JUNIPER NETWORKS | Engineering Simplicity

Secure Data Center Fabric with Juniper SRX Series Firewalls - Juniper Validated Design Extension (JVDE)

Published
2026-03-26

Table of Contents

About this Document | 1

Solution Benefits | 1

Use Case and Reference Architecture | 3

Solution Architecture | 5

Configuration Walkthrough | 9

Validation Framework | 35

Test Objectives | 38

Results Summary and Analysis | 39

Recommendations | 40

Revision History | 41

Secure Data Center Fabric with Juniper SRX Series Firewall - Juniper Validated Design Extension (JVDE)

Juniper Networks Validated Designs provide you with a comprehensive, end-to-end blueprint for deploying Juniper solutions in your network.These designs are created by Juniper's expert engineers and tested to ensure they meet your requirements.Using a validated design, you can reduce the risk of costly mistakes, save time and money, and ensure that your network is optimized for maximum performance.

About this Document

This document is an extension to the published 3-Stage Data Center Design with Juniper Apstra (JVD) which covers the 3-stage clos design and deployment using Apstra. The 3-stage data center design is validated separately. For more information on deployment of 3-stage data center with Apstra, see 3-Stage Data Center Design with Juniper Apstra (JVD) . This JVDE document outlines the design and architecture of Secured Data Center Fabric using Juniper SRX4600 and SRX4700 firewall. This document is intended for an audience familiar with Juniper technologies such as Junos OS, SRX, and Multinode High Availability (MNHA).

Solution Benefits

IN THIS SECTION

- Juniper SRX Series Firewall Integration with EVPN VXLAN Fabric | 2

This document provides comprehensive guidance on the benefits of implementing a secure data center design by integrating security into a modern three-stage fabric using EVPN-VXLAN. For the purposes of this JVDE, the Juniper SRX4600 referenced in this design is a next-generation firewall platform engineered to deliver high-performance security within data center and cloud-enabled environments. The 3-Stage Fabric with security integrated is designed to meet the needs of most of Juniper’s customers. Advanced JVDE testing by Juniper combined with widespread adoption simplifies

troubleshooting and shortens the support cycle, leading to a secure and stable data center fabric, reducing operational costs.

The data center connected Security JVD is an extension to 3-Stage Data Center Design with Juniper Apstra (JVD). Like all Juniper data centre JVDs, it is based on best practices as determined by Juniper's subject matter experts, and Juniper support teams have extensive training and resources necessary to support networks based on JVDs.

Juniper SRX Series Firewall Integration with EVPN VXLAN Fabric

The integration of **Ethernet VPN (EVPN)** with **Virtual Extensible LAN (VXLAN)** on Juniper SRX Series Firewall enables secure, scalable multi-tenant data center interconnect (DCI) and segmentation. By combining the flexibility of VXLAN overlays with EVPN's control plane, organizations can extend Layer 2 and Layer 3 networks across geographically distributed locations while maintaining high levels of visibility and security.

With Juniper SRX Native EVPN VXLAN Type 5 integration, SRX can secure EVPN VXLAN fabrics while significantly reducing operational overhead. SRX can inspect the traffic with advanced security services such as, IDP, App Secure, and Content Security for both North-South and East-West traffic. For high-availability, the SRX4600 device is configured in MNHA cluster. Since SRX decapsulates the VXLAN traffic, all the security features can be applied to the traffic.

- Unified policy
- Application Security:
  - Application Identification
  - Application Based Routing
- Content Security
  - Web Filtering
  - Antivirus
  - Content Filtering
  - Anti-Spam
- IDP
- SSL Inspection
- Advance Threat Prevention
  - Security intelligence

    - Advance Anti-malware
    - Adaptive Threat Profiling
- DNS Security
- Encrypted Traffic Insights
- User/Device Authentication

The SRX4600 can inspect North-South and east-west traffic as described in the "Use Case and Reference Architecture" on page 3.

Use Case and Reference Architecture

There are two main use cases that will be addressed as part of this JVDE.

1. **Inter-VRF Traffic** : In an EVPN/VXLAN data center fabric, the SRX4600 firewall is configured with all the Virtual Routing and Forwarding (VRF) instances, configuration covered in section "SRX Configuration" on page 20. Inter-VRF traffic is forwarded to the SRX firewall by leaf (through spine), the SRX Series Firewall applies appropriate security policies on the inner VXLAN packet header. Based on the security policies, SRX then routes the traffic toward its intended destination while preserving VXLAN encapsulation across the fabric—without terminating the VXLAN tunnel on the firewall. This design eliminates the need for ACL configurations on the fabric switches.
2. **North-South:** North-South traffic (fabric- to- Internet) is sent through the SRX firewall, which applies both basic and advanced security policies. Outbound traffic is routed through -the SRX using the default route injected by the underlying fabric architecture from the border leaf.

Figure 1 on page 4 shows the EVPN-VXLAN architecture with SRX devices, and Figure 2 on page 5 shows the typical traffic flow through this architecture.

Figure 1: Architecture: With SRX Peering with the Spine

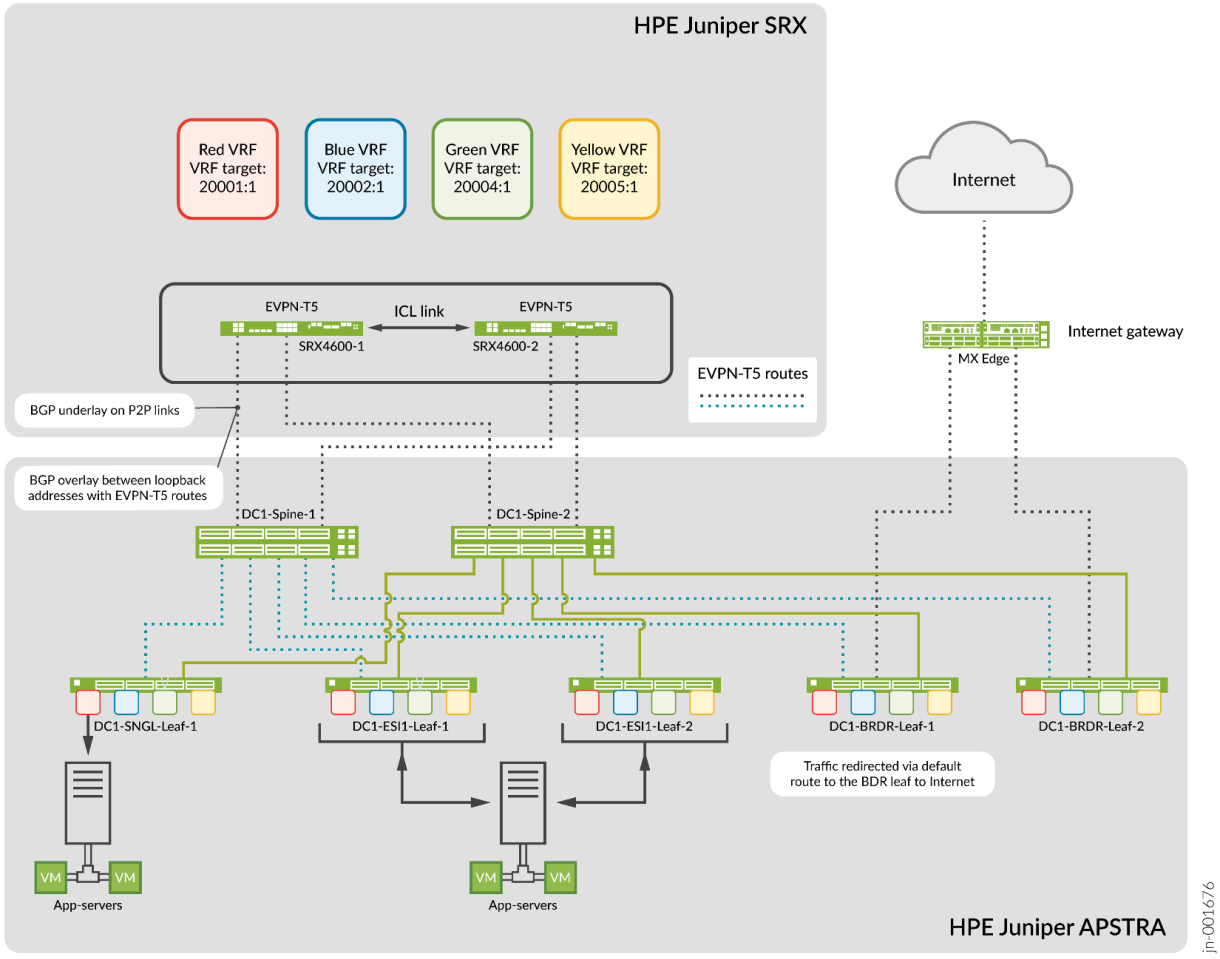
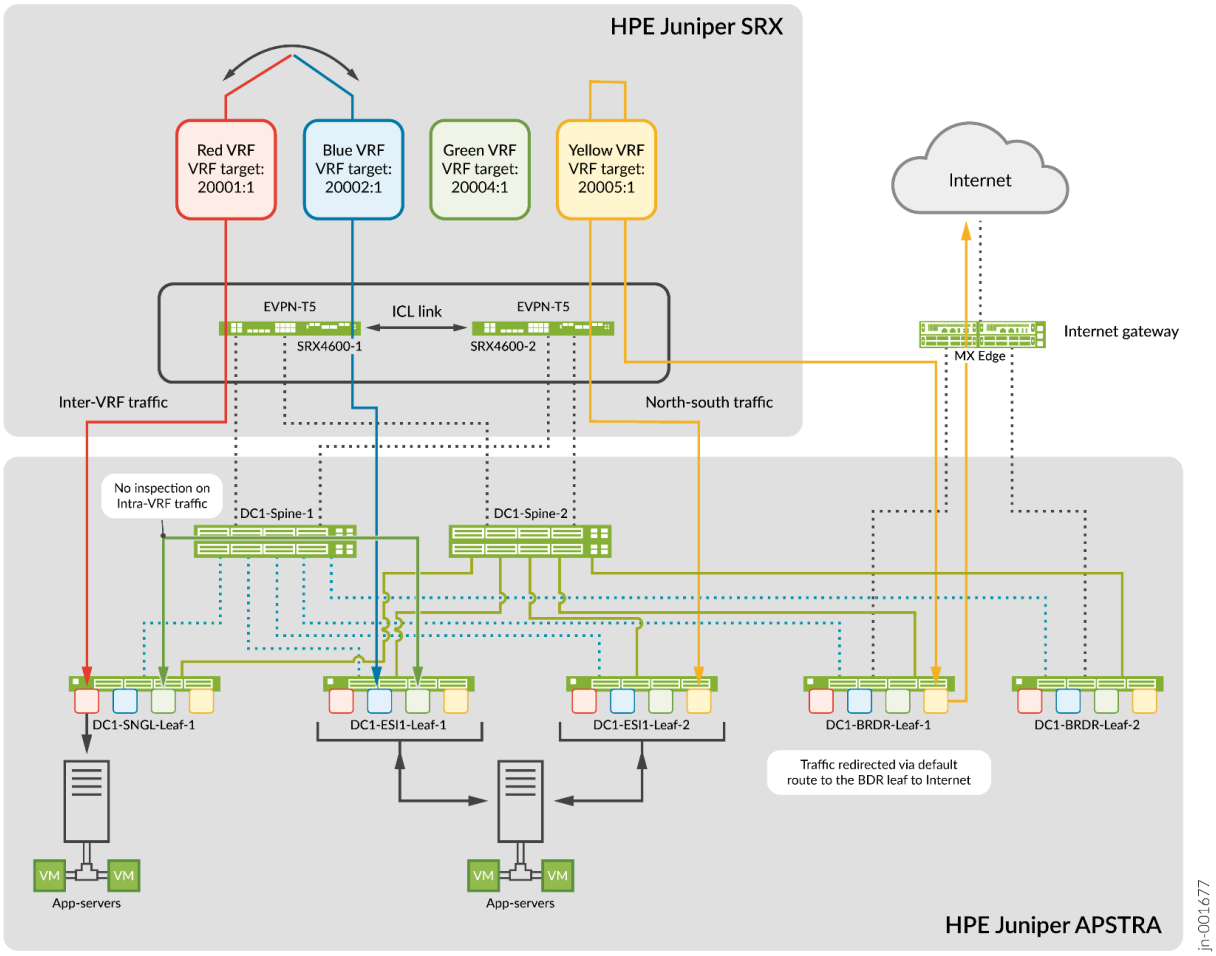


Figure 2: Traffic Flow: With SRX Peering with the Spine



Solution Architecture

IN THIS SECTION

- Juniper Hardware and Software Components | 7
- Validated Functionality | 8

JVDE document is built upon the 3-Stage Data Center Design with Juniper Apstra (JVD). This JVDE covers the basic architecture and components along with deployment of the 3-Stage data center fabric.

The 3-stage data center design is based on the ERB architecture where the Spine switches are lean spine, and the server leaf switches are the VXLAN tunnel endpoint (VTEPs) connected to the hosts.

In this JVDE, the reference architecture includes two SRX4600 devices connected to the Spine switches in a (MNHA cluster. For more information refer the Multi-Node High availability services guide. The two SRX4600 firewall devices are connected to Spine switches using the revenue interfaces and are configured to be active-active MNHA mode and decision to send traffic to the SRX would be determined on the routing done through the fabric. The nodes communicate with each other using an inter-chassis link (ICL) which is logical link. It is recommended to bind the ICL to the loopback interface and have more than one physical link (LAG/LACP) to ensure path diversity for the highest resiliency. The nodes backup each other to ensure a fast synchronized failover in case of system or hardware failure.

> **NOTE**: This JVDE specifies a deployment model in which the SRX4600 is connected directly to the spine switches, representing one supported method of integrating the firewall into the data center fabric. Alternatively, connecting SRX4600 to the leaf switch is also a valid design, however this is not covered in this JVD. Both architectures are fully supported and considered equally viable design options. Connecting the SRX4600 to the spine keeps the design as an Edge Routed Bridging (ERB) architecture because the VTEPs are not terminated on the SRX4600 devices.

Below set of baseline features that have been validated and advanced features that are supported with this integration with the SRX firewalls in MNHA.

- EVPN Type 5 route signaling
- VXLAN encapsulation and decapsulation
- Firewall policy inspection
- Unified policy

Advanced features

- Application Security:
  - Application Identification
  - Application Based Routing
- Content Security
  - Web Filtering
  - Antivirus
  - Content Filtering

  - Anti-Spam
- IDP
- SSL Inspection
- Advance Threat Prevention
  - Security intelligence
  - Advance Anti Malware
  - Adaptive Threat Profiling
- DNS Security
- Encrypted Traffic Insights
- User/Device Authentication

Juniper Hardware and Software Components

The Juniper products and software versions are listed below. The design documented in this JVDE is considered the baseline representation for the validated solution.

Juniper Hardware Components

Table 1 on page 7 through Table 2 on page 8 lists the platforms tested for this JVDE during initial qualification. For more details on all supported platforms and OS versions, see the Validated Platforms and Software section in the JVDE document.

Table 1: Supported Devices and Positioning

Supported Devices and Positioning **Solution: Secure Data Center Fabric with Juniper SRX**	
DC Server Leaf	QFX5120-48Y-8C
DC ESI Leaf	QFX5120-48YM-8C
Spine	QFX5220-32CD
DC Border Leaf	QFX5130-32CD

Table 1: Supported Devices and Positioning *(Continued)*

Supported Devices and Positioning **Solution: Secure Data Center Fabric with Juniper SRX**	
External Firewalls	SRX4600/SRX4700

Table 2: Juniper Software Version

Juniper Software	
Juniper Products	Software or Image Version
Junos OS Evolved & Junos OS images	23.4R2
Juniper Apstra	6.0

Validated Functionality

The Secure Data Center Fabric with SRX4600 was validated using the following parameters in its configuration:

- This JVDE consists of a 3-stage CLOS with an ERB network architecture using EVPN-VXLAN.
- Servers will be connected and tested both in single-homed and multi-homed configurations.
- In the case of multihomed ESI servers, LACP is enabled between the servers and the leaf switches.
- Both the overlay and underlay of the fabric are built using eBGP.
- Learn and advertise EVPN Type 5 routes.
- SRX4600 in MNHA cluster.
- SRX4600 is peering with Spine1 and Spine2 underlay and overlay.
- SRX4600 creates a new Route Distinguisher and pushes the routes back to the fabric to attract the traffic.
- VRF to VRF traffic(East to West) and VRF to internet traffic are inspected (North-South).
- SRX is part of the EVPN VXLAN fabric performing security inspection of user traffic.

Configuration Walkthrough

IN THIS SECTION

- Apstra Configuration for Configuring SRX4600 | 9
- Create Generic Server for SRX4600 and add links | 9
- Assign ASN Numbers to SRX4600 | 11
- Create Underlay Connectivity from SRX Device to the Spine | 14
- Assign Underlay IP Addresses to Spine and SRX Device Interfaces | 16
- Configure Second SRX Device Connectivity to Spines | 16
- Committing Configuration on Spines | 17
- SRX Configuration | 20

Apstra Configuration for Configuring SRX4600

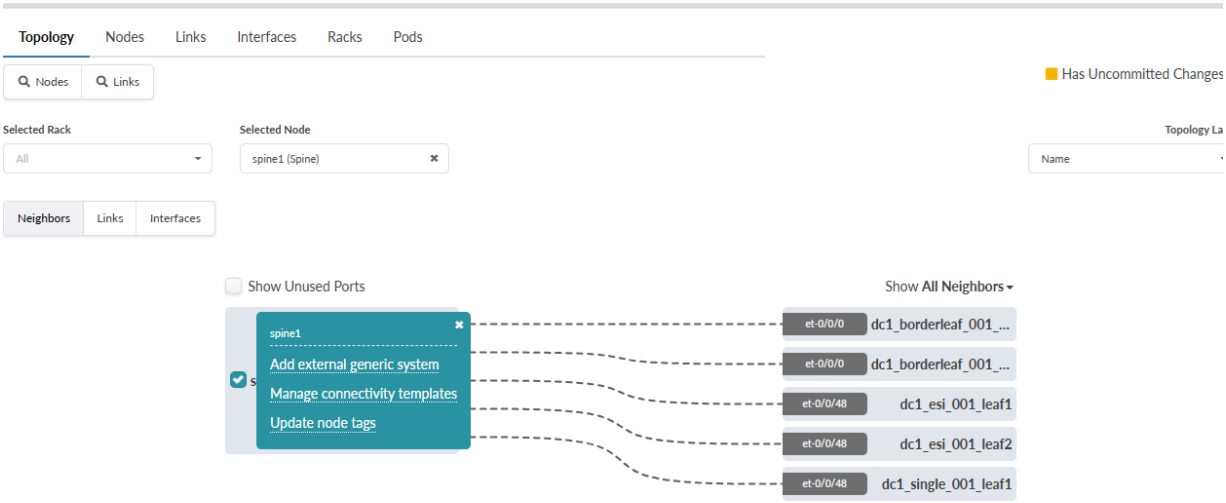
This JVDE walks through the steps to create connectivity between the SRX4600 and 3-stage data center fabric Spine switches using Juniper Apstra. For the purposes of this JVDE, two SRX4600 are connected to each Spine switch. In the current Apstra version, SRX4600 is not supported under device profiles. Therefore, the two SRX4600 are added as external generic servers connected to the Spines and must be added as a Remote EVPN Gateway (also known as external Gateway). Hence Apstra does not manage the SRX4600.

NOTE: For high availability, the SRX devices are configured as MNHA. For more information on MNHA refer to the SRX user guide or to the techpost that provides different architecture deployments.

Create Generic Server for SRX4600 and add links

The process discussed here covers configurations for the first SRX4600 as the same steps apply to the second SRX4600. For more information on adding SRX4600 as an external generic system refer to the Apstra User guide.

Figure 3: Apstra Add External Generic System



After creating a generic server for SRX devices, add links to each Spine to create physical connectivity to the spine. See the Apstra guide for adding links. A link to Spine 1 and another link to Spine-2 is created as shown below in Figure 4 on page 10 and Figure 5 on page 11.

Figure 4: SRX4600 to Spine1 Link Created Using Apstra

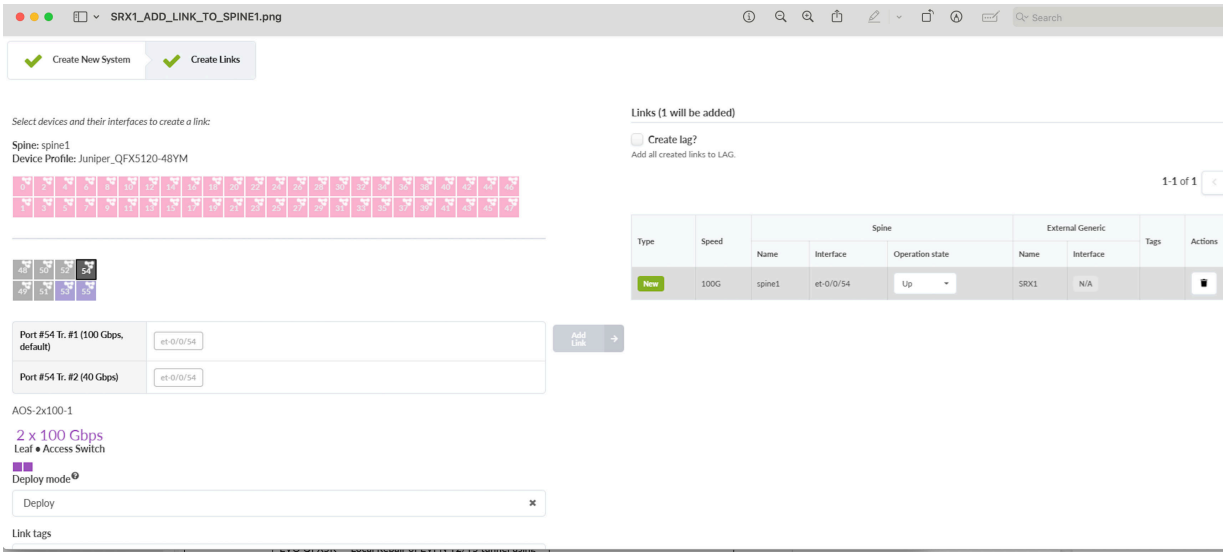
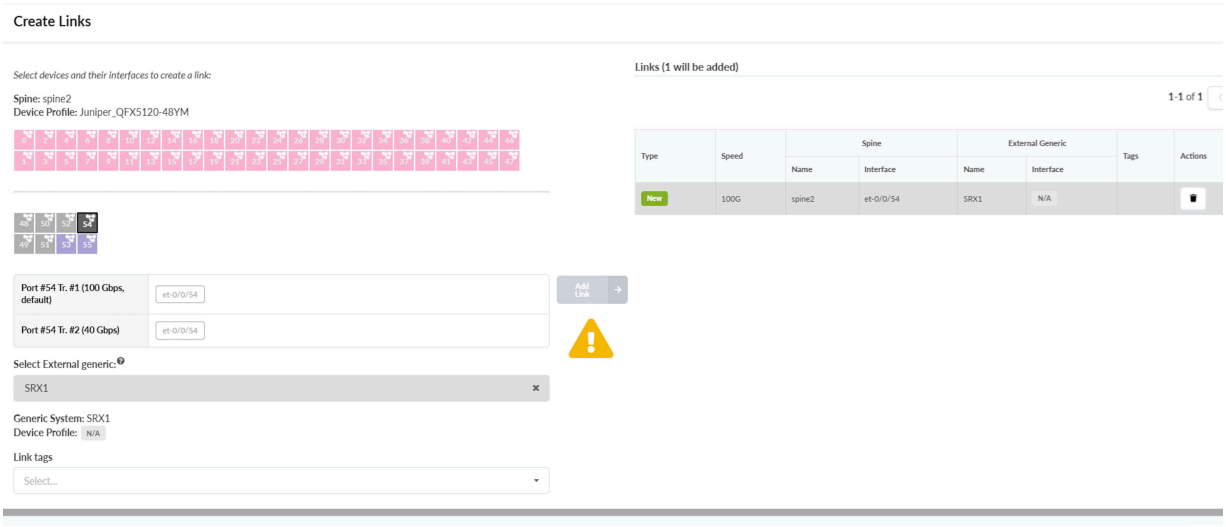


Figure 5: SRX4600 to Spine2 Link Created Using Apstra

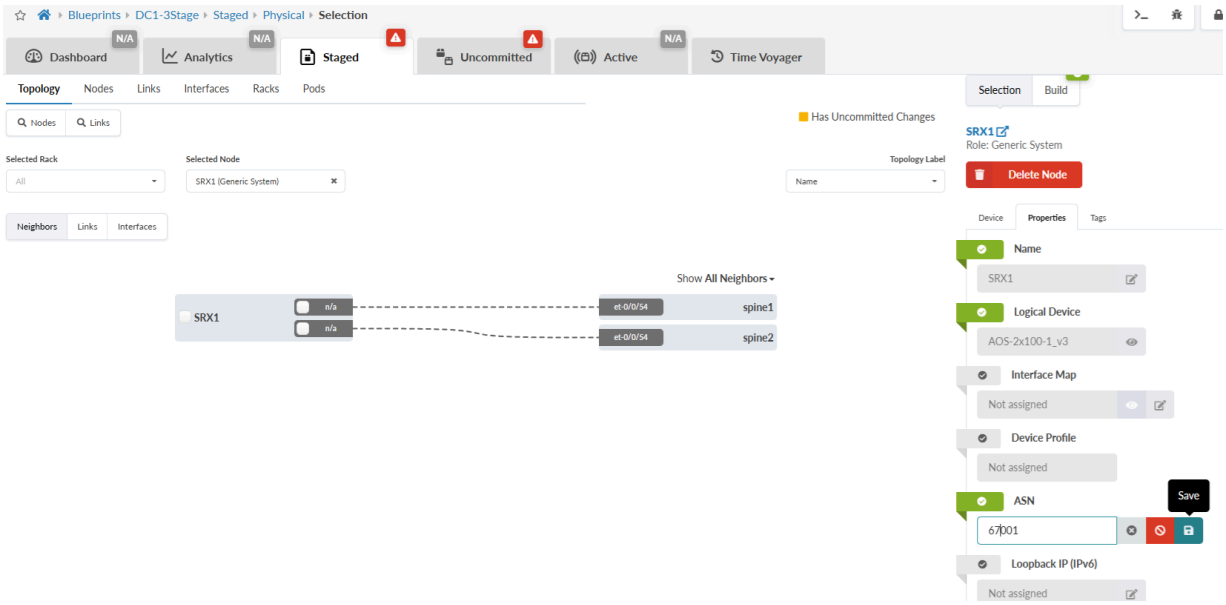


NOTE: Ensure that the cabling correctly reflects the interface names for Spine switches. Navigate to **Blueprint** > {*Your Blueprint*} > **Staged** > **Physical** and then click on link and edit cabling.

Assign ASN Numbers to SRX4600

Next assign ASN numbers to the SRX4600 as shown in Figure 6 on page 12.

Figure 6: Assign ASN Numbers to SRX4600

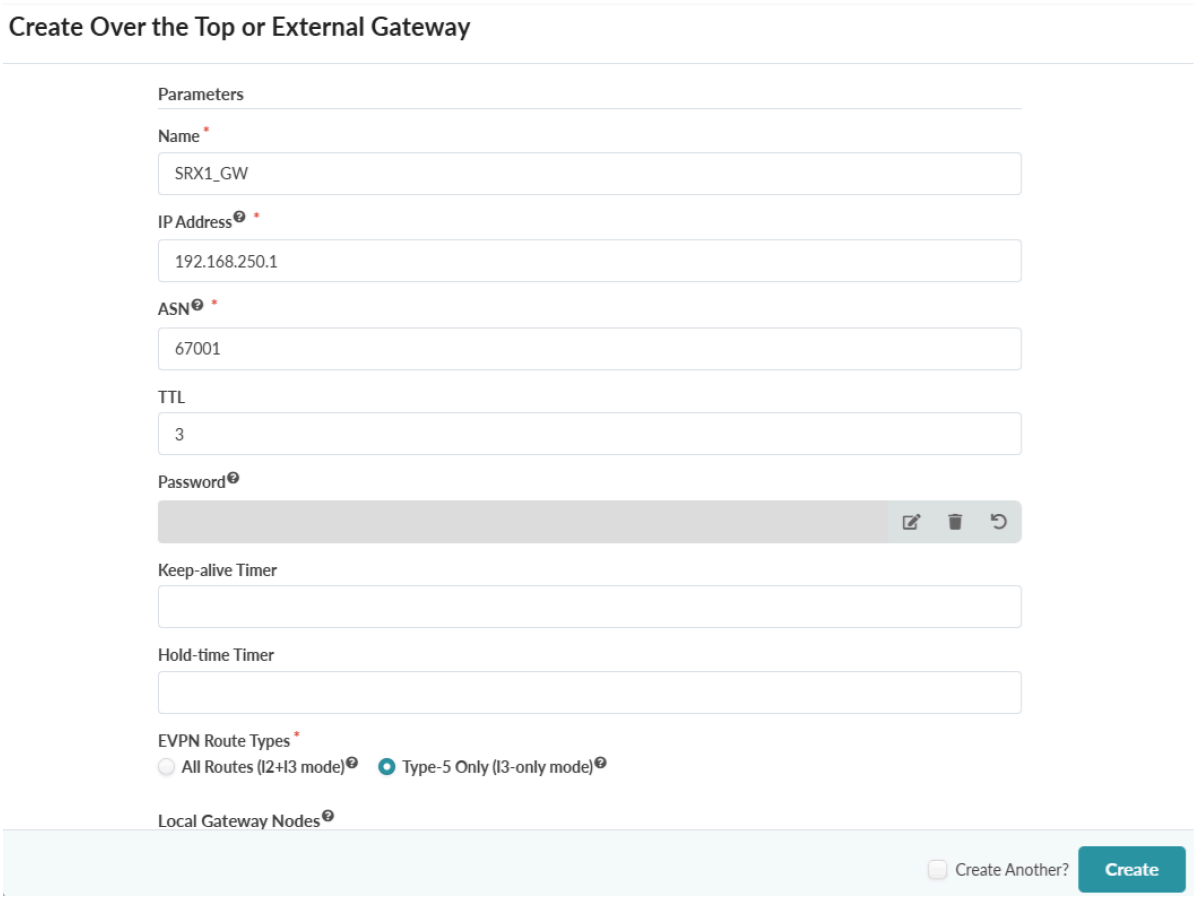


SRX Device as Remote EVPN Gateway (External Gateway) in Apstra

The SRX devices are added as external gateways/Remote EVPN Gateway in Apstra. Refer the Apstra guide for information on the External Gateway/Remote EVPN Gateway. Apstra guide also covers the steps to add Remote EVPN Gateway to extend the control plane.

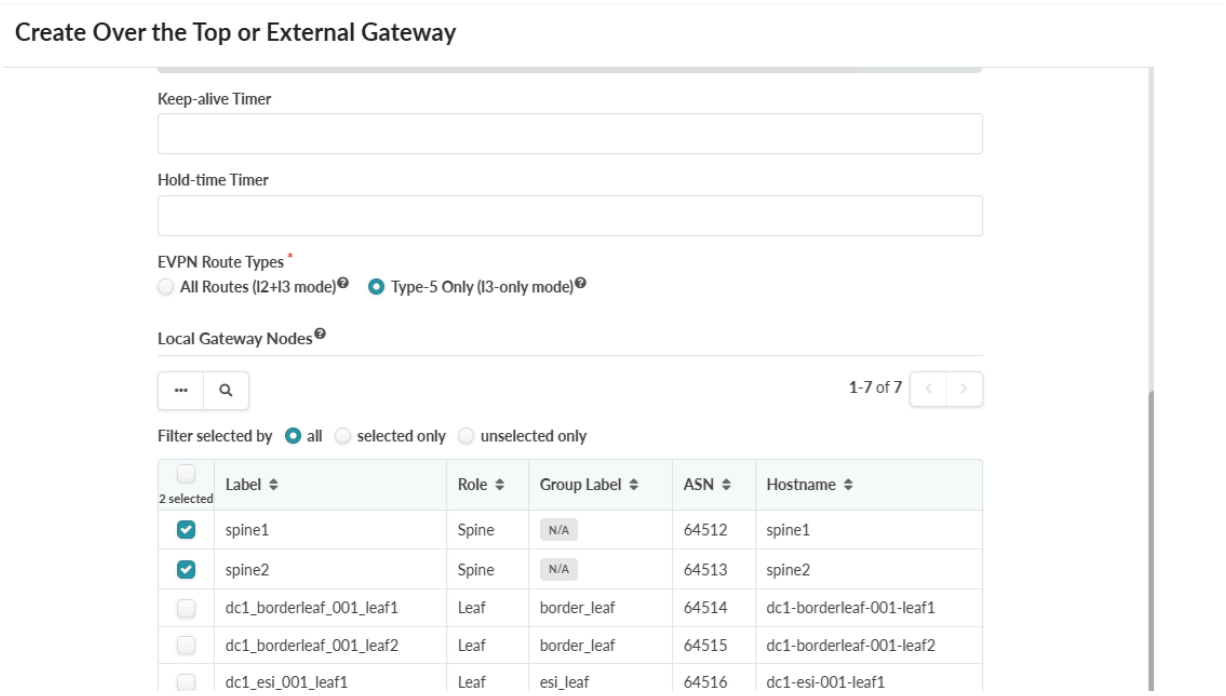
From **Blueprints** > {***Blueprint name***} > **Staged** > **DCI** navigate to Over the top or External Gateways and click on **Create over the top or External Gateway** and create external gateway.

Figure 7: Creating External Gateway for each SRX4600



Assign the SRX gateway to the Spines as shown in Figure 8 on page 14.

Figure 8: Assigning SRX to the Spines to Create evpn Gateway



Create Underlay Connectivity from SRX Device to the Spine

Use the Create Connectivity Template in Apstra to create the underlay connectivity by configuring the routing connections from each SRX device to the different Spine switches. To create the underlay connectivity, assign the primitive types for IP link, BGP Peers and Routing Policy in Apstra Connectivity templates. For more information on Connectivity Templates see the Apstra Guide.

As shown in Figure 9 on page 15, connectivity template is created and the Default Routing Zone is used for VRF to the SRX devices, BGP peering is created between SRX device and Spine, and the IPV4 address is used to setup for connectivity. Next the Routing policy is also assigned for route exchange.

After creating the connectivity template, the template is then assigned to the Spine switches as shown in Figure 10 on page 15.

Figure 9: SRX Devices Underlay Connectivity Template

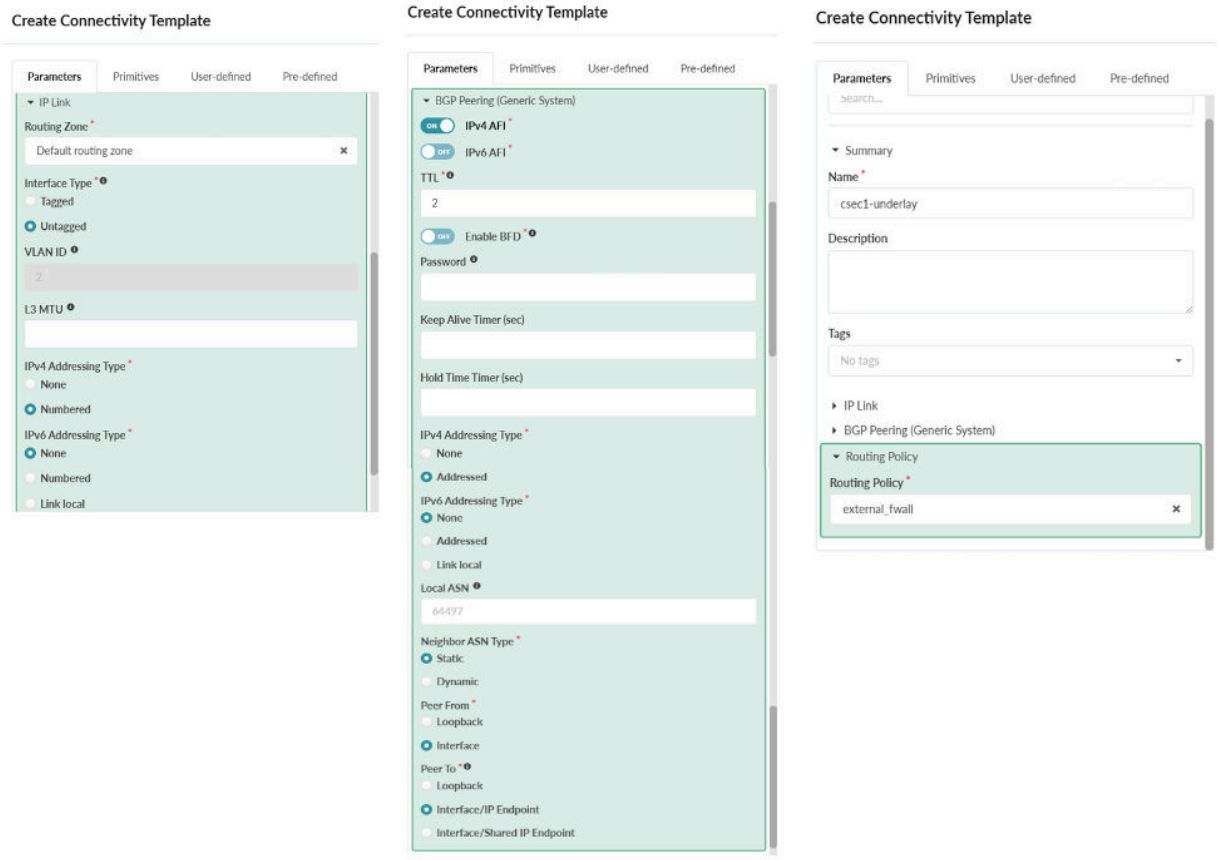
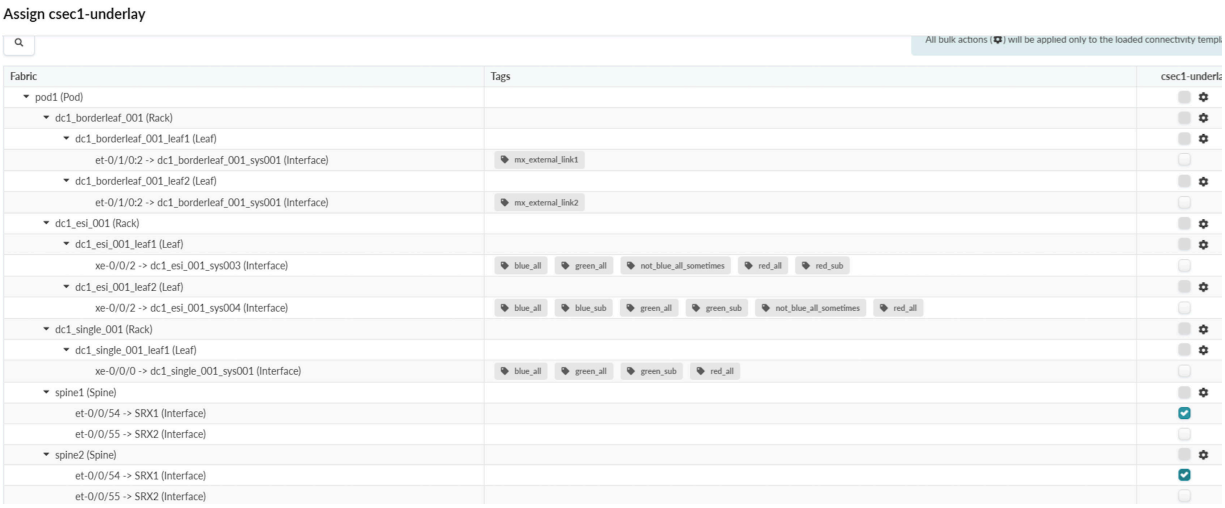


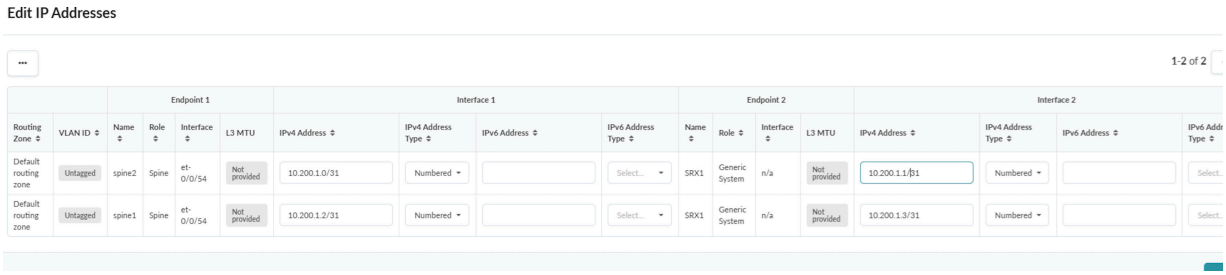
Figure 10: Assign Connectivity Template to the Spines Facing the SRX Device Interfaces



Assign Underlay IP Addresses to Spine and SRX Device Interfaces

Once the connectivity template is assigned, Apstra displays build errors (under **Blueprints** > {***Blueprint Name***} > **Uncommitted**) with suggestions when IP addresses need to be assigned to the interfaces. Navigate from the **Blueprint** > {*Your Blueprint*} > **Staged** > **Virtual Network** > **Routing Zones** and edit the Default Routing Zone as was assigned on the Connectivity template and scroll down to assign IP addresses as shown in Figure 11 on page 16.

Figure 11: Assign IP Addresses to Spine and SRX Device



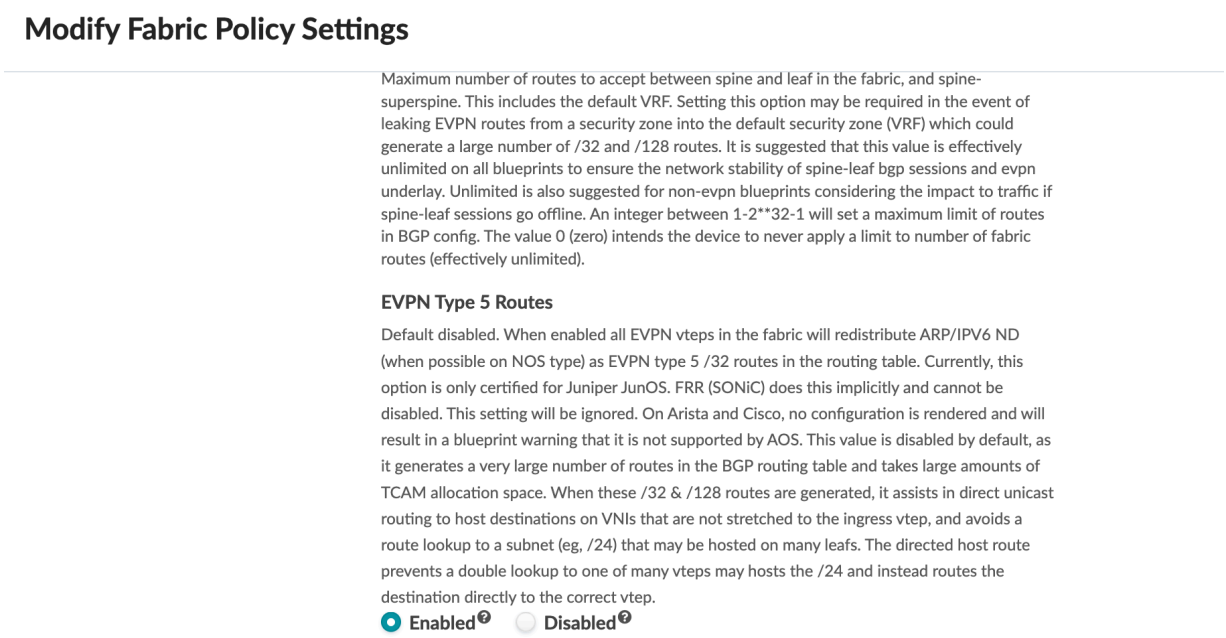
Configure Second SRX Device Connectivity to Spines

Once all the above configuration in Apstra is completed, repeat all the above steps for the second SRX device to create the underlay connectivity and the Remote EVPN gateway to Spine switches.

Enabling EVPN Type 5 Host Specific Routes

For host specific routes in Apstra Fabric setting, enable EVPN Type 5 routes as shown in Figure 12 on page 17. This will increase routes in the BGP routing table depending on the number of hosts in the fabric. The default setting for EVPN Type 5 routes is disabled. Navigate to **Blueprint > Staged > Fabric Settings**. If this setting is disabled, then the routes shared will be the subnet prefix that is configured on the Virtual Network IP Subnet.

Figure 12: Fabric Setting to Enable Host Specific IP Routes



Committing Configuration on Spines

Once all the connectivity is configured in Apstra, commit the configuration to push the configuration on the Spine switches by navigating to **Blueprints** > { ***Blueprint name***} > **Uncommitted**. For more information on committing the configuration, see Apstra guide. At this point any issues or errors should be cleared for the commit to be successful. However, the connectivity between Spine and SRX device will not be established until the SRX device is configured. This is discussed in next section of this JVDE.

After committing the configuration in Apstra, the below BGP configuration for the underlay and Overlay BGP peering with SRX is applied by Apstra on Spine switches. The configuration below shows the import and export of routes between Spine and the connected SRX device. In an EVPN-VXLAN data center fabric, routing information is distributed to all leaf switches. This allows each leaf to learn every route and its associated next hop, enabling traffic forwarding toward other leaf switches or toward external destinations. Through the overlay BGP peering, EVPN Type 5 routes are shared by Spine with SRX firewall. As a result, the SRX learns all Type-5 routes originating from the leaf switches.

By default, the behaviour of the `EBGP multi-hop` statement is to rewrite the next hop. But to prevent this, Apstra also applies the `no-nexthop-change` statement to prevent the Spine switches from changing the next-hop for routes received from the leaf switches. Since leaf switches learn the SRX firewall as the next-

hop, traffic is steered towards the SRX firewall where appropriate security policies are applied for relevant Inter-VRF traffic as well as North-South traffic.

```
protocols {
   bgp {
        group l3rtr {
            type external;
            multihop {
                ttl 1;
            }
            family inet {
                unicast {
                    loops 2;
                }
            }
            family inet6 {
                unicast {
                    loops 2;
                }
            }
            multipath {
                multiple-as;
            }
            neighbor 10.200.1.1 {
                description facing_srx1;
                multihop {
                    ttl 2;
                }
                local-address 10.200.1.0;
                import ( RoutesFromExt-default-external_fwall );
                family inet {
                    unicast;
                }
                export ( RoutesToExt-default-external_fwall );
                peer-as 67001;
            }
            neighbor 10.200.1.5 {
                description facing_srx2;
                multihop {
                    ttl 2;
                }
                local-address 10.200.1.4;
```

```
                import ( RoutesFromExt-default-external_fwall );
                family inet {
                    unicast;
                }
                export ( RoutesToExt-default-external_fwall );
                peer-as 67002;
            }
            vpn-apply-export;
        }
        group evpn-gw {
            type external;
            multihop {
                ttl 30;
                no-nexthop-change;
            }
            multipath {
                multiple-as;
            }
            neighbor 192.168.250.1 {
                description facing_srx1-gw-evpn-gateway;
                multihop {
                    ttl 10;
                }
                local-address 192.168.255.5;
                hold-time 9;
                import ( EVPN_GW_IN_TYPE5 );
                family evpn {
                    signaling;
                }
                export ( EVPN_GW_OUT_TYPE5 );
                peer-as 67001;
            }
            neighbor 192.168.250.2 {
                description facing_srx2-gw-evpn-gateway;
                multihop {
                    ttl 10;
                }
                local-address 192.168.255.5;
                hold-time 9;
                import ( EVPN_GW_IN_TYPE5 );
                family evpn {
                    signaling;
                }
```

```
                export ( EVPN_GW_OUT_TYPE5 );
                peer-as 67002;
            }
            vpn-apply-export;
        }
        log-updown;
        graceful-restart {
            dont-help-shared-fate-bfd-down;
        }
       multipath;
   }
```

SRX Configuration

Multi-node high availability configuration

For Multinode high availability with two SRX4600 firewall devices, high availability configuration was set up as shown below on one of the nodes. The Inter-chassis link is configured for synchronization between the SRX firewall devices/nodes. The ICL is bound to the loopback interface, and an aggregated interface is used to ensure resiliency. The security zones are created for the ICL zone, and the policy are configured to allow communication between the nodes over the ICL link.

```
chassis {
    high-availability {
        local-id {
            2;
            local-ip 192.168.250.11;
        }
        peer-id 1 {
            peer-ip 192.168.250.10;
            interface ae0.0;
            routing-instance ICL;
            liveness-detection {
                minimum-interval 500;
                multiplier 3;
            }
        }
    }

 }
```

```
security {

    policies {

        from-zone ICL to-zone ICL {
            policy allow {
                match {
                    source-address default;
                    destination-address default;
                    application any;
                }
                then {
                    permit;
                }
            }
        }
    }

    zones {

        security-zone ICL {
            host-inbound-traffic {
                system-services {
                    ike;
                    high-availability;
                    ping;
                }
                protocols {
                    bfd;
                }
            }
            interfaces {
                ae0.0;
                lo0.1;
            }
        }
    }
}
interfaces {
    xe-1/1/0 {
        ether-options {
            802.3ad ae0;
        }
```

```
    }
    xe-1/1/1 {
        ether-options {
            802.3ad ae0;
        }
    }
    xe-1/1/2 {
        ether-options {
            802.3ad ae0;
        }
    }
    xe-1/1/3 {
        ether-options {
            802.3ad ae0;
        }
    }
    ae0 {
        unit 0 {
            family inet {
                address 10.222.222.2/30;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.250.2/32;
            }
        }
        unit 1 {
            family inet {
                address 192.168.250.11/32;
            }
        }
    }
}
routing-instances {
    ICL {
        instance-type virtual-router;
        routing-options {
            static {
                route 192.168.250.10/32 next-hop 10.222.222.1;
            }
```

```
        }
        interface ae0.0;
        interface lo0.1;
    }
}
```

BGP peering between SRX and spine

After configuring the spines to connect to SRX devices. Next configure the underlay and overlay peering on SRX devices:

```
root@mft-srx4600-01> show configuration protocols bgp
group underlay {
    type external;
    export loops;
    neighbor 10.200.1.0 {
        peer-as 64512;
    }
    neighbor 10.200.1.2 {
        peer-as 64513;
    }
}
group overlay {
    type external;
    multihop;
    advertise-peer-as;
    family inet {
        unicast;
    }
    family evpn {
        signaling;
    }
    export overlay-out;
    multipath {
        multiple-as;
    }
    as-override;
    neighbor 192.168.255.5 {
        local-address 192.168.250.1;
        peer-as 64512;
    }
    neighbor 192.168.255.6 {
```

```
        local-address 192.168.250.1;
        peer-as 64513;
    }
    vpn-apply-export;
}
```

Next, configure the policy options to export/import routes for the VRF and define community.

```
root@mft-srx4600-01> show configuration policy-options
policy-statement blue_EXPORT {
    term 1 {
        then {
            community add COMM_blue;
            accept;
        }
    }
}
policy-statement blue_IMPORT {
    term 1 {
        from community [ COMM_blue COMM_red COMM_green COMM_yellow ];
        then accept;
    }
}
policy-statement green_EXPORT {
    term 1 {
        then {
            community add COMM_green;
            accept;
        }
    }
}
policy-statement green_IMPORT {
    term 1 {
        from community [ COMM_blue COMM_red COMM_green COMM_yellow ];
        then accept;
    }
}
policy-statement loops {
    term 1 {
        from {
            protocol direct;
            interface lo0.0;
```

```
        }
        then accept;
    }
}
policy-statement overlay-out {
    term 1 {
        from {
            instance red;
            family evpn;
            route-filter 0.0.0.0/0 exact;
        }
        then accept;
    }
    term 2 {
        from {
            instance red;
            family evpn;
            route-filter 0::0/0 exact;
        }
        then accept;
    }
    term 10 {
        from {
            instance blue;
            family evpn;
            route-filter 0.0.0.0/0 exact;
        }
        then accept;
    }
    term 11 {
        from {
            instance blue;
            family evpn;
            route-filter 0::0/0 exact;
        }
        then accept;
    }
    term 20 {
        from {
            instance green;
            family evpn;
            route-filter 0.0.0.0/0 exact;
        }
```

```
        then accept;
    }
    term 21 {
        from {
            instance green;
            family evpn;
            route-filter 0::0/0 exact;
        }
        then accept;
    }
    term 700 {
        from {
            instance red;
            family evpn;
        }
        then reject;
    }
    term 800 {
        from {
            instance green;
            family evpn;
        }
        then reject;
    }
    term 900 {
        from {
            instance blue;
            family evpn;
        }
        then reject;
    }
}
policy-statement red_EXPORT {
    term 1 {
        then {
            community add COMM_red;
            accept;
        }
    }
}
policy-statement red_IMPORT {
    term 1 {
        from community [ COMM_red COMM_blue COMM_green COMM_yellow ];
```

```
        then accept;
    }
}
policy-statement yellow_EXPORT {
    term 1 {
        then {
            community add COMM_yellow;
            accept;
        }
    }
}
policy-statement yellow_IMPORT {
    term 1 {
        from community [ COMM_red COMM_blue COMM_green COMM_yellow ];
        then accept;
    }
}
community COMM_blue members target:20002:1;
community COMM_green members target:20004:1;
community COMM_red members target:20001:1;
community COMM_yellow members target:20005:1;
root@mft-srx4600-01>
```

Configure the routing instance with EVPN-VXLAN configuration and include the different route distinguishers. Then, configure the VRF or routing instance with import and export policies to allow inter-VRF communication.

```
root@mft-srx4600-01> show configuration routing-instances blue
instance-type vrf;
routing-options {
    rib blue.inet6.0 {
        static {
            route 0::0/0 {
                discard;
                preference 180;
            }
        }
    }
    static {
        route 0.0.0.0/0 {
            discard;
            preference 180;
```

```
        }
    }
    multipath;
}
protocols {
    evpn {
        interconnect {
            vrf-target target:20002:1;
            route-distinguisher 192.168.250.1:3;
        }
        ip-prefix-routes {
            advertise direct-nexthop;
            encapsulation vxlan;
            vni 20002;
        }
    }
}
route-distinguisher 192.168.250.1:11;
vrf-import blue_IMPORT;
vrf-export blue_EXPORT;
vrf-table-label;
root@mft-srx4600-01> show configuration routing-instances yellow
instance-type vrf;
routing-options {
    multipath;
}
protocols {
    evpn {
        interconnect {
            vrf-target target:20005:1;
            route-distinguisher 192.168.250.1:5;
        }
        ip-prefix-routes {
            advertise direct-nexthop;
            encapsulation vxlan;
            vni 20005;
        }
    }
}
route-distinguisher 192.168.250.1:13;
vrf-import yellow_IMPORT;
vrf-export yellow_EXPORT;
vrf-table-label;
```

```
root@mft-srx4600-01> show configuration routing-instances green
instance-type vrf;
routing-options {
    rib green.inet6.0 {
        static {
            route 0::0/0 {
                discard;
                preference 180;
            }
        }
    }
    static {
        route 0.0.0.0/0 {
            discard;
            preference 180;
        }
    }
    multipath;
}
protocols {
    evpn {
        interconnect {
            vrf-target target:20004:1;
            route-distinguisher 192.168.250.1:4;
        }
        ip-prefix-routes {
            advertise direct-nexthop;
            encapsulation vxlan;
            vni 20004;
        }
    }
}
route-distinguisher 192.168.250.1:12;
vrf-import green_IMPORT;
vrf-export green_EXPORT;
vrf-table-label;
root@mft-srx4600-01>
root@mft-srx4600-01>
root@mft-srx4600-01> show configuration routing-instances red
instance-type vrf;
routing-options {
    rib red.inet6.0 {
        static {
```

```
            route 0::0/0 {
                discard;
                preference 180;
            }
        }
    }
    static {
        route 0.0.0.0/0 {
            discard;
            preference 180;
        }
    }
    multipath;
}
protocols {
    evpn {
        interconnect {
            vrf-target target:20001:1;
            route-distinguisher 192.168.250.1:2;
        }
        ip-prefix-routes {
            advertise direct-nexthop;
            encapsulation vxlan;
            vni 20001;
        }
    }
}
route-distinguisher 192.168.250.1:10;
vrf-import red_IMPORT;
vrf-export red_EXPORT;
vrf-table-label;
```

A VRF group bundles multiple VRFs (routing instances) together so they can be governed by a single security policy.

```
root@mft-srx4600-01> show configuration security l3vpn
vrf-group VRF_GROUP1 {
    vrf [ blue green ];
}
root@mft-srx4600-01>
```

Define the security zone

To define a security policy, it is essential to understand the concept of security zones within your network. The concept of zones is a fundamental building block for security architecture on Juniper devices. It is used to segment the network based on trust levels and enforces granular control on the flow of traffic between the different network segments.

A security zone on a Juniper SRX firewall is a logical grouping of one or more interfaces that share the same security requirements. Instead of applying security policies to individual interfaces, you can apply them to zones. This simplifies management and configuration. All interfaces must belong to a zone. By default, interfaces are placed in a "null zone" where all traffic is dropped. Traffic is allowed only when the interface is configured in a zone as shown below.

```
root@mft-srx4600-01> show configuration security zones security-zone untrust
host-inbound-traffic {
    system-services {
        all;
    }
    protocols {
        all;
    }
}
interfaces {
    et-1/0/0.0;
    et-1/0/1.0;
    lo0.0;
}
```

Define the security policy

In the configuration below, we created an intra zone policy, where traffic enters the SRX device from zone “untrust” and exits through zone “untrust” on the SRX device.

```
root@mft-srx4600-01> show configuration security policies from-zone untrust to-zone untrust
policy VRF-BLOCK {
    match {
        source-address [ blue_hosts green_hosts green_sub_hosts blue_sub_hosts blue_v6_hosts
green_v6_hosts blue_v6_sub_hosts green_v6_sub_hosts default default_v6 ];
        destination-address [ red_hosts red_sub_hosts red_v6_hosts red_sub_v6_hosts ];
        application any;
        source-l3vpn-vrf-group VRF_GROUP1;
    }
    then {
        reject;
```

```
    }
}
policy VRF-POl1 {
    match {
        source-address [ blue_hosts green_hosts green_sub_hosts blue_sub_hosts blue_v6_hosts
green_v6_hosts blue_v6_sub_hosts green_v6_sub_hosts ];
        destination-address [ blue_hosts blue_sub_hosts green_hosts green_sub_hosts
blue_v6_hosts green_v6_hosts blue_v6_sub_hosts green_v6_sub_hosts default default_v6 ];
        application any;
        source-l3vpn-vrf-group VRF_GROUP1;
    }
    then {
        permit;
    }
}
policy untr-untr-bfd {
    match {
        source-address [ SPINE1-LOOPBACK SPINE2-LOOPBACK LOOPBACK ];
        destination-address [ SPINE1-LOOPBACK SPINE2-LOOPBACK LOOPBACK ];
        application any;
    }
    then {
        permit;
    }
}
root@mft-srx4600-01>
```

To verify that the firewalls are functional and the changes are configured, log into the console or CLI of each of the firewalls. From the shell, enter the following Junos OS CLI command:

```
show bgp summary | no-more
```

The output of this command should be similar to the output below. It shows that BGP is established from each spine to SRX underlay and overlay.

```
root@mft-srx4600-01> show bgp summary
Threading mode: BGP I/O
Default eBGP mode: advertise - accept, receive - accept
Groups: 3 Peers: 4 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
```

```
                      12          7          0          0          0          0
bgp.evpn.0
                   11060       5530          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.200.1.0            64512        147        146       0       0     1:04:40 Establ
  inet.0: 6/6/6/0
10.200.1.2            64513        146        150       0       0     1:04:28 Establ
  inet.0: 1/6/6/0
192.168.255.5         64512       1839       1735       0       0     1:04:30 Establ
  bgp.evpn.0: 3537/5530/5530/0
  blue.evpn.0: 3537/5530/5530/0
  green.evpn.0: 3537/5530/5530/0
  red.evpn.0: 3537/5530/5530/0
  yellow.evpn.0: 3537/5530/5530/0
192.168.255.6         64513       1832       1807       0       0     1:04:26 Establ
  bgp.evpn.0: 1993/5530/5530/0
  blue.evpn.0: 1993/5530/5530/0
  green.evpn.0: 1993/5530/5530/0
  red.evpn.0: 1993/5530/5530/0
  yellow.evpn.0: 1993/5530/5530/0
root@mft-srx4600-01>
```

If the output of the *show bgp summary | no-more* command resembles the screenshot above, a bare-bones network fabric is now complete with SRX integration.

Verify EVPN under each routing instance.

```
root@mft-srx4600-01> show evpn l3-context extensive
L3 context: blue
  Type: Configured
  Advertisement mode: Direct nexthop, Router MAC: 9c:5a:80:3e:31:c4
  Encapsulation: VXLAN, VNI: 20002
  IPv4 source VTEP address: 192.168.250.1
  Flags: 0x61d <Configured Sys-MAC IRB-MAC New ROUTING ROUTING-NEW>
  Change flags: 0x33c0c <Adv-Mode Encap Dci-Config Dci-Rd-Change Dci-Import-Policy Dci-Export-
Policy VXLAN-VNI-Update-RTT-OPQ>
  Composite nexthop support: Disabled
  Route Distinguisher: 192.168.250.1:11
  Reference count: 3343
  EVPN Multicast Routing mode: CRB
L3 context: green
```

```
  Type: Configured
  Advertisement mode: Direct nexthop, Router MAC: 9c:5a:80:3e:31:c4
  Encapsulation: VXLAN, VNI: 20004
  IPv4 source VTEP address: 192.168.250.1
  Flags: 0x61d <Configured Sys-MAC IRB-MAC New ROUTING ROUTING-NEW>
  Change flags: 0x33c0c <Adv-Mode Encap Dci-Config Dci-Rd-Change Dci-Import-Policy Dci-Export-
Policy VXLAN-VNI-Update-RTT-OPQ>
  Composite nexthop support: Disabled
  Route Distinguisher: 192.168.250.1:12
  Reference count: 3343
  EVPN Multicast Routing mode: CRB
L3 context: red
  Type: Configured
  Advertisement mode: Direct nexthop, Router MAC: 9c:5a:80:3e:31:c4
  Encapsulation: VXLAN, VNI: 20001
  IPv4 source VTEP address: 192.168.250.1
  Flags: 0x61d <Configured Sys-MAC IRB-MAC New ROUTING ROUTING-NEW>
  Change flags: 0x33c0c <Adv-Mode Encap Dci-Config Dci-Rd-Change Dci-Import-Policy Dci-Export-
Policy VXLAN-VNI-Update-RTT-OPQ>
  Composite nexthop support: Disabled
  Route Distinguisher: 192.168.250.1:10
  Reference count: 3343
  EVPN Multicast Routing mode: CRB
L3 context: yellow
  Type: Configured
  Advertisement mode: Direct nexthop, Router MAC: 9c:5a:80:3e:31:c4
  Encapsulation: VXLAN, VNI: 20005
  IPv4 source VTEP address: 192.168.250.1
  Flags: 0x61d <Configured Sys-MAC IRB-MAC New ROUTING ROUTING-NEW>
  Change flags: 0x33c0c <Adv-Mode Encap Dci-Config Dci-Rd-Change Dci-Import-Policy Dci-Export-
Policy VXLAN-VNI-Update-RTT-OPQ>
  Composite nexthop support: Disabled
  Route Distinguisher: 192.168.250.1:13
  Reference count: 3343
  EVPN Multicast Routing mode: CRB
root@mft-srx4600-01>
```

Verify the route leaking is in effect properly across the routing-instance.

```
root@mft-srx4600-01> show route forwarding-table destination 10.1.0.101/32 table blue
Routing table: blue.inet
Internet:
```

```
Destination        Type RtRef Next hop           Type Index    NhRef Netif
10.1.0.0/24        user     0                    comp      885   136
root@mft-srx4600-01> show route forwarding-table destination 10.1.0.101/32 table green
Routing table: green.inet
Internet:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
10.1.0.0/24        user     0                    comp      886   136
root@mft-srx4600-01> show route forwarding-table destination 10.1.0.101/32 table red
Routing table: red.inet
Internet:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
10.1.0.0/24        user     0                    comp      884   136
root@mft-srx4600-01> show route forwarding-table destination 10.1.0.101/32 table yellow
Routing table: yellow.inet
Internet:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
10.1.0.0/24        user     0                    comp      887   136
root@mft-srx4600-01>
```

Validation Framework

IN THIS SECTION

- Test Bed | 35
- VRF Characteristics: | 36
- Platforms / Devices Under Test (DUT) | 37
- Test Bed Configuration | 37

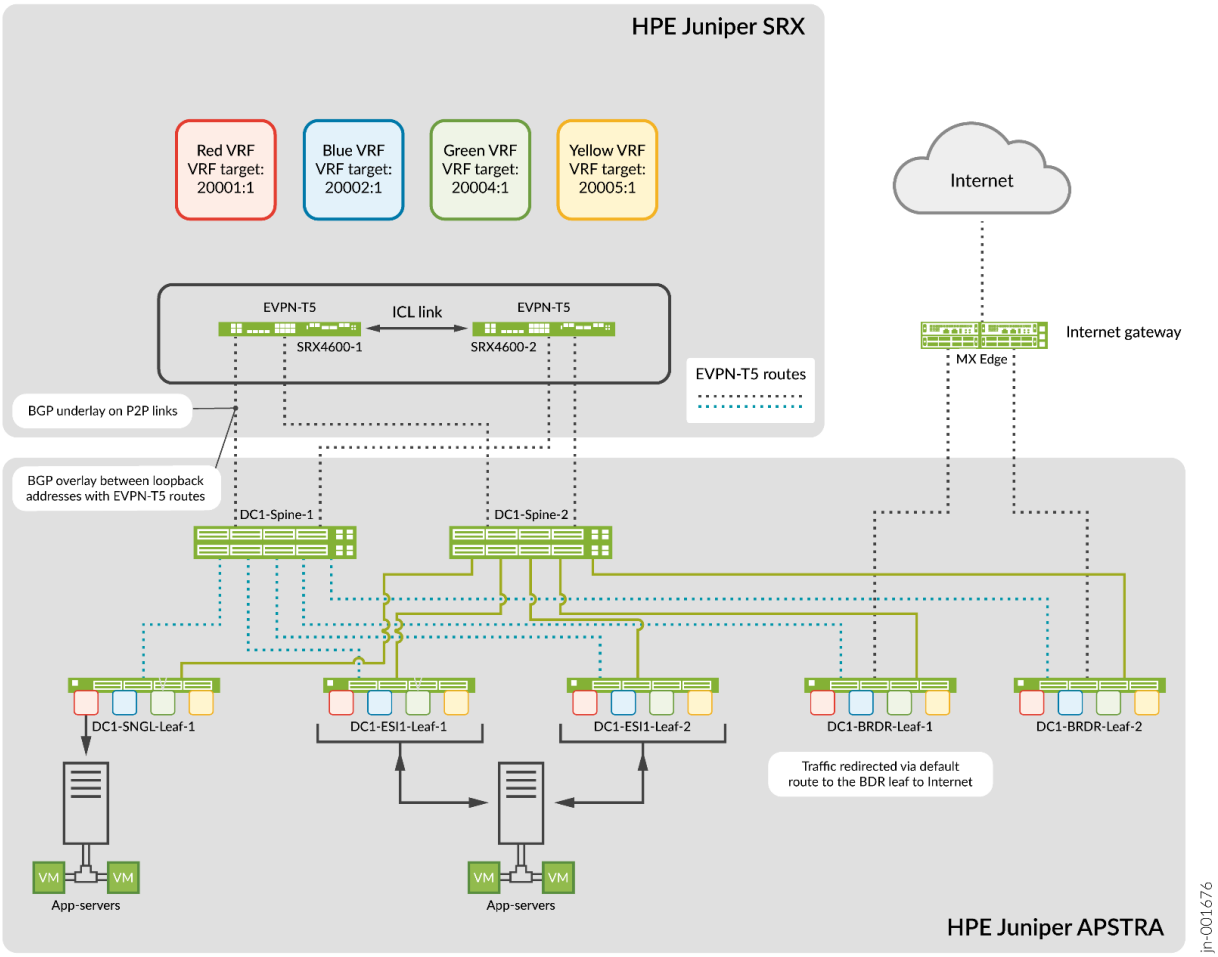
Test Bed

The test bed environment consists of a 3-stage EVPN/VXLAN fabric managed by Juniper Apstra, with four ESI server leaf switches configured as two redundant pairs, one single (non-redundant) server leaf (non-ESI), and two redundant border leaf switches connected to two spines. An external router is also

connected to the border leaf switches. A traffic generator is connected to the test ports on the external router and the ESXi servers.

To ensure all the platforms specified in the Supported Devices and Positioning Table 1 on page 7 are validated, two data center topologies connected using DCI were used. Since there were multiple devices for each role, the devices were swapped, and the tests were repeated with each combination. For instance, border leaf switches were swapped with QFX5130-32CD, PTX10001-36MR, ACX7100-32CD, and so on.

Figure 13: Lab Topology



VRF Characteristics:

Blue VRF

- VNI 20002
- Route Distinguisher: 192.168.250.1:3
- Route Target 20002:1

Green VRF

- VNI 20004
- Route Distinguisher: 192.168.250.1:4
- Route Target 20004:1

Yellow VRF

- VNI 20005
- Route Distinguisher: 192.168.250.1:5
- Route Target 20005:1

Red VRF

- VNI 20001
- Route Distinguisher: 192.168.250.1:2
- Route Target 20001:1

Platforms / Devices Under Test (DUT)

To review the software versions and platforms on which this JVDE was validated by Juniper Networks, see the Validated Platforms and Software section in this document.

Test Bed Configuration

Contact your Juniper Networks representative to obtain the full archive of the test bed configuration used for this JVDE.

Test Objectives

IN THIS SECTION

- Test Goals | 38

The primary objective of this JVDE testing is the qualification testing of the Secure Data Center Fabric with SRX as the firewall. The design is based on an ERB (Type 5) EVPN/VXLAN fabric with the spine, server leaf, and border leaf switches. The SRX4600 is part of the EVPN signalling and route learning. The goal is to ensure the design is well-documented and will produce a reliable, predictable deployment for the customer. The qualification objectives include validation of deployment, device upgrade, incremental configuration pushes/provisioning, Telemetry/Analytics checking, failure mode analysis, and verification of host traffic.

Test Goals

The Secure Data Center Fabric with SRX4600 testing uses the following flow:

- Initial design and deployment
- Validate Apstra AOS can successfully provision the Spine-to-Firewall BGP underlay/overlay
- Validate SRX participation in EVPN signaling (Control Plane)
- Validate SRX capability to decapsulate and encapsulate VXLAN traffic (Data Plane)
- Validate firewall policy process by SRX on the VXLAN traffic
- Validate Application Identification by SRX on EVPN-VXLAN traffic
- Ensure resiliency and redundancy through MNHA at the SRX layer
- Validation of SRX security services operation and monitoring
- Scale testing
- Validation of end-to-end traffic flow
- System health, ARP, ND, MAC, BGP (route, next hop), interface traffic counters, and so on

- Test for anomalies
- To pass validation, the Secure Data Center Fabric with SRX4600 must also pass the following scenarios:
  - Node Reboot - simulated real-world switch outage.
  - Field scenarios such as interface down/up and Laser on/off impact to the fabric and check anomalies reporting in Apstra.
  - Traffic recovery was validated after all failure scenarios.

For more information, see the test report.

Results Summary and Analysis

For the Secure Data Center Fabric with Juniper SRX JVDE, comprehensive functional testing was performed on devices listed in Validated Platforms and Software section in this document:

- Baseline System Test:
  - Add two SRX devices as external generic systems
  - Add connectivity templates for underlay BGP sessions from spines to SRX
  - Add SRXs also as External Gateway devices to establish overlay BGP for Type5 EVPN routes.
- Operational and Trigger Tests:
  - Operational testing of switches was carried out for the following:
    - Daemon failures such as RPD, ppmd, jsrpd
    - Reboot devices cause no issues when the devices boot up
  - Connectivity tests for the following items were carried out:
    - ICL link failure between MNHA nodes
    - Link flap between firewall and spine
    - Link fails between firewall and spine
    - Primary MNHA node fails
    - Spine failure
  - Resiliency tests for overlay connectivity testing for the below scenarios:

- Inter VRF Traffic for application identification
- SSL inspection on inter-VRF traffic and VRF to internet traffic
- Infected host and SecIntel feed for inter VRF traffic and VRF to internet traffic
- Destination NAT on North-South traffic.

Scale Testing numbers are as follows:

Table 3: Multi-dimensional Scale Numbers Tested

Features	Tested Scale Numbers
VLANs VNIs with IRB	1500
Host per VNI per port	5
DCI VTEPS	30
BGP total path	426468
BGP active path	213202

The scale numbers above are not device maximums; they only reference the scale at which these multidimensional test cases are performed.

Overall, the JVDE validation testing didn't detect any issues, and all performance parameters were within the threshold and performed as expected. Traffic profiles were tested on SRX4600 for both inter-VRF and external.

Recommendations

The Secure Data Center Fabric with Juniper SRX4600 JVDE follows an industry-standard ERB design. It simplifies security integration in data centre and secures data center EVPN/VXLAN fabric with Juniper SRX4600/SRX4700. As described in the document, the SRX firewall devices should be configured as MNHA cluster that offers high resiliency in case of node failures. This JVDE uses an industry-standard ERB design, which simplifies security integration and makes it easier to apply security policies to both East-West and North-South traffic. Junos OS Release 24.4R2 is the minimum recommended software version for this JVDE on SRX4600/SRX4700, and has been tested with data center devices

recommended in the 3-Stage Data Center Design with Juniper Apstra (JVD) with 23.4R2 and Apstra 6.0 release.

The Juniper hardware listed in the Validated Platforms and Software section in this document are the best-suited switch platforms in terms of features, performance, and the roles that are specified in this JVDE.

Revision History

Revision History

Date	Version	Description
March 2026	JVD-DCFABRIC-CSEC-01-01	Initial Publish