JUNIPER NETWORKS | Engineering Simplicity

Network Configuration Example

Configuring SR-IOV 10-Gigabit High Availability on vSRX 3.0

Published
2023-09-13

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

Network Configuration Example Configuring SR-IOV 10-Gigabit High Availability on vSRX 3.0

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

YEAR 2000 NOTICE

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

END USER LICENSE AGREEMENT

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

Table of Contents

About This Guide

This document describes the 10-Gigabit high availability single-root I/O virtualization (SR-IOV) deployment scenario for vSRX 3.0 instances.

1

CHAPTER

Configuring SR-IOV 10-Gigabit High Availability on vSRX

About This Network Configuration Example

If you have a physical network interface card (NIC) that supports single-root I/O virtualization (SR-IOV), you can attach SR-IOV-enabled vNICs or virtual functions (VFs) to the vSRX instance to improve the performance. We recommend you to configure all revenue ports of vSRX as SR-IOV if you use SR-IOV on vSRX instances.This document describes different 10-Gigabit high availability and standalone SR-IOV deployment scenarios for vSRX instances. It also provides a step-by-step configuration example for each of the different scenarios.

This document focuses on Juniper Networks® vSRX instances.

Understanding SR-IOV Usage

You can enable communication between a Linux-based virtualized device and a Network Functions Virtualization (NFV) module using suitable hardware and SR-IOV.

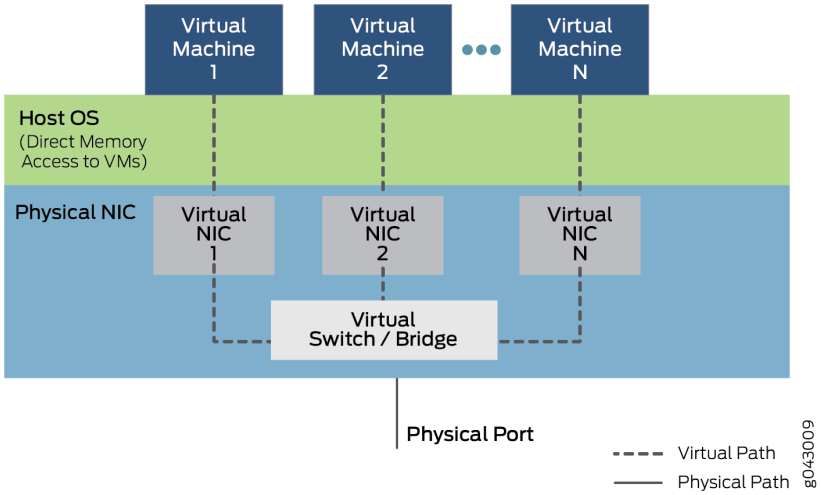
When a physical device is virtualized, both the physical NIC interfaces and external physical switches as well as the virtual NIC interfaces and internal virtual switches coexist. When the isolated virtual machines (VMs) or containers in the device, each with their own memory and disk space and CPU cycles, attempt to communicate with each other, multiple ports, MAC addresses, and IP addresses in use pose a challenge.

SR-IOV extends the concept of virtualized functions down to the physical NIC. The single physical card is divided into partitions per physical NIC port that correspond to the virtual functions running at the higher layers. Communication between these virtual functions are handled the same way that communication between devices with individual NICs are usually handled with a bridge. SR-IOV includes a set of standard methods for creating, deleting, listing, and querying the SR-IOV NIC switch, as well as a set of standard parameters that you can set.

The single-root of SR-IOV refers to only one primary piece of the NIC that controls all operations. An SR-IOV-enabled NIC is a standard Ethernet port that provides the same physical bit-by-bit function of any network card.

The SR-IOV provides several virtual functions, which are accomplished by using simple queues to handle input and output tasks. Each VNF running on the device is mapped to one of the NIC partitions so that the VNFs themselves have direct access to NIC hardware resources. The NIC has a simple Layer 2 sorter function, which classifies frames into traffic queues. Packets are moved directly to and from the network virtual function to the VM's memory using direct memory access (DMA), bypassing the hypervisor completely. The role of the NIC in the SR-IOV operation is shown in Figure 1 on page 3.

Figure 1: VNF Communication Using SR-IOV



The hypervisor is involved in assigning VNFs and managing the physical card, but not in the transfer of the data inside the packets. VNF-to-VNF communication is performed by Virtual NIC 1, Virtual NIC 2, and Virtual NIC N. The NIC also keeps track of all the virtual functions and the sorter to move traffic among the VNFs and external device ports.

SR-IOV support depends on the hardware platform, specifically the NIC hardware, and the software of the VNFs or containers to use DMA for data transfer.

Example: How to Set Up SR-IOV 10GbE High Availability on vSRX 3.0 with Ubuntu on a KVM Server

IN THIS SECTION

This example shows how to set up SR-IOV 10GbE high availability deployment on vSRX 3.0 instances.

Requirements

This example uses the following hardware, software components, and operating systems:

Device

- vSRX 3.0

Software

- Junos OS Release 20.4R1

Hardware

- NIC: Intel Corporation Ethernet Controller X710/X520/82599
- Driver: i40e version: 2.1.14-k or ixgbe version: 5.1.0-k
- CPU: Intel (R) Xeon (R) Gold 5120 CPU @ 2.20 GHz
- 56 CPUs
- 0- 55 online CPUs list
- 2 threads per core
- 14 cores per socket
- 2 sockets
- 2 non-uniform memory access (NUMA) nodes

For more information on NICs, Hypervisors, and ports supported with SR-IOV see Hardware Specifications.

Operating System

Table 1: SR-IOV HA Supported KVM OS and Network Adapter Information

KVM OS and Network Adapters	Support
Intel 82599/X520/X540 (82599 ixgb driver based)	Yes

Table 1: SR-IOV HA Supported KVM OS and Network Adapter Information *(Continued)*

KVM OS and Network Adapters	Support
Intel X710/XL710/XXV710/X722 (i40e driver based)	Yes
Mellanox ConnectX-4/ConnectX-4 Lx	No
Ubuntu 18.04 (kernel:4.15.0 + libvirt:4.0.0) and 20.04 (kernel:5.4.0 + libvirt:6.0.0) LTS	Yes
Redhat 8.2 (kernel:4.18.0 + libvirt:4.5.0)	Yes

Operating Systems used in this example are:

- Ubuntu 18.04.3 LTS on a KVM server
- Kernel: 4.15.0-64-generic
- Kernel: 4.18.0-193.1.2.el8_2.x86_64
- redhat rhel 8.2

Overview

This example shows how to:

- Set up the 10-Gigabit high availability deployment
- Build VFs bus information on NIC interfaces and change the XML template
- Configure basic vSRX 3.0 instances

In a high availability environment, the control link and fabric data links are key communication channels for chassis cluster stability. Both links are part of the same Linux bridge. The host operating system (Ubuntu) shares the CPU allotted for the vSRX 3.0 control plane for routine tasks and with one of the vSRX 3.0 PFE data plane threads for packet processing. This contention for resources coupled with the lack of a dedicated VLAN or NIC for the control link could contribute to heartbeat misses.

NOTE: Fabric links over SR-IOV are not supported when the `use-actual-mac-on-physical-interfaces` command is in use.

Furthermore, interrupt handling on the host can also impact the performance. When packets arrive at the NIC, a hardware interrupt indication and the CPU core that services the vSRX 3.0 control plane must stop and service the interrupt. A large number of packets from the NIC can lead to more hardware interruptions and less CPU resources to service the vSRX 3.0 control plane.

To overcome the design constraints and the CPU resource contention, we recommend the following changes:

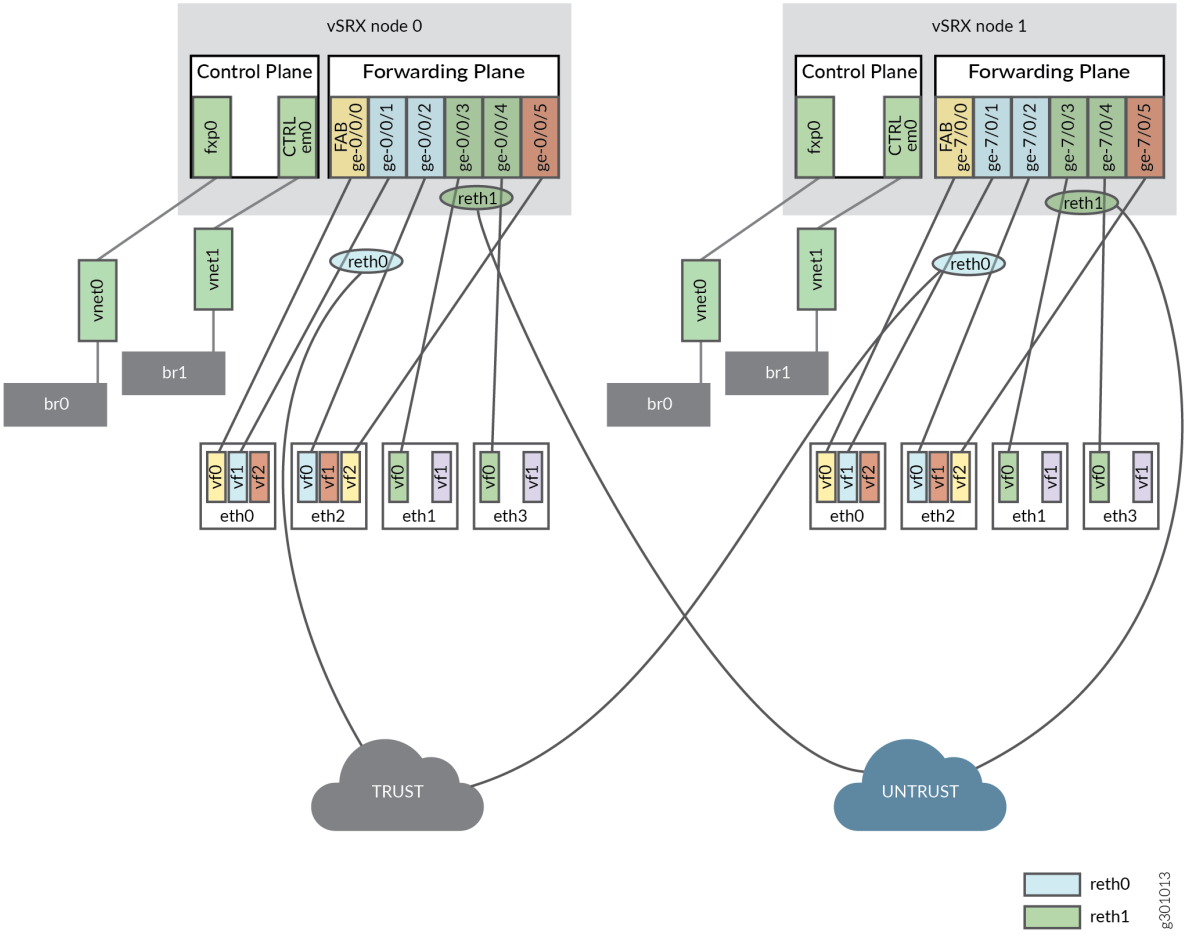
- Allot dedicated CPU to each vSRX 3.0 control plane, vSRX 3.0 data plane, and the host operating system.
- Allot required memory on the host.
- Leverage SR-IOV for fabric interface in a high availability deployment.
- Remove GRE for control link communication and use multicast in high availability deployments.
- Enable IRQ affinity to avoid the interrupts handled by the CPUs for vSRX 3.0 control plane and data plane.
- Enlarge the physical NIC descriptor from 512 to 4096 bytes.

We recommend you configure all revenue ports of vSRX 3.0 as SR-IOV. Also, on KVM you can configure SR-IOV high availability on management port -fxp0/ control port- em0 / fabric port-ge-0/0/*.

NOTE: SR-IOV high availability Layer 2 function is not supported. Also, VMware and Mellanox NIC do not support SR-IOV high availability functionality.

Figure 2 on page 7 shows the topology used in this example.

Figure 2: High Availability Trust and Untrust Dual NIC Topology



Configuration

IN THIS SECTION

- Verification | 19

SR-IOV High Availability Deployment

Step-by-Step Procedure

To configure the SR-IOV high availability deployment:

1. Enable the SR-IOV port.

```
#modprobe i40en max_vfs=8,8
or
echo 8 > /sys/class/net/ETH-X/device/sriov_numvfs
 echo 8 > /sys/class/net/ETH-Y/device/sriov_numvfs
```

   Enter the required inputs for availing ports.

```
8 = means set sriov_numvfs as 8 vfs
0 = For disable SRIOV port
```

2. Make the following changes in the grub file:

```
GRUB_CMDLINE_LINUX_DEFAULT="default_hugepagesz=1G hugepagesz=1G hugepages=18 iommu=pt
intel_iommu=on isolcpus=4-55 transparent_hugepage=never"
```

3. Execute upgrade grub.

```
update-grub
```

4. Reboot the host for changes to take effect.

```
reboot
```

5. (Optional) Cores 0-3 switch to interrupt context - Interrupt Service Routine (ISR) to handle the coming interrupt. Cores 4-13 on NUMA 0 are used for vSRXs. Run the following script:

```
cat irq.sh
#!/bin/bash
# Disable IRQ and set IRQ SMP affinity to core 0
```

```
disable_irq_balance_and_set_irq_affinity_core_0()
{
   echo f >  /proc/irq/default_smp_affinity
    #Disable_IRQ_Balance
    if [ -f /etc/init.d/irqbalance ]; then
        /etc/init.d/irqbalance stop
    fi
    #set_irq_affinity_core_0
    #for IRQ in `seq 0 512`;
    for IRQDIR in `ls -d /proc/irq/*`;
    do
        if [ -d $IRQDIR ]; then
            echo f > $IRQDIR/smp_affinity 2>/dev/null
            cat $IRQDIR/smp_affinity
        fi
    done
}
```

6. Increase tx and rx buffer size to 4096 on all NICs.

```
ethtool -G <ethx> rx 4096
      ethtool -G <ethx> tx 4096
```

7. Turn off flow control.

```
ethtool -A <ethx> autoneg off rx off tx off
```

8. Check for the server persistent after reboot.

```
cat /etc/rc.local
#!/bin/bash

echo 7 > /sys/class/net/eth0/device/sriov_numvfs
echo 7 > /sys/class/net/eth1/device/sriov_numvfs
echo 7 > /sys/class/net/eth2/device/sriov_numvfs
echo 7 > /sys/class/net/eth3/device/sriov_numvfs

/bin/irq.sh
```

9. Set SR-IOV VF trust mode on and spoof checking off.

```
# The Linux setting for SR-IOV VF Trust Mode:    --ip link set dev [PF] vf [VF_index] trust
off/on
 # The setting for SR-IOV VF spoof checking:    --ip link set dev [PF] vf [VF_index] spoof
checking on/off
```

Or, you can also add below command to rc.local script:

```
nic=eth0;for i in $(seq 0 15);do ip link set $nic vf $i spoofchk off trust on promisc on mtu
9000;done
nic=eth1;for i in $(seq 0 15);do ip link set $nic vf $i spoofchk off trust on promisc on mtu
9000;done
nic=eth2;for i in $(seq 0 15);do ip link set $nic vf $i spoofchk off trust on promisc on mtu
9000;done
nic=eth3;for i in $(seq 0 15);do ip link set $nic vf $i spoofchk off trust on promisc on mtu
9000;done
```

Build Bus Information of Virtual Functions on NICs

Step-by-Step Procedure

To build bus information of VFs on NICs:

1. Now that we know the backup interfaces, we need to identify the bus information of all VFs on each NIC.

   For backup interfaces in the trust network, we need bus information on the first three VFs.

```
# ls -l /sys/class/net/eth0/device/virtfn*

/sys/class/net/eth0/device/virtfn0 ->../0000:18:02.0
/sys/class/net/eth0/device/virtfn1 -> ../0000:18:02.1
/sys/class/net/eth0/device/virtfn2 -> ../0000:18:02.2
# ls -l /sys/class/net/eth2/device/virtfn*

/sys/class/net/eth2/device/virtfn0 ->../0000:18:0a.0
/sys/class/net/eth2/device/virtfn1 -> ../0000:18:0a.1
/sys/class/net/eth2/device/virtfn2 -> ../0000:18:0a.2
```

For backup interfaces in the untrust network, we need bus information on the first two VFs.

```
# ls -l /sys/class/net/eth1/device/virtfn*

/sys/class/net/eth1/device/virtfn0 ->../0000:18:06.0
/sys/class/net/eth1/device/virtfn1 -> ../0000:18:06.1
# ls -l /sys/class/net/eth1/device/virtfn*

/sys/class/net/eth3/device/virtfn0 ->../0000:18:0e.0
/sys/class/net/eth3/device/virtfn1 -> ../0000:18:0e.1
```

2. Table 2 on page 11 explains the XML to Junos interface-mapping required to build the template.

Table 2: XML to Junos Interfaces Mapping

NIC	VF	Bus Information	Interface		XML Position
			fxp0	fxp0	1
			em0	em0	2
eth0	0	0000:18:02.0	ge-0/0/0 fab0	ge-7/0/0 fab1	3
	1	0000:18:02.1	ge-0/0/1	ge-7/0/1	4
	2	0000:18:02.2	ge-0/0/5	ge-7/0/5	8
eth1	0	0000:18:06.0	ge-0/0/3	ge-7/0/3	6
eth2	0	0000:18:0a.0	ge-0/0/2	ge-7/0/2	5
eth3	0	0000:18:0e.0	ge-0/0/4	ge-7/0/4	7

The XML to Junos configuration is sequential. The first interface is assigned to fxp0 , second interface is assigned to em0 and the last interface is assigned to ge-0/0/9 as shown in Table 3 on page 12.

3. Develop the following Table 3 on page 12 based on Table 2 on page 11.

Table 3: Junos Interfaces and Bus Information

XML Position	BUS Information	Junos interfaces
1	BR0	fxp0
2	BR1	em0
3	0000:18:02.0	ge-0/0/0
4	0000:18:02.1	ge-0/0/1
5	0000:18:0a.0	ge-0/0/2
6	0000:18:06.0	ge-0/0/3
7	0000:18:0e.0	ge-0/0/4
8	0000:18:02.2	ge-0/0/5

4. Modify the interface stanza 2,3,4,8 and 12 in XML template below as per Table 3 on page 12.

```
<domain type='kvm'>
  <name>vm-name</name>
  <uuid>f5679184-a066-446b-a812-4fda2e9278dd</uuid>
  <memory unit='KiB'>8388608</memory>
  <currentMemory unit='KiB'>8388608</currentMemory>
  <memoryBacking>
    <hugepages/>
    <locked/>
  </memoryBacking>
  <vcpu placement='static' cpuset='4-9'>6</vcpu>
  <cputune>
    <vcpupin vcpu='0' cpuset='4'/>
    <vcpupin vcpu='1' cpuset='5'/>
    <vcpupin vcpu='2' cpuset='6'/>
```

```
    <vcpupin vcpu='3' cpuset='7'/>
    <vcpupin vcpu='4' cpuset='8'/>
    <vcpupin vcpu='5' cpuset='9'/>
  </cputune>
  <numatune>
    <memory mode='strict' nodeset='0'/>
  </numatune>
  <resource>
    <partition>/machine</partition>
  </resource>
  <os>
    <type arch='x86_64' machine='pc-i440fx-xenial'>hvm</type>
    <boot dev='hd'/>
  </os>
  <features>
    <acpi/>
    <apic/>
  </features>
  <cpu mode='host-passthrough' check='none'>
    <feature policy='require' name='pbe'/>
    <feature policy='require' name='tm2'/>
    <feature policy='require' name='est'/>
    <feature policy='require' name='vmx'/>
    <feature policy='require' name='aes'/>
    <feature policy='require' name='osxsave'/>
    <feature policy='require' name='smx'/>
    <feature policy='require' name='ss'/>
    <feature policy='require' name='ds'/>
    <feature policy='require' name='vme'/>
    <feature policy='require' name='dtes64'/>
    <feature policy='require' name='monitor'/>
    <feature policy='require' name='ht'/>
    <feature policy='force' name='dca'/>
    <feature policy='require' name='pcid'/>
    <feature policy='require' name='tm'/>
    <feature policy='require' name='pdcm'/>
    <feature policy='require' name='pdpe1gb'/>
    <feature policy='require' name='ds_cpl'/>
    <feature policy='require' name='xtpr'/>
    <feature policy='require' name='acpi'/>
    <feature policy='disable' name='invtsc'/>
  </cpu>
  <clock offset='utc'>
```

```
    <timer name='rtc' tickpolicy='catchup'/>
    <timer name='pit' tickpolicy='delay'/>
    <timer name='hpet' present='no'/>
  </clock>
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>restart</on_crash>
  <pm>
    <suspend-to-mem enabled='no'/>
    <suspend-to-disk enabled='no'/>
  </pm>
  <devices>
    <emulator>/usr/bin/kvm-spice</emulator>
    <disk type='file' device='disk'>
      <driver name='qemu' type='qcow2'/>
      <source file='/var/lib/libvirt/images/sriovvsrx/vSRX_Image.qcow2'/>
      <target dev='hda' bus='ide'/>
      <address type='drive' controller='0' bus='0' target='0' unit='0'/>
    </disk>
    <controller type='usb' index='0' model='ich9-ehci1'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x0c' function='0x7'/>
    </controller>
    <controller type='usb' index='0' model='ich9-uhci1'>
      <master startport='0'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x0c' function='0x0'
multifunction='on'/>
    </controller>
    <controller type='usb' index='0' model='ich9-uhci2'>
      <master startport='2'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x0c' function='0x1'/>
    </controller>
    <controller type='usb' index='0' model='ich9-uhci3'>
      <master startport='4'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x0c' function='0x2'/>
    </controller>
    <controller type='pci' index='0' model='pci-root'/>
    <controller type='ide' index='0'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0x1'/>
    </controller>
    <controller type='virtio-serial' index='0'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x0b' function='0x0'/>
    </controller>
    <interface type='bridge'>
```

```
      <mac address='2001:db8:00:46:05:b6'/>
      <source bridge='br0'/>
      <model type='virtio'/>
      <mtu size='9100'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
      <driver queues='8'/> # delete from existing templates
    </interface>
    <interface type='bridge'>
      <mac address='2001:db8:00:5e:c9:06'/>
      <source bridge='br1'/>
      <model type='virtio'/>
      <mtu size='9100'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x0'/>
    </interface>
   <interface type='hostdev' managed='yes'>
      <mac address='2001:db8:00:4e:f6:89'/>
      <driver name='vfio'/>
      <source>
        <address type='pci' domain='0x0000' bus='0x18' slot='0x02' function='0x0'/>
      </source>
<vlan>
<tag id='3681'/>
</vlan>
<address type='pci' domain='0x0000' bus='0x00' slot='0x05' function='0x0'/>
    </interface>
    <interface type='hostdev' managed='yes'>
      <mac address='2001:db8:00:4e:f5:f9'/>
      <driver name='vfio'/>
      <source>
        <address type='pci' domain='0x0000' bus='0x18' slot='0x02' function='0x1'/>
      </source>

<address type='pci' domain='0x0000' bus='0x18' slot='0x06' function='0x0'/>
    </interface>
    <interface type='hostdev' managed='yes'>
      <mac address='2001:db8:00:fa:b0:04'/>
      <driver name='vfio'/>
      <source>
        <address type='pci' domain='0x0000' bus='0x18' slot='0x0a' function='0x0'/>
      </source>

<address type='pci' domain='0x0000' bus='0x00' slot='0x07' function='0x0'/>
```

```
    </interface>
    <interface type='hostdev' managed='yes'>
      <mac address='2001:db8:00:da:87:b6'/>
      <driver name='vfio'/>
      <source>
        <address type='pci' domain='0x0000' bus='0x18' slot='0x06' function='0x0'/>
      </source>

<address type='pci' domain='0x0000' bus='0x00' slot='0x08' function='0x0'/>
    </interface>
    <interface type='hostdev' managed='yes'>
      <mac address='2001:db8:00:2e:e8:88'/>
      <driver name='vfio'/>
      <source>
        <address type='pci' domain='0x0000' bus='0x18' slot='0x0e' function='0x0'/>
      </source>

<address type='pci' domain='0x0000' bus='0x00' slot='0x09' function='0x0'/>
    </interface>
    <interface type='hostdev' managed='yes'>
      <mac address='2001:db8:00:6a:3c:f2'/>
      <driver name='vfio'/>
      <source>
        <address type='pci' domain='0x0000' bus='0x18' slot='0x02' function='0x2'/>
      </source>

<address type='pci' domain='0x0000' bus='0x00' slot='0x0a' function='0x0'/>
        <serial type='tcp'>
      <source mode='bind' host='192.0.2.1' service='8636' tls='no'/>
      <protocol type='telnet'/>
      <target type='isa-serial' port='0'>
        <model name='isa-serial'/>
      </target>
    </serial>
    <console type='tcp'>
      <source mode='bind' host='192.0.2.1' service='8636' tls='no'/>
      <protocol type='telnet'/>
      <target type='serial' port='0'/>
    </console>
    <channel type='spicevmc'>
      <target type='virtio' name='com.redhat.spice.0'/>
      <address type='virtio-serial' controller='0' bus='0' port='1'/>
    </channel>
```

```
    <input type='mouse' bus='ps2'/>
    <input type='keyboard' bus='ps2'/>
    <graphics type='spice' autoport='yes' listen='192.0.2.1'>
      <listen type='address' address='192.0.2.1'/>
      <image compression='off'/>
    </graphics>
    <sound model='ich6'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x0a' function='0x0'/>
    </sound>
    <video>
      <model type='qxl' ram='65536' vram='65536' vgamem='16384' heads='1' primary='yes'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x0'/>
    </video>
    <redirdev bus='usb' type='spicevmc'>
      <address type='usb' bus='0' port='1'/>
    </redirdev>
    <redirdev bus='usb' type='spicevmc'>
      <address type='usb' bus='0' port='2'/>
    </redirdev>
    <memballoon model='virtio'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x0d' function='0x0'/>
    </memballoon>
  </devices>
  <seclabel type='dynamic' model='apparmor' relabel='yes'/>
  <seclabel type='dynamic' model='dac' relabel='yes'/>
</domain>
```

Configure vSRX 3.0

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the `[edit]` hierarchy level, and then enter `commit` from configuration mode.

NOTE: ge-0/0/3, ge-0/0/4, ge-7/0/3, ge-7/0/4 are not used in this configuration.

```
set groups node0 system host-name host-name-node0
set groups node0 system backup-router 198.51.100.254
```

```
set groups node0 system backup-router destination 0.0.0.0/0
set groups node0 interfaces fxp0 unit 0 family inet address 198.51.100.248/20
set groups node1 system host-name host-name-node1
set groups node1 system backup-router 198.51.100.254
set groups node1 system backup-router destination 0.0.0.0/0
set groups node1 interfaces fxp0 unit 0 family inet address 198.51.100.249/20
set apply-groups "${node}"
set chassis cluster reth-count 2
set chassis cluster redundancy-group 0 node 0 priority 100
set chassis cluster redundancy-group 0 node 1 priority 1
set chassis cluster redundancy-group 1 node 0 priority 100
set chassis cluster redundancy-group 1 node 1 priority 1
set chassis cluster redundancy-group 2 node 0 priority 100
set chassis cluster redundancy-group 2 node 1 priority 1
set interfaces ge-0/0/1 gigether-options redundant-parent reth0
set interfaces ge-0/0/2 gigether-options redundant-parent reth1
set interfaces ge-7/0/1 gigether-options redundant-parent reth0
set interfaces ge-7/0/2 gigether-options redundant-parent reth1
set interfaces fab0 fabric-options member-interfaces ge-0/0/0
set interfaces fab1 fabric-options member-interfaces ge-7/0/0
set interfaces fab0 fabric-options member-interfaces ge-0/0/5
set interfaces fab1 fabric-options member-interfaces ge-7/0/5
set interfaces reth0 redundant-ether-options redundancy-group 1
set interfaces reth0 unit 0 family inet address 192.168.10.1/24
set interfaces reth1 redundant-ether-options redundancy-group 2
set interfaces reth1 unit 0 family inet address 192.168.11.1/24
set interfaces reth0 vlan-tagging
set interfaces reth0 unit 0 vlan-id 3601
set interfaces reth1 vlan-tagging
set interfaces reth1 unit 0 vlan-id 3602
set security zones security-zone TRUST host-inbound-traffic system-services all
set security zones security-zone TRUST host-inbound-traffic protocols all
set security zones security-zone TRUST interfaces reth0.0
set security zones security-zone UNTRUST host-inbound-traffic system-services all
set security zones security-zone UNTRUST host-inbound-traffic protocols all
set security zones security-zone UNTRUST interfaces reth1.0
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Chassis Cluster Status

Purpose

Verify the chassis cluster status, statistics, and redundancy group information.

Action

From operational mode, enter the following commands.

```
{primary:node0}
user@host> show chassis cluster interfaces

Control link status: Up

Control interfaces:
    Index   Interface   Monitored-Status   Internal-SA   Security
    0       em0         Up                 Disabled      Disabled

Fabric link status: Up

Fabric interfaces:
    Name    Child-interface    Status                    Security
                               (Physical/Monitored)
     fab0      ge-0/0/0           Up / Up                 Disabled
 fab0      ge-0/0/5           Up / Up                 Disabled
 fab1      ge-7/0/0           Up / Up                 Disabled
 fab1      ge-7/0/5           Up / Up                 Disabled
Redundant-ethernet Information:
    Name         Status      Redundancy-group
```

```
    reth0        Down        Not configured
    reth1        Up          1
    reth2        Up          2

Redundant-pseudo-interface Information:
    Name         Status      Redundancy-group
    lo0          Up          0
```

```
{primary:node0}
user@host> show chassis cluster statistics

Control link statistics:
    Control link 0:
        Heartbeat packets sent: 1797825
        Heartbeat packets received: 1797280
        Heartbeat packet errors: 0
Fabric link statistics:
    Child link 0
        Probes sent: 1329328
        Probes received: 1328840
    Child link 1
        Probes sent: 0
        Probes received: 0
Services Synchronized:
    Service name                            RTOs sent    RTOs received
    Translation context                     0            0
    Incoming NAT                            0            0
    Resource manager                        0            0
    DS-LITE create                          0            0
    Session create                          0            0
    IPv6 session create                     0            0
    Session close                           0            0
    IPv6 session close                      0            0
    Session change                          0            0
    IPv6 session change                     0            0
    ALG Support Library                     0            0
    Gate create                             0            0
    Session ageout refresh requests         0            0
    IPv6 session ageout refresh requests    0            0
    Session ageout refresh replies          0            0
    IPv6 session ageout refresh replies     0            0
```

```
    IPSec VPN                                0            0
    Firewall user authentication             0            0
    MGCP ALG                                 0            0
    H323 ALG                                 0            0
    SIP ALG                                  0            0
    SCCP ALG                                 0            0
    PPTP ALG                                 0            0
    JSF PPTP ALG                             0            0
    RPC ALG                                  0            0
    RTSP ALG                                 0            0
    RAS ALG                                  0            0
    MAC address learning                     0            0
    GPRS GTP                                 0            0
    GPRS SCTP                                0            0
    GPRS FRAMEWORK                           0            0
    JSF RTSP ALG                             0            0
    JSF SUNRPC MAP                           0            0
    JSF MSRPC MAP                            0            0
    DS-LITE delete                           0            0
    JSF SLB                                  0            0
    APPID                                    0            0
    JSF MGCP MAP                             0            0
    JSF H323 ALG                             0            0
    JSF RAS ALG                              0            0
    JSF SCCP MAP                             0            0
    JSF SIP MAP                              0            0
    PST_NAT_CREATE                           0            0
    PST_NAT_CLOSE                            0            0
    PST_NAT_UPDATE                           0            0
    JSF TCP STACK                            0            0
    JSF IKE ALG                              0            0
```

```
{primary:node0}
user@host> show chassis cluster control-plane statistics
Control link statistics:
    Control link 0:
        Heartbeat packets sent: 1797861
        Heartbeat packets received: 1797316
        Heartbeat packet errors: 0
Fabric link statistics:
    Child link 0
```

```
        Probes sent: 1329400
        Probes received: 1328912
    Child link 1
        Probes sent: 0
        Probes received: 0
```

```
{primary:node0}
user@host> show chassis cluster data-plane statistics
Services Synchronized:
    Service name                              RTOs sent    RTOs received
    Translation context                       0            0
    Incoming NAT                              0            0
    Resource manager                          0            0
    DS-LITE create                            0            0
    Session create                            0            0
    IPv6 session create                       0            0
    Session close                             0            0
    IPv6 session close                        0            0
    Session change                            0            0
    IPv6 session change                       0            0
    ALG Support Library                       0            0
    Gate create                               0            0
    Session ageout refresh requests           0            0
    IPv6 session ageout refresh requests      0            0
    Session ageout refresh replies            0            0
    IPv6 session ageout refresh replies       0            0
    IPSec VPN                                 0            0
    Firewall user authentication              0            0
    MGCP ALG                                  0            0
    H323 ALG                                  0            0
    SIP ALG                                   0            0
    SCCP ALG                                  0            0
    PPTP ALG                                  0            0
    JSF PPTP ALG                              0            0
    RPC ALG                                   0            0
    RTSP ALG                                  0            0
    RAS ALG                                   0            0
    MAC address learning                      0            0
    GPRS GTP                                  0            0
    GPRS SCTP                                 0            0
    GPRS FRAMEWORK                            0            0
```

```
    JSF RTSP ALG                              0            0
    JSF SUNRPC MAP                            0            0
    JSF MSRPC MAP                             0            0
    DS-LITE delete                            0            0
    JSF SLB                                   0            0
    APPID                                     0            0
    JSF MGCP MAP                              0            0
    JSF H323 ALG                              0            0
    JSF RAS ALG                               0            0
    JSF SCCP MAP                              0            0
    JSF SIP MAP                               0            0
    PST_NAT_CREATE                            0            0
    PST_NAT_CLOSE                             0            0
    PST_NAT_UPDATE                            0            0
    JSF TCP STACK                             0            0
    JSF IKE ALG                               0            0
```

```
{primary:node0}
user@host> show chassis cluster status redundancy-group 1
Monitor Failure codes:
    CS  Cold Sync monitoring        FL  Fabric Connection monitoring
    GR  GRES monitoring             HW  Hardware monitoring
    IF  Interface monitoring        IP  IP monitoring
    LB  Loopback monitoring         MB  Mbuf monitoring
    NH  Nexthop monitoring          NP  NPC monitoring
    SP  SPU monitoring              SM  Schedule monitoring
    CF  Config Sync monitoring      RE  Relinquish monitoring
    IS  IRQ storm

Cluster ID: 1
Node   Priority Status               Preempt Manual   Monitor-failures

Redundancy group: 1 , Failover count: 1
node0  200      primary              no      no       None
node1  1        secondary            no      no       None
```

Verification of deployment results

```
[user@host-kvm126 libvirt]#  virsh domiflist vm-name
Interface  Type       Source     Model       MAC
-------------------------------------------------------
```

```
vnet0    bridge      bro virtio 52:54:00:a5:6a:59
vnet1    bridge      br1 virtio 52:54:00:34:03:53
-        hostdev      -    -    52:54:00:ef:43:b6
-        hostdev      -    -    52:54:00:83:5f:e2
-        hostdev      -    -    52:54:00:99:85:ac
-        hostdev      -    -    52:54:00:f5:6b:30
-        hostdev      -    -    52:54:00:67:83:5f
-        hostdev      -    -    52:54:00:78:db:79
[user@host-kvm126 libvirt]# ip -d link show dev p2p2 |grep "vf 1 "
vf 1     link/ether 52:54:00:ef:43:b6 brd ff:ff:ff:ff:ff:ff, vlan 3681, spoof checking off, link-
state auto, trust on
[root@cnrd-kvm126 libvirt]# ip -d link show dev p2p3 |grep "vf 2 "
    vf 2     link/ether 52:54:00:83:5f:e2 brd ff:ff:ff:ff:ff:ff, spoof checking off, link-state
auto, trust on
[root@cnrd-kvm126 libvirt]# ip -d link show dev p2p3 |grep "vf 3 "
    vf 3     link/ether 52:54:00:99:85:ac brd ff:ff:ff:ff:ff:ff, spoof checking off, link-state
auto, trust on
[root@cnrd-kvm126 libvirt]#
```

Meaning

The sample output shows that there are no manual failover in chassis cluster status and provides you the spoof checking status and SR-IOV VF trust mode state.

Results

From configurational mode, confirm your configuration by entering the `show security zones`, and `show chassis` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show security zones
security-zone TRUST {
    host-inbound-traffic {
        system-services {
            all;
        }
        protocols {
            all;
        }
    }
```

```
    interfaces {
        reth0.0;
    }
}
security-zone UNTRUST {
    host-inbound-traffic {
        system-services {
            all;
        }
        protocols {
            all;
        }
    }
    interfaces {
        reth1.0;
    }
}
```

```
[edit]
user@host# show chassis
cluster {
    reth-count 3;
    redundancy-group 0 {
        node 0 priority 200;
        node 1 priority 1;
    }
    redundancy-group 1 {
        node 0 priority 200;
        node 1 priority 1;
    }
    redundancy-group 2 {
        node 0 priority 200;
        node 1 priority 1;
    }
}
```