The anomalies in the dashboard should disappear when the applied configuration takes effect.

## Case Study: Device Software Upgrade

Few tasks bring more trepidation to the network administrator than upgrading the software on a device. While using Apstra to upgrade device software does not guarantee upgrade success, you can rest assured that Apstra validates device operation after the upgrade to alert you to any anomalies.

This case study shows you how to upgrade software on the Leaf1 switch. Before you can upgrade the device software, you must register (upload) the software image to Apstra. Registering the software in Apstra is easy to do but is outside the scope of this document.

1. Before upgrading, ensure no anomalies are active in the dashboard.
2. In the left-nav bar, select **Devices>System Agents>Agents** to open the Agents page.
3. Click the **OFFBOX** tab to open the OFFBOX Agents page.
4. Find the row for the Leaf1 switch and click the **OS Upgrade** icon at the far right of the row (Figure 33 on page 52).

**Figure 33: Device OS Upgrade**

0 selected	Device Address	Operation Mode	Platform	Platform Version	Job State	Connection State	System ID	Hostname	Device State	Actions
<input type="checkbox"/>	10.123.151.2	FULL CONTROL	JUNOS	20.2R2-S1.3	SUCCESS	CONNECTED	WS377		IS-ACTIVE	OS Upgrade
<input type="checkbox"/>	10.123.162.1	FULL CONTROL	JUNOS	20.2R2-S1.3	SUCCESS	CONNECTED	EP226		IS-ACTIVE	
<input type="checkbox"/>	10.123.151.1	FULL CONTROL	JUNOS	20.2R2-S1.3	SUCCESS	CONNECTED	WS371		IS-ACTIVE	
<input type="checkbox"/>	10.123.162.2	FULL CONTROL	JUNOS	20.2R2-S1.3	SUCCESS	CONNECTED	EP249		IS-ACTIVE	

The Upgrade OS Image window appears.

5. Use the drop-down list to select the desired **OS Image** and click **Upgrade OS Image**. Only registered images for the platform are available for selection.

The upgrade starts and the Job State shows IN PROGRESS.

6. Watch for the Job State and Connection State to both show SUCCESS, and for the Platform Version to reflect the upgraded software release. An upgrade can take over 30 minutes to complete.
7. Go back to the dashboard to confirm anomalies clear as BGP sessions establish and routing table converge.  
Anomalies clear after a few minutes as Apstra validates the network against the reference design. Once the anomalies clear, you'll know the upgrade was successful.

## Summary

### IN THIS SECTION

- What's Next | 53

In this use case, you used the Juniper Apstra to design, build, and deploy a fabric of QFX Series switches, including the overlay networks that run on top of the fabric.

The procedures you used to create the network in this use case apply equally well to cloud-scale data centers. The notion of configuring the fabric as a whole, while leaving individual device configuration details to Apstra, means these procedures scale gracefully. Being able to create multiple blueprints from a single design template enables you to take a cookie cutter approach when deploying multiple sites.

The Apstra design process is highly intuitive because you base your design on physical building blocks such as ports, devices, and racks. By creating these building blocks and specifying what ports are used, you've given Apstra all the information necessary to come up with a reference design for your fabric.

The reference design built by Apstra contains not only configuration information, but also the roles and responsibilities of various components, enforcement mechanisms, and the expectations that need to be met. This allows Apstra to use the reference design not only to configure the fabric but to validate the fabric once you deploy the configuration, and on an ongoing basis thereafter.

We've shown how this works by creating cabling and configuration issues for Apstra to detect as well as performing a device software upgrade. This ability to reliably handle controlled changes and unexpected network events is what separates Juniper Apstra from other fabric management solutions.

## WHAT'S NEXT

To reinforce your learning, try the **Apstra** lab at [Juniper Networks Virtual Labs](#). You can find the **Apstra** lab by scrolling down to the Switching category.

## RELATED DOCUMENTATION

[EVPN-VXLAN Technology and Architectures](#)