JDM User Guide for NFX250 Network Services Platform

Modified: 2018-12-12

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

JDM User Guide for NFX250 Network Services Platform

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

List of Figures

List of Tables

About the Documentation

Documentation and Release Notes

To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at https://www.juniper.net/documentation/.

If the information in the latest release notes differs from the information in the documentation, follow the product Release Notes.

Juniper Networks Books publishes books by Juniper Networks engineers and subject matter experts. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration. The current list can be viewed at https://www.juniper.net/books.

Using the Examples in This Manual

If you want to use the examples in this manual, you can use the **load merge** or the **load merge relative** command. These commands cause the software to merge the incoming configuration into the current candidate configuration. The example does not become active until you commit the candidate configuration.

If the example configuration contains the top level of the hierarchy (or multiple hierarchies), the example is a *full example*. In this case, use the **load merge** command.

If the example configuration does not start at the top level of the hierarchy, the example is a *snippet*. In this case, use the **load merge relative** command. These procedures are described in the following sections.

Merging a Full Example

To merge a full example, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration example into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following configuration to a file and name the file **ex-script.conf**. Copy the **ex-script.conf** file to the **/var/tmp** directory on your routing platform.

   ```
   system {
     scripts {
       commit {
         file ex-script.xsl;
       }
     }
   }
   interfaces {
     fxp0 {
       disable;
       unit 0 {
         family inet {
           address 10.0.0.1/24;
         }
       }
     }
   }
   ```

2. Merge the contents of the file into your routing platform configuration by issuing the **load merge** configuration mode command:

   ```
   [edit]
   user@host# load merge /var/tmp/ex-script.conf
   load complete
   ```

Merging a Snippet

To merge a snippet, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration snippet into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following snippet to a file and name the file **ex-script-snippet.conf**. Copy the **ex-script-snippet.conf** file to the **/var/tmp** directory on your routing platform.

   ```
   commit {
     file ex-script-snippet.xsl; }
   ```

2. Move to the hierarchy level that is relevant for this snippet by issuing the following configuration mode command:

```
[edit]
user@host# edit system scripts
[edit system scripts]
```

3. Merge the contents of the file into your routing platform configuration by issuing the **load merge relative** configuration mode command:

```
[edit system scripts]
user@host# load merge relative /var/tmp/ex-script-snippet.conf
load complete
```

For more information about the **load** command, see CLI Explorer.

Documentation Conventions

Table 1 on page xv defines notice icons used in this guide.

Table 1: Notice Icons

Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.
	Tip	Indicates helpful information.
	Best practice	Alerts you to a recommended use or implementation.

Table 2 on page xvi defines the text and syntax conventions used in this guide.

Table 2: Text and Syntax Conventions

Convention	Description	Examples
Bold text like this	Represents text that you type.	To enter configuration mode, type the **configure** command: user@host> **configure**
`Fixed-width text like this`	Represents output that appears on the terminal screen.	`user@host>` **show chassis alarms** `No alarms currently active`
Italic text like this	• Introduces or emphasizes important new terms. • Identifies guide names. • Identifies RFC and Internet draft titles.	• A policy *term* is a named structure that defines match conditions and actions. • *Junos OS CLI User Guide* • RFC 1997, *BGP Communities Attribute*
Italic text like this	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name: [edit] root@# **set system domain-name** *domain-name*
Text like this	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	• To configure a stub area, include the **stub** statement at the **[edit protocols ospf area area-id]** hierarchy level. • The console port is labeled **CONSOLE**.
< > (angle brackets)	Encloses optional keywords or variables.	**stub <default-metric** ***metric*****>;**
| (pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	**broadcast | multicast** **(*string1* | *string2* | *string3*)**
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	**rsvp { # Required for dynamic MPLS only**
[ ] (square brackets)	Encloses a variable for which you can substitute one or more values.	**community name members [** ***community-ids*** **]**

Indention and braces ({ })	Identifies a level in the configuration hierarchy.	[edit]
		routing-options {
		  static {
		  route default {
		  nexthop *address*;
		  retain;
		  }
		  }
		}

; (semicolon)	Identifies a leaf statement at a configuration hierarchy level.	

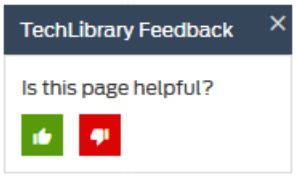
Table 2: Text and Syntax Conventions (continued)

Convention	Description	Examples
GUI Conventions		
Bold text like this	Represents graphical user interface (GUI) items you click or select.	• In the Logical Interfaces box, select **All Interfaces**. • To cancel the configuration, click **Cancel**.
> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select **Protocols>Ospf**.

Documentation Feedback

We encourage you to provide feedback so that we can improve our documentation. You can use either of the following methods:

- Online feedback system—Click TechLibrary Feedback, on the lower right of any page on the Juniper Networks TechLibrary site, and do one of the following:



  - Click the thumbs-up icon if the information on the page was helpful to you.
  - Click the thumbs-down icon if the information on the page was not helpful to you or if you have suggestions for improvement, and use the pop-up form to provide feedback.

- E-mail—Send your comments to **techpubs-comments@juniper.net**. Include the document or topic name, URL or page number, and software version (if applicable).

Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active J-Care or Partner Support Service support contract, or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at https://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf.
- Product warranties—For product warranty information, visit https://www.juniper.net/support/warranty/.

- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: https://www.juniper.net/customers/support/
- Search for known bugs: https://prsearch.juniper.net/
- Find product documentation: https://www.juniper.net/documentation/
- Find solutions and answer questions using our Knowledge Base: https://kb.juniper.net/
- Download the latest versions of software and review release notes: https://www.juniper.net/customers/csc/software/
- Search technical bulletins for relevant hardware and software notifications: https://kb.juniper.net/InfoCenter/
- Join and participate in the Juniper Networks Community Forum: https://www.juniper.net/company/communities/
- Open a case online in the CSC Case Management tool: https://www.juniper.net/cm/

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: https://entitlementsearch.juniper.net/entitlementsearch/

Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at https://www.juniper.net/cm/.
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see https://www.juniper.net/support/requesting-support.html.

PART 1

Architecture Overview

CHAPTER 1

Architecture Overview

Understanding Disaggregated Junos OS

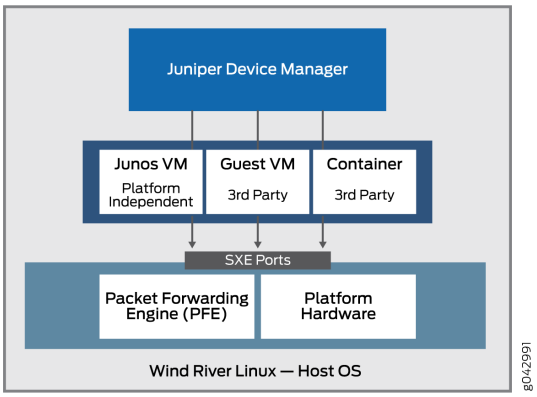
Many network equipment vendors have traditionally bound their software to purpose-built hardware and sold customers the bundled and packaged software–hardware combination. However, with the disaggregated Junos OS architecture, Juniper Network devices are now aligned with networks that are cloud-oriented, open, and rely on more flexible implementation scenarios.

The basic principle of the disaggregated Junos OS architecture is decomposition (*disaggregation*) of the tightly bound Junos OS software and proprietary hardware into virtualized components that can potentially run not only on Juniper Networks hardware, but also, on white boxes or bare-metal servers. In this new architecture, the Juniper Device Manager (JDM) is a virtualized root container that manages software components.

The JDM is the only root container in the disaggregated Junos OS architecture (there are other industry models that allow more than one root container, but the disaggregated Junos OS architecture is not one of them). The disaggregated Junos OS is a *single-root* model. One of the major functions of JDM is to prevent modifications and activities on the platform from impacting the underlying host OS (usually Linux). As the root entity, the JDM is well-suited for that task. The other major function of JDM is to make the hardware of the device look as much like a traditional Junos OS–based physical system as possible. This also requires some form of root capabilities.

Figure 1 on page 4 illustrates the important position JDM occupies in the overall architecture.

Figure 1: Position of the Juniper Device Manager



A VNF is a consolidated offering that contains all the components required for supporting a fully virtualized networking environment. A VNF has network optimization as its focus.

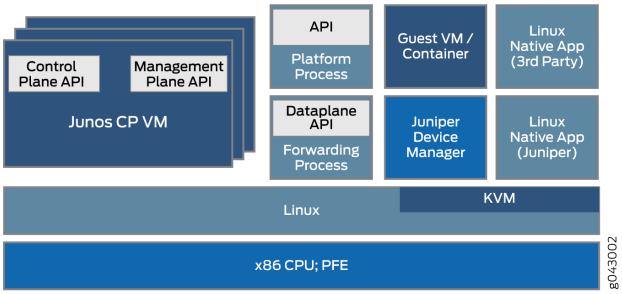
JDM enables:

- Management of guest virtualized network functions (VNFs) during their life cycle.
- Installation of third-party modules.
- Formation of VNF service chains.
- Management of guest VNF images (their binary files).
- Control of the system inventory and resource usage.

Note that some implementations of the basic architecture include a Packet Forwarding Engine as well as the usual Linux platform hardware ports. This allows better integration of the Juniper Networks data plane with the bare-metal hardware of a generic platform.

The disaggregated Junos OS architecture enables JDM to handle virtualized network functions such as a firewall or Network Address Translation (NAT) functions. The other VNFs and containers integrated with JDM can be Juniper Networks products or third-party products as native Linux applications. The basic architecture of the disaggregated Junos OS is shown schematically in Figure 2 on page 4.

Figure 2: Basic Disaggregated Junos OS Architecture



NOTE: There are multiple ways to implement the basic disaggregated Junos OS architecture on various platforms. Details can vary greatly. This topic describes the overall architecture.

The virtualization of the simple software process running on fixed hardware poses several challenges in the area of interprocess communication. How does, for example, a VNF with a NAT function work with a firewall running as a container on the same device? After all, there might be only one or two external Ethernet ports on the whole device, and the processes are still internal to the device. One benefit is the fact that the interfaces between these virtualized processes are often virtualized themselves, perhaps as *SXE ports*; which means that you can configure a type of MAC-layer bridge between processes directly, or between a process and the host OS and then between the host OS and another process. This supports the chaining of services as traffic enters and exits the device.

JDM provides users with a familiar Junos OS CLI and handles all interactions with underlying Linux kernel to maintain the “look and feel” of a Juniper Networks device.

Some of the benefits of the disaggregated Junos OS are:

- The whole system can be managed like managing a server platform.
- Customers can install third-party applications, tools, and services, such as Chef, Wiireshark, or Quagga, in a virtual machine (VM) or container.
- These applications and tools can be upgraded by using typical Linux repositories and are independent of Junos OS releases.
- Modularity increases reliability because faults are contained within the module.
- The control and data planes can be programmed directly through APIs.

Related Documentation

- Disaggregated Junos OS VMs on page 6
- Understanding Physical and Virtual Components on page 12
- Understanding Virtio Usage on page 8
- Understanding SR-IOV Usage on page 10
- Comparing Virtio and SR-IOV on page 11

Disaggregated Junos OS VMs

Cloud computing enables applications to run in a virtualized environment, both for end-user server functions and network functions needed to connect scattered endpoints across a large data center, or even among multiple data centers. Applications and network functions can be implemented by virtualized network functions (VNFs). What are the differences between these two types of packages and why would someone use one type or the other?

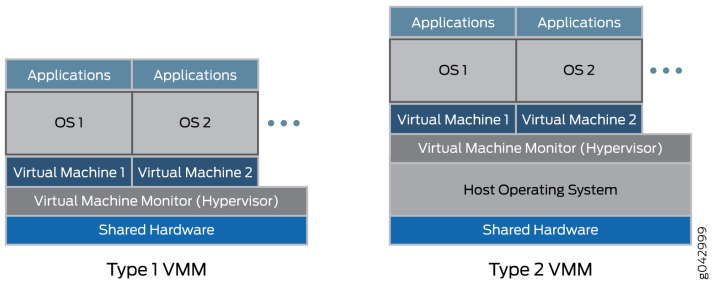
Both VNFs and containers allow the multiplexing of hardware with tens or hundreds of VNFs sharing one physical server. This allows not only rapid deployment of new services, but also extension and migration of workloads at times of heavy use (when extension can be used) or physical maintenance (when migration can be used).

In a cloud computing environment, it is common to employ VNFs to do the heavy work on the massive server farms that characterize big data in modern networks. Server virtualization allows applications written for different development environments, hardware platforms, or operating systems to run on generic hardware that runs an appropriate software suite.

VNFs rely on a hypervisor to manage the physical environment and allocate resources among the VNFs running at any particular time. Popular hypervisors include Zen, KVM, and VMWare ESXi, but there are many others. The VNFs run in the user space on top of the hypervisor and include a full implementation of the VM application's operating system. For example, an application written in the C++ language and complied and run on Microsoft Windows operating system can be run on a Linux operating system using the hypervisor. In this case, Windows is a guest operating system.

There are two types of virtual machine monitors (VMMs) in use. In a Type 1 VMM, the hypervisor is placed directly on top of the shared hardware. In a Type 2 VMM, the hypervisor is placed on top of the host OS. In both cases, the VNFs still use the hypervisor, but in some contexts the differences are significant. The two types of VMMs are shown in Figure 3 on page 6.

Figure 3: Virtual Machine Monitors



The hypervisor provides the guest operating system with an emulated view of the hardware of the VNFs. Among other resources such as disk space of memory, the hypervisor provides a virtualized view of the network interface card (NIC) when endpoints

for different VMs reside on different servers or hosts (a common situation). The hypervisor manages the physical NICs and exposes only virtualized interfaces to the VNFs.

The hypervisor also runs a virtual switch environment, which allows the VNFs at the VLAN frame layer to exchange packets inside the same box, or over a (virtual) network.

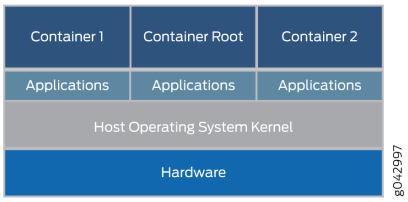
The biggest advantage of VNFs is that most applications can be easily ported to the hypervisor environment and run well without modification.

The biggest drawback is that, often the resource-intense overhead of the guest operating system must include a complete version of the operating system even if the function of the entire VNF is to provide a simple service such as a domain name system (DNS).

Containers, unlike VNFs, are purpose-built to be run as independent tasks in a virtual environment. Containers do not bundle an entire operating system inside like VNFs do. Containers can be coded and bundled in many ways, but there are also ways to build standard containers that are easy to maintain and extend. Standard containers are much more open than containers created in a haphazard fashion.

Standard Linux containers define a unit of software delivery called a standard container. Instead of encapsulating the whole guest operating system, the standard container encapsulates only the application and any dependencies required to perform the task the application is programmed to perform. This single runtime element can be modified, but then the container must be rebuilt to include any additional dependencies that the extended function might need. The overall architecture of containers in shown in Figure 4 on page 7.

Figure 4: Containers–Overall Architecture



The containers run on the host OS kernel and not on the hypervisor. The container architecture uses a container engine to manage the underlying platform. If you still want to run VNFs, the container can package up a complete hypervisor and guest OS environment as well.

Standard containers include:

- A configuration file.
- A set of standard operations.
- An execution environment.

The name *container* is borrowed from the shipping containers that are used to transport goods around the world. Shipping containers are standard delivery units that can be loaded, labelled, stacked, lifted, and unloaded by equipment built specifically to handle the containers. No matter what is inside, the container can be handled in a standard

fashion, and each container has its own user space that cannot be used by other containers. Although Docker is a popular container management system to run containers on a physical server, there are alternatives such as Drawbridge or Rocket to consider.

Each container is assigned a virtual interface. Container management systems such as Docker include a virtual Ethernet bridge connecting multiple virtual interfaces and the physical NIC. Configuration and environment variables in the container determine which containers can communicate with each other, which can use the external network, and so on. External networking is usually accomplished with NAT although there are other methods because, containers often use the same network address space.

The biggest advantage of containers is that they can be loaded on a device and executed much faster than VNFs. Containers also use resources much more sparingly— you can run many more containers than VNFs on the same hardware. This is because containers do not require a full guest operating system or boot time. Containers can be loaded and run in milliseconds, not tens of seconds. However, the biggest drawback with containers is that they have to be written specifically to conform to some standard or common implementation, whereas VNFs can be run in their native state.

Related Documentation

- Understanding Disaggregated Junos OS on page 3
- Understanding Physical and Virtual Components on page 12
- Understanding Virtio Usage on page 8
- Understanding SR-IOV Usage on page 10
- Comparing Virtio and SR-IOV on page 11

Understanding Virtio Usage

You can enable communication between a Linux-based virtualized device and a virtualized network function (VNF) module by bridging the two using a library called virtio.

When a physical device is virtualized, both physical NIC interfaces and external physical switches as well as the virtual NIC interfaces and internal virtual switches coexist. So when the isolated VNFs in the device, each with their own memory and disk space and CPU cycles, attempt to communicate with each other, the multiple ports, MAC addresses, and IP addresses in use pose a challenge. With the virtio library, traffic flow between the isolated virtual functions becomes simpler and easier.

Virtio is part of the standard Linux libvirt library of useful virtualization functions and is normally included in most versions of Linux. Virtio is a software-only approach to inter-VNF communication. Virtio provides a way to connect individual virtual processes. The bundled nature of virtio makes it possible for any Linux-run device to use virtio.

Virtio enables VNFs and containers to use simple internal bridges to send and receive traffic. Traffic can still arrive and leave through an external bridge. An external bridge uses a virtualized internal NIC interface on one end of the bridge and a physical external NIC interface on the other end of the bridge to send and receive packets and frames. An internal bridge, of which there are several types, links two virtualized internal NIC interfaces

by bridging them through a virtualized internal switch function in the host OS. The overall architecture of virtio is shown in Figure 5 on page 9.

Figure 5: VNF Bridging with Virtio

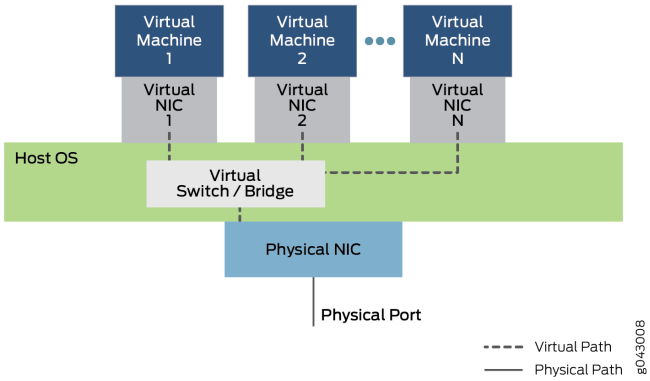


Figure 5 on page 9 shows the inner structure of a server device with a single physical NIC card running a host OS (the outer cover of the device is not shown). The host OS contains the virtual switch or bridge implemented with virtio. Above the OS, several virtual machines employ virtual NICs that communicate through virtio. There are multiple virtual machines running, numbered 1 to N in the figure. The standard “dot dot dot” notation indicates possible virtual machines and NICs not shown in the figure. The dotted lines indicate possible data paths using virtio. Note that traffic entering or leaving the device does so through the physical NIC and port.

Figure 5 on page 9 also shows traffic entering and leaving the device through the internal bridge. Virtual Machine 1 links its virtualized internal NIC interface to the physical external NIC interface. Virtual Machine 2 and Virtual Machine N link internal virtual NICs through the internal bridge in the host OS. Note that these interface might have VLAN labels associated with them, or internal interface names. Frames sent across this internal bridge between VNFs never leave the device. Note the position of the bridge (and virtualized switch function) in the host OS. Note the use of simple bridging in the device. These bridges can be configured either with *regular* Linux commands or the use of CLI configuration statements. Scripts can be used to automate the process.

Virtio is a virtualization standard for disk and network device drivers. Only the guest device driver (the devices driver for the virtualized functions) needs to *know* that it is running in a virtual environment. These drivers cooperate with the hypervisor and the virtual functions get performance benefits in return for the added complication. Virtio is architecturally similar to, but not the same as, Xen paravirtualized device drivers (drivers added to a guest to make them faster when running on Xen). VMWare’s Guest Tools are also similar to virtio.

Note that much of the traffic is concentrated on the host OS CPU—more explicitly, on the virtualized internal bridges. Therefore, the host CPU must be able to perform adequately as the device scales.

Related Documentation

- Understanding Disaggregated Junos OS on page 3
- Understanding Physical and Virtual Components on page 12
- Disaggregated Junos OS VMs on page 6
- Understanding SR-IOV Usage on page 10
- Comparing Virtio and SR-IOV on page 11

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 single-root I/O virtualization (SR-IOV).

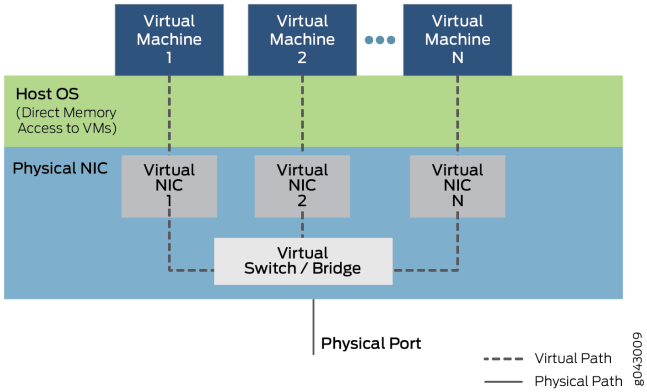
When a physical device is virtualized, both physical network interface card (NIC) interfaces and external physical switches as well as the virtual NIC interfaces and internal virtual switches coexist. So when the isolated virtual machines (VMs) or containers in the device, each with their own memory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

SR-IOV extends the concept of virtualized functions right down to the physical *NIC* . The single physical card is divided into up to 16 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 communications between devices with individual *NIC* are usually handled: with a bridge. SR-IOV includes a set of standard methods for creating, deleting, enumerating, and querying the SR-IOV NIC switch, as well as the standard parameters than can be set.

The *single-root* part of SR-IOV refers to the fact that there is really only one *primary* piece of the *NIC* controlling all operations. An SR-IOV-enabled *NIC* is just a standard Ethernet port providing the same physical *bit-by-bit function* of any network card.

However, the SR-IOV also provides several virtual functions, which are accomplished by simple queues to handle input and output tasks. Each VNF running on the device is mapped to one of these NIC partitions so that VNFs themselves have direct access to NIC hardware resources. The NIC also 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 6 on page 11.

Figure 6: VNF Communication Using SR-IOV



The hypervisor is still involved in the assignment of the VNFs to the virtual network functions, and in the management of the physical card, but not in the transfer of the data inside the packets. Note that VNF-to-VNF communication is performed by Virtual NIC 1, Virtual NIC 2, and Virtual NIC N. There is also a portion of the NIC (not shown) that keeps track of all the virtual functions and the sorter to shuttle traffic among the VNFs and external device ports.

Note that the ability to support SR-IOV is dependent on the platform hardware, specifically the NIC hardware, and the software of the VNFs or containers to employ DMA for data transfer. Partitionable NICs, and the internal bridging required, tend to be more expensive, because of which, their use can increase the cost on smaller devices by an appreciable amount. Rewriting VNFs and containers is not a trivial task either.

Related Documentation

- Understanding Disaggregated Junos OS on page 3
- Understanding Physical and Virtual Components on page 12
- Disaggregated Junos OS VMs on page 6
- Understanding Virtio Usage on page 8
- Comparing Virtio and SR-IOV on page 11

Comparing Virtio and SR-IOV

You can enable communication between a Linux-based virtualized device and a Network Functions Virtualization (NFV) module either by using virtio or by using suitable hardware and single-root I/O virtualization (SR-IOV). Each method has distinct characteristics.

Virtio is part of the standard libvirt library of helpful virtualization functions and is normally included in most versions of Linux. Virtio adopts a software-only approach. SR-IOV requires software written in a certain way and specialized hardware, which means an increase in cost, even with a simple device.

Generally, using virtio is quick and easy. Libvirt is part of every Linux distribution and the commands to establish the bridges are well-understood. However, virtio places all of

the burden of performance on the host OS, which normally bridges all the traffic between VNFs, into and out of the device.

Generally, SR-IOV can provide lower latency and lower CPU utilization—in short, almost native, non-virtual device performance. But VNF migration from one device to another is complex because the VNF is dependent on the NIC resources on one machine. Also, the forwarding state for the VNF resides in the Layer 2 switch built into the SR-IOV NIC. Because of this, forwarding is no longer quite as flexible because the rules for forwarding are coded into the hardware and cannot be changed often.

While support for virtio is nearly universal, support for SR-IOV varies by NIC hardware and platform. The Juniper Networks NFX250 Network Services Platform supports SR-IOV capabilities and allows 16 partitions on each physical NIC port.

Note that a given VNF can use either virtio or SR-IOV, or even both methods simultaneously, if supported.

NOTE: **Virtio is the recommended method for establishing connection between a virtualized device and an NFV module.**

Related Documentation

- Understanding Disaggregated Junos OS on page 3
- Understanding Physical and Virtual Components on page 12
- Disaggregated Junos OS VMs on page 6
- Understanding Virtio Usage on page 8
- Understanding SR-IOV Usage on page 10

Understanding Physical and Virtual Components

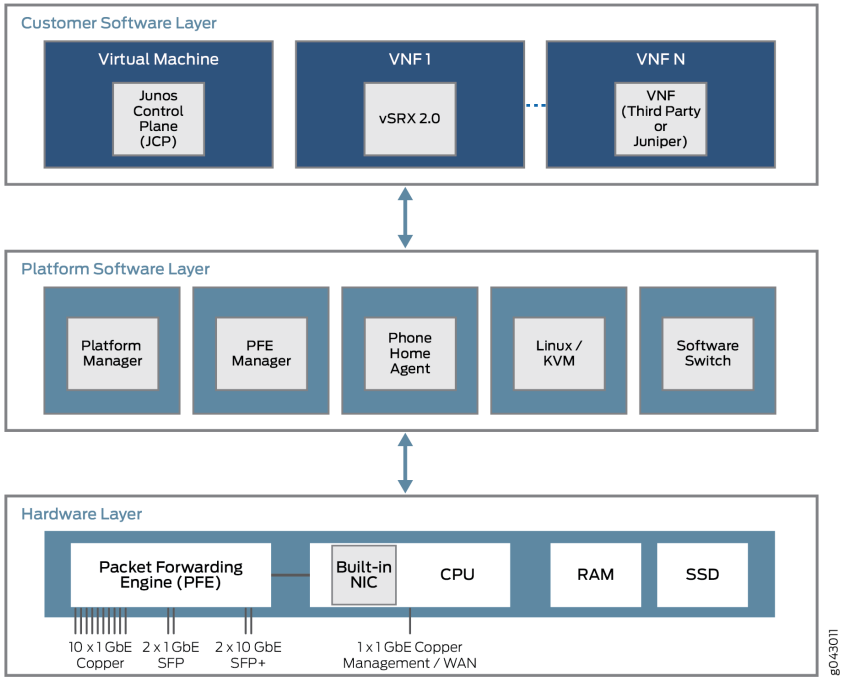
In the disaggregated Junos OS Network Functions Virtualization (NFV) environment, device components might be physical or virtual. The same physical-virtual distinction can be applied to interfaces (ports), the paths that packets or frames take through the device, and other aspects such as CPU cores or disk space.

The disaggregated Junos OS specification includes an architectural model. The architectural model of a house can have directions for including a kitchen, a roof, and a dining room, and can represent various kinds of dwellings; from a seaside cottage to a palatial mansion. All these houses look very different, but still follow a basic architectural model and share many characteristics.

Similarly, in the case of the disaggregated Junos OS architectural models, the models cover vastly different types of platforms, from simple customer premises equipment (CPE) to complex switching equipment installed in a large data center, but have some basic characteristics that the platforms share.

What characteristics do these platforms share? All disaggregated Junos OS platforms are built on three layers. These layers and some possible content are shown in Figure 7 on page 13.

Figure 7: Physical and Virtual Layers in the Disaggregated Junos OS



The lowest layer is the hardware layer. In addition to memory (RAM) and disk space (SSD), the platform hardware has a multi-core CPU with an external NIC port used for management. In some cases, there will be a single NIC port used for the control and data plane, but that port can also be used to communicate with a Packet Forwarding Engine for user traffic streams.

The platform software layer sits on top of the hardware layer. All platform-dependent functions take place here. These functions can include a software switching function for various virtual components to bridge traffic between them. A Linux or kernel-based virtual machine (KVM) runs the platform, and, in some models, a phone home agent contacts a vendor or service provider device to perform autoconfiguration tasks. The phone home agent is particularly preferred for smaller CPE platforms.

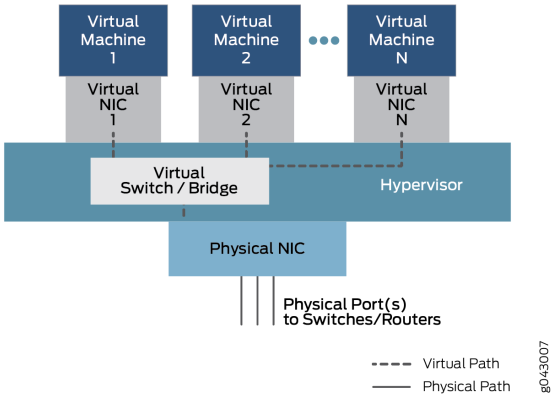
Above the platform software layer is the customer software layer, which perfoms various platform-independent functions. Some of the components might be Juniper Networks virtual machines, such as a virtual SRX device (vSRX) or the Junos Control Plane (JCP). The JCP works with the JDM to make the device resemble a dedicated Juniper Networks platform, but one with a lot more flexibility. Much of this flexibility comes from the ability to support one or more VNFs that implement a virtualized network function (VNF). These VNFs consist of many types of tasks, such as Network Address Translation (NAT),specialized Domain Name System (DNS) server lookups, and so on.

Generally, there are a fixed number of CPU cores, and a finite amount of disk space. But in a virtual environment, resource allocation and use is more complex. Virtual resources such as interfaces, disk space, memory, or cores are parceled out among the VNFs running at the time, as determined by the VNF image.

The VNFs, whether virtual machines (VMs) or containers, which share the physical device are often required to communicate with each other. Packets or frames enter a device through a physical interface (a port) and are distributed to some initial VNF. After some processing of the traffic flow, the VNF passes the traffic over to another VNF if configured to do so, and then to another, before the traffic leaves the physical device. These VNFs form a data plane service chain that is traversed inside the device.

How do the VNFs, which are isolated VMs or containers, pass traffic from one to the other? The service chain is configured to pass traffic from a physical interface to one or more internal virtual interfaces. Therefore, there are virtual NICs associated with each VM or Container, all connected by a virtual switch or bridge function inside the device. This generic relationship, which enables communication between physical and virtual interfaces is shown in Figure 8 on page 14.

Figure 8: Physical and Virtual Component Communication



In this general model, which can have variations in different platforms, data enters through a port on the physical NIC and is bridged through the virtual switch function to Virtual Machine1 through Virtual NIC 1, based on destination MAC address. The traffic can also be bridged through another configured virtual interface to Virtual Machine2 or more VNFs until it is passed back to a physical port and exits the device.

For configuration purposes, these interfaces might have familiar designations such as ge-0/0/0 or fxp0, or new designations such as sxe0 or hsxe0. Some might be *real*, but internal ports (such as sxe0), and some might be completely virtual constructs (such as hsxe0) needed to make the device operational.

Related Documentation

- Understanding Disaggregated Junos OS on page 3
- Disaggregated Junos OS VMs on page 6
- Understanding Virtio Usage on page 8

PART 2

Installation

CHAPTER 2

Installation

Managing Software Installation on NFX250 Network Services Platform

This topic lists the commands to be used for installing a software package and upgrading an image on NFX250 Network Services Platform and rebooting the NFX250 platform. It also lists the commands to be used for formatting and reverting the system to factory state.

To install a new package on the NFX250 Network Services Platform:

```
[edit]
user@jdm> request system software add package [reboot]
```

Reboot is an option to reboot the device after installing the new software package.

Replace ***package*** with the following path:

For a software package in a local directory on the platform—**/var/tmp/package.tgz**

To reboot the platform:

```
[edit]
user@jdm> request system reboot
```

To format the system by deleting all user data, configuration details, and reinstall current software on the NFX250 Network Services platform:

```
user@jdm> request system zeroize
```

To format the system by deleting all user data, configuration details, and to upgrade the software on the NFX250 Network Services platform:

```
user@jdm> request system software add package clean-install
```

NOTE: The zeroize and clean-install commands work only for primary installation and do not work for backup installation.

CAUTION: The zeroize and clean-install commands might remove all user installed software packages, VNF files of the user, and so on. After completing these operations, you must fetch these information and reinstall the software. You might require a console access to configure the basic remote network access if the system is in factory state.

Related Documentation

- Understanding the JDM CLI on page 32
- Accessing the JDM Shell, JDM CLI, and JCP Prompts in a Disaggregated Junos OS Platform on page 32

Upgrading an Image on the Disaggregated Junos OS Platform

- To upgrade the images of JCP, JDM, and the host OS on the disaggregated Junos OS platform:

```
user@jdm>request system software add jinstall reboot
```

For example:

```
user@jdm>request system software add
  /var/tmp/jinstall-nfx-2-flex-15.1X53-D40.3.secure-domestic-signed.tgz reboot |
  no-more
  System software upgrade in progress, please wait...
  Pushing Junos image package to the host...
  Installing /var/tmp/install-media-nfx-2-junos-15.1X53-D45.3.secure.tgz
  Extracting the package ...
  total 1191772
  -rw-r--r-- 1 20607 758 313873261 Nov 22 09:18
  jinstall-nfx-2-junos-15.1X53-D45.3.secure-linux.tgz
  -rw-r--r-- 1 20607 758 906487459 Nov 22 09:18
  jinstall-nfx-2-junos-15.1X53-D45.3.secure-app.tgz

  ============================================
  Host OS upgrade is FORCED
  Current Host kernel version : 3.14.61-rt58-WR7.0.0.13_ovp
  Package Host kernel version : 3.14.61-rt58-WR7.0.0.13_ovp
  Current Host version        : 3.0.2
  Package Host version        : 3.0.2
  Min host version required for applications: 2.2.0
  ============================================

  Validate linux image...
  upgrade_platform: -------------------
```

```
upgrade_platform: Parameters passed:
upgrade_platform: silent=0
upgrade_platform:
package=/var/tmp/tmp.LKb5WwiFu8junos_cli_upg/jinstall-nfx-2-junos-15.1X53-D45.3.secure-linux.tgz
upgrade_platform: clean install=0
upgrade_platform: on primary   =0
upgrade_platform: clean upgrade=0
upgrade_platform: Need reboot after staging=1
upgrade_platform: -------------------
upgrade_platform:
upgrade_platform: Checking input
/var/tmp/tmp.LKb5WwiFu8junos_cli_upg/jinstall-nfx-2-junos-15.1X53-D45.3.secure-linux.tgz
 ...
upgrade_platform: Input package
/var/tmp/tmp.LKb5WwiFu8junos_cli_upg/jinstall-nfx-2-junos-15.1X53-D45.3.secure-linux.tgz
 is valid.
Secure Boot is enforced.
ALLOW:usr/secureboot/grub/BOOTX64.EFI
ALLOW:boot/bzImage-intel-x86-64.bin
ALLOW:boot/initramfs.cpio.gz
Setting up Junos host applications for installation ...
Installing Host OS ...
upgrade_platform: -------------------
upgrade_platform: Parameters passed:
upgrade_platform: silent=0
upgrade_platform:
package=/var/tmp/jinstall-nfx-2-junos-15.1X53-D45.3.secure-linux.tgz
upgrade_platform: clean install=0
upgrade_platform: on primary   =0
upgrade_platform: clean upgrade=0
upgrade_platform: Need reboot after staging=0
upgrade_platform: -------------------
upgrade_platform:
upgrade_platform: Checking input
/var/tmp/jinstall-nfx-2-junos-15.1X53-D45.3.secure-linux.tgz ...
upgrade_platform: Input package
/var/tmp/jinstall-nfx-2-junos-15.1X53-D45.3.secure-linux.tgz is valid.
Secure Boot is enforced.
ALLOW:usr/secureboot/grub/BOOTX64.EFI
ALLOW:boot/bzImage-intel-x86-64.bin
ALLOW:boot/initramfs.cpio.gz
upgrade_platform: Staging the upgrade package -
/var/tmp/jinstall-nfx-2-junos-15.1X53-D45.3.secure-linux.tgz..
./
./bzImage-intel-x86-64.bin
./bzImage-intel-x86-64.bin.psig
./grub/
./grub/grub.conf
./grub/grub.efi
./initramfs.cpio.gz
./initramfs.cpio.gz.psig
./linux.checksum
./version.txt
./upgrade_platform
./host-version
./platform_info
./initrd.cpio.gz
bzImage-intel-x86-64.bin: OK
initramfs.cpio.gz: OK
version.txt: OK
```

```
upgrade_platform: Checksum verified and OK...
1528703 blocks
upgrade_platform: Staging of
/var/tmp/jinstall-nfx-2-junos-15.1X53-D45.3.secure-linux.tgz completed
upgrade_platform: System need *REBOOT* to complete the upgrade
upgrade_platform: Run upgrade_platform with option -r | --rollback to rollback
 the upgrade

Host OS upgrade staged. Reboot the system to complete installation!

Rebooting ...
System going down for reboot in 30 seconds...
System reboot in progress...
Shutting down virtual-machines...
Waiting for virtual-machines to shutdown, retry = 0
Waiting for virtual-machines to shutdown, retry = 1
Waiting for virtual-machines to shutdown, retry = 2
No virtual-machines active now.
Rebooting the system...
INIT: Sending processes the TERM signal

{master:0}
root@nfx250-m-p2a-sys11-jdm> Stopping OpenBSD Secure Shell server: sshdstopped
 /usr/sbin/sshd (pid 4169)
.
Unmount Junos cgroup... Done
Stopping atd: OK
Stopping domain name service: named.
Unmounting cgroups...Done
Stopping system message bus: dbus.
stopping DNS forwarder and DHCP server: dnsmasq... stopped /usr/bin/dnsmasq
 (pid 9449 9448)
done.
Stopping docker:
Stopping HOSTAP Daemon: no /usr/sbin/hostapd found; none killed
hostapd.
Shutting down irqbalance: no irqbalance found; none killed
done
Stopping ntpd: done
stopping rsyslogd ... done
Stopping internet superserver: xinetd.

Waiting for sanlock to stop: Success

Clearing ebtables rulesets: filter nat broute done. ok
Stopping crond: OK
Stopping S.M.A.R.T. daemon: smartd.
Stopping network management services: snmpd snmptrapd libvirtMib_subagent.
 * Stopping virtualization library daemon: libvirtd
Deconfiguring network interfaces... done.
Stopping tcsd: OK
Stopinit: Failed to release D-Bus name: Did not receive a reply. Possible
causes include: the remote application did not send a reply,
the message bus security policy blocked the reply, the reply timeout expired,
 or the network connection was broken.
ping redis-server...
Sending all processes the TERM signal...
Sending all processes the KILL signal...
Unmounting remote filesystems...
Deactivating swap...
```

```
Unmounting local filesystems...
Rebooting... RE-FPGA-DRV: reboot notifier called with 0x0001
RE-FPGA-DRV: Please standby while rebooting.
.
..
..
..
.

Booting from Flash A

FPGA Reset Reason = 0x80

Primary BIOS version CBDE_SFP_00.21_01.01

Total Memory Size = 16GB

Checking Primary BIOS code integrity...Passed!
Press Esc for boot options
ME is in normal operational state

Booting HDD00.1 (StorFly VSFBM6CC100G-JUN)...

Secure boot is enforced
Welcome to GRUB!

Secure Grub2 Diskboo
```

Related Documentation

- Reverting the System to the Factory-Default Configuration on page 27
- Rebooting the System on page 27

Downloading Software

- Downloading and Installing Software on page 23

Downloading and Installing Software

You can download the software package you need in one of two ways:

- Downloading Software by Using a Browser on page 24
- Installing Software by Using the Command-Line Interface on page 25

Downloading Software by Using a Browser

You download the software package you need from the Juniper Networks Support website at https://www.juniper.net/support/.

NOTE: To access the download section, you must have a service contract and an access account. If you need help obtaining an account, complete the registration form at the Juniper Networks website: https://userregistration.juniper.net/entitlement/setupAccountInfo.do.

To download the software image:

1. From your browser, go to https://www.juniper.net/support/.

   The Online Support (CSC) page opens.

2. Click the **Download Software** link.

   The Download Software page appears.

3. Select the software package that you want to download. You can select software that supports a specific platform or technology.

4. On the page that appears, click the **Software** tab and select the Junos OS installation package to download.

5. Log in with your username and password.

6. On the Download Software page that appears, the following options are available:

   - If you want to download the software on your local host, click the **CLICK HERE** link and save the file to your system. If you want to place the file on a remote system, you must make sure that the file can be accessible by the router, switch, or services gateway by using HTTP, FTP, or SCP. Proceed with the installation. See "Installing Software by Using the Command-Line Interface" on page 25 for more details.

   - If you want to download the software on your device, use the following procedure to download and install the software on the device.

     a. Click **Copy** to copy the generated URL generated to the clipboard.

        NOTE: The URL string generated remains active only for 15 minutes.

     b. Log in to your device.

     c. In operational mode, enter the **file copy "*URL*" *destination*** command.

        In the command, paste the copied URL string (for ***URL***) and then enter **/var/tmp** (as the destination on your hard disk).

        Example:

```
user@host> file copy
  "https://cdn.juniper.net/software/ittest/software_target/agileEcotTest/Dev_Binary_Build.tar?
  SM_USER=user1=1507622971_dce164fa854b4a27550c254eef950dd8"
  /var/tmp
```

NOTE: Ensure that the URL string is enclosed within quotation marks. Also ensure that there is sufficient free space available on the device.

The software image is downloaded on your device.

d. (Optional) Validate the software image by using the **request system software validate** *package-name* command.

   Example:

```
user@host> request system software validate /var/tmp/
  junos-install-mx-x86-32-17.3R1.10.tgz
```

   For more details, see request system software validate.

e. Install the software by using the **request system software add** *package-name* command.

   Example:

```
user@host> request system software add /var/tmp/
  junos-install-mx-x86-32-17.3R1.10.tgz
```

   Your software is installed on the device.

See Also

- Junos Platforms - Download Software

Installing Software by Using the Command-Line Interface

Download the software package you need from the Juniper Networks Support website at https://www.juniper.net/support/, and place the package on a local system. You can then transfer the downloaded package to the device using either the router or switch command-line interface, or the local system command-line interface.

NOTE: To access the download section, you must have a service contract and an access account. If you need help obtaining an account, complete the registration form at the Juniper Networks website: https://userregistration.juniper.net/entitlement/setupAccountInfo.do.

Before you transfer the software package, ensure that the FTP service is enabled on the device.
Enable the FTP service using the **set system services ftp** command:

```
user@host# set system services ftp
```

To transfer the software package using the device command-line interface:

1. From the router or switch command line, initiate an FTP session with the local system (host) where the package is located by using the **ftp** command:

   ```
   user@host> ftp host
   ```

   host is the hostname or address of the local system.

2. Log in with your customer support–supplied username and password:

   ```
   User Name: username
   331 Password required for username.
   Password: password
   ```

   After your credentials are validated, the FTP session opens.

3. Navigate to the software package location on the local system, and transfer the package by using the **get** command:

   ```
   user@host> get installation-package
   ```

   Following is an example of an *installation-package* name:
   junos-install-mx-x86-32-17.3R1.10.tgz

4. Close the FTP session by using the **bye** command:

   ```
   user@host> bye
   Goodbye
   ```

To transfer the package by using the local system command-line interface:

1. From the local system command line, initiate an FTP session with the device using the **ftp** command:

   ```
   user@host> ftp host
   ```

   host is the hostname or address of the router or switch.

2. Log in with your customer support–supplied username and password:

   ```
   User Name: username
   331 Password required for username.
   Password: password
   ```

   After your credentials are validated, the FTP session opens.

3. Navigate to the software package location on the local system, and transfer the package by using the **put** command:

```
user@host> put installation-package
```

Following is an example of an *installation-package* name:
junos-install-mx-x86-32-17.3R1.10.tgz

4. Close the FTP session by using the **bye** command:

```
user@host> bye
Goodbye
```

See Also

- Junos Platforms - Download Software

Reverting the System to the Factory-Default Configuration

To revert the system to factory-default configuration:

```
user@jdm# load factory-default
  warning: activating factory configuration
```

Related Documentation

- Upgrading an Image on the Disaggregated Junos OS Platform on page 20
- Rebooting the System on page 27

Rebooting the System

To reboot the system:

```
user@jdm>request system reboot
```

For example:

```
user@jdm>request system reboot
  Reboot the system ? [yes,no] (no) yes
  System reboot operation started, please wait...
  System going down for reboot in 30 seconds...
  System reboot in progress...
  Shutting down virtual-machines...

  . . . .
```

NOTE: The time taken to reboot the system depends on the number of active VNFs. The system is rebooted only after all the active VNFs are shut down.

Related Documentation

- Upgrading an Image on the Disaggregated Junos OS Platform on page 20
- Reverting the System to the Factory-Default Configuration on page 27

PART 3

Management

CHAPTER 3

Management

Understanding the JDM CLI

Junos Device Manager (JDM) can be configured using the JDM CLI. In most cases, you are logged into the JDM CLI by default when you access a disaggregated Junos OS platform.

The JDM CLI is similar to the Junos OS CLI in look and feel. It provides the same value-added facilities as the Junos OS CLI, which include:

- Separate configuration and command modes
- Commit check
- Configuration save, restore, and rollback
- NETCONF and YANG support

The JDM CLI is based on the Junos OS CLI and follows many of its processes and procedures. Like **Junos OS**, the JDM CLI has an operational mode and configuration mode. You use the **configuration** command to go from operational mode to configuration mode, and the **exit** command to exit configuration mode. Many operational mode commands—such as **show** and **request** commands—are available in the JDM CLI and the Junos OS CLI. Many configuration commands available in the Junos OS CLI are also available in the JDM CLI, and are often entered using the same command at the same hierarchy level.

If you are placed in the JDM shell for any reason, such as if you logged in to the disaggregated Junos OS platform as the root user, enter the **cli** command from the JDM shell prompt to get to the JDM CLI prompt:

```
root# cli
 root@jdm>
```

Related Documentation

- Accessing the JDM Shell, JDM CLI, and JCP Prompts in a Disaggregated Junos OS Platform on page 32
- Understanding Disaggregated Junos OS on page 3

Accessing the JDM Shell, JDM CLI, and JCP Prompts in a Disaggregated Junos OS Platform

This topic describes how to access the JDM shell, JDM CLI, and JCP prompts in a disaggregated Junos OS platform.

It contains the following sections:

- Accessing the JDM CLI on page 33
- Accessing the JDM Shell on page 33
- Accessing the JCP Prompt from the JDM CLI on page 33

Accessing the JDM CLI

You are in operational mode in the JDM CLI if you see the **@jdm>** prompt. The JDM CLI also includes the configuration prompt **@jdm#**, which can be accessed by entering the **configure** command at the JDM CLI operational mode prompt.

By default, most logins to a disaggregated Junos OS platform take the user to the operational mode in the JDM CLI prompt.

The JDM CLI prompt can also be accessed from the JDM shell.

To access the JDM CLI from the JDM shell, enter the **cli** command at the JDM shell prompt:

```
root@jdm:~# cli
root@jdm>
```

Accessing the JDM Shell

Be default, you are placed in the JDM shell when you login to the console port as the root user. The JDM shell uses the **~#** prompt.

To access the JDM shell from the JDM CLI, enter the **start shell** command at the JDM CLI prompt:

```
root@jdm> start shell
jdm:~#
```

Accessing the JCP Prompt from the JDM CLI

To access the JCP prompt from the JDM CLI, enter the **ssh vjunos0** statement at the JDM CLI prompt:

```
root@jdm> ssh vjunos0
The authenticity of host 'vjunos0 (192.168.1.2)' can't be established.

ECDSA key fingerprint is 18:83:1f:95:88:db:1b:75:9f:07:ce:2c:4a:45:a3:b0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'vjunos0,192.168.1.2' (ECDSA) to the list of known
 hosts.
Password:
Last login: Thu Oct  8 09:47:42 2015
--- JUNOS 15.1I20150824_0501 built 2015-08-24 05:13:01 UTC

root@:RE:0% cli
root>
```

Accessing the Hypervisor from the JDM CLI

To access the Hypervisor from the JDM CLI, enter the **ssh hypervisor** statement at the JDM CLI prompt:

```
root@jdm> ssh hypervisor
Last login: Sun Jan 18 15:01:55 2015 from jdm
```

NOTE: Only a root user can use this option.

Accessing the ipsec-nm from the JDM CLI

To access the ipsec-nm from the JDM CLI, enter the **ssh ipsec-nm** statement at the JDM CLI prompt:

```
root@jdm> ssh ipsec-nm
Last login: Sun Jan 18 15:01:55 2015 from jdm

root@ipsec-nm: % cli
root@ipsec-nm>
```

Related Documentation

- Understanding the JDM CLI on page 32
- Understanding Disaggregated Junos OS on page 3

Understanding User Accounts

On a disaggregated Junos OS platform, all computing elements are separate compute entities, and their user accounts and passwords are managed separately. For example, JDM user accounts, including the root user account, are completely separate from the Junos VM user accounts.

- Root Account on page 34
- Other User Accounts on page 34
- User Authentication on page 35

Root Account

In the factory-default configuration, the JDM is set up with a root user account. However, there is no password set for the account. You must configure a root password as part of the initial configuration. If the initial configuration of the platform is performed through the phone home feature, the configuration must contain the root password setting. Until you configure a root password, you cannot access some of the user prompts and you cannot commit a configuration by using the JDM CLI.

You can set the root password only from the JDM CLI. You cannot set or change the root password from the JDM shell. The JDM root password is automatically propagated to the JDM shell.

Other User Accounts

You can create user accounts other than the root account in the JDM. To do this, you must use the JDM CLI. You cannot use the JDM shell to create user accounts.

The JDM supports the same features for user accounts as does Junos OS. That is, the JDM supports login classes, custom password requirements, limits on the number of login attempts, and so on.

User Authentication

The JDM supports two of the three methods for user authentication that Junos OS supports: local password authentication and TACACS+ authentication. It does not support RADIUS authentication.

Related Documentation

- Configuring JDM User Accounts and Authentication on page 35
- Understanding the JDM CLI on page 32

Configuring JDM User Accounts and Authentication

You create user accounts and configure authentication for those accounts in JDM the same way you do in Junos OS. This topic provides some brief guidance on how to configure user accounts and authentication. For more details, consult the Junos OS documentation.

- To set the JDM root password:

```
root@jdm# set system root-authentication plain-text-password
```

  You must use the JDM CLI to set the root password. You cannot set the root password using the JDM shell.

- To create a new JDM user account:

```
root@jdm# set system login user user-name class class-name authentication
plain-text-password
```

  You cannot create JDM user accounts from the JDM shell.

- To configure SSH keys for a user to enable SSH without a password:

```
root@jdm# set system login user user-name load-key-file URL-to-ssh-key-file
```

- To configure TACAS+ authentication for user accounts:

```
root@jdm# set system tacplus-server server-address secret password
```

NOTE: TACACS+ is used to support SSH authentication, and once configured, TACACS+ configuration is applicable for both, JDM and host SSH authentication. On the host, TACACS+ is used to authenticate SSH requests only for the root account and when requested from outside the device.

Optionally, you can specify the TACACS+ authentication server port number and the timeout period. To do so:

```
root@jdm# set system tacplus-server server-address port port-number
```

```
root@jdm# set system tacplus-server server-address timeout period
```

NOTE: By default, the TACACS+ port number is set to 49, and the timeout period is set to 5 seconds.

Related Documentation

- Understanding User Accounts on page 34
- Understanding Disaggregated Junos OS on page 3

Understanding JDM Management Interfaces

You can access JDM through the system console, a dedicated out-of-band management interface, or an in-band management interface.

- Console Interface on page 36
- Out-of-Band Management Interface on page 36
- In-Band Management Interface on page 37

Console Interface

On disaggregated Junos OS platforms that host Juniper Device Manager (JDM), you are placed in either the JDM shell prompt or the JDM CLI when you first connect to a device using the device console port. You can access the JDM CLI or other VMs, including the Junos Control Plane (JCP) CLI that is used to manage Junos OS software, from this prompt. See "Accessing the JDM Shell, JDM CLI, and JCP Prompts in a Disaggregated Junos OS Platform" on page 32.

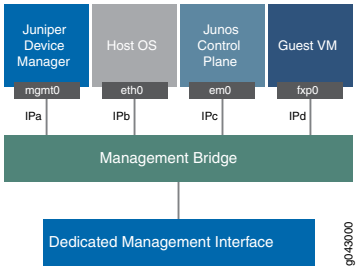
Out-of-Band Management Interface

The JDM out-of-band management interface is named jmgmt0. The jmgmt0 interface is directly connected to the dedicated Ethernet management port on the disaggregated Junos OS platform.

The jmgmt0 interface in a disaggregated Junos OS platform is analogous to the em0, me0, or fxp0 interfaces on a Juniper Networks switch or a router running traditional Junos OS software. To use jmgmt0 as a management port, you must configure a logical interface (jmgmt0.0) on it with a valid IP address. You can then connect to the management interface over the network using utilities such as SSH or Telnet. SNMP can be used on the management interface to gather statistics.

The dedicated Ethernet management port on the platform is shared by other compute entities. For example, JCP uses the dedicated Ethernet management port for its out-of-band management interface, em0, as shown in Figure 9 on page 37.

Figure 9: Out-of-band Management Interface

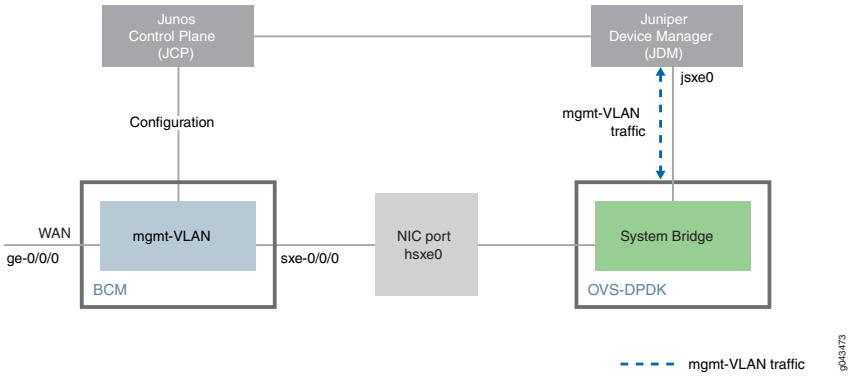


In-Band Management Interface

JDM has an interface—jsxe0—that can be used as in-band management interface. Unlike the out-of-band management interface jmgmt0, this interface is not directly connected to a physical port. You must connect jsxe0 to a physical interface through VLAN bridging—that is, you must configure both the physical interface and jsxe0 to be in the same management VLAN.

Figure 10 on page 37 illustrates how a network port is bridged to jsxe0. In this figure, ge-0/0/0 is the network port being used for in-band management. Interface sxe-0/0/0 is a JCP interface. Both ge-0/0/0 and sxe-0/0/0 are managed by JCP, and are configured in JCP to be part of the management VLAN, mgmt-vlan. JDM interface jsxe0 is also configured to part of the mgmt-vlan.

Figure 10: In-Band Management Interface Network



Related Documentation

- Configuring the In-Band Management Interface for JDM on page 39
- Understanding JDM Management Interfaces on page 36

Configuring the Out-of-Band Management Interface for JDM

This topic discusses how to configure an out-of-band management interface for JDM in a disaggregated Junos OS platform.

On a disaggregated Junos OS platform, the out-of-band management interface for JDM is named mgmt0. The mgmt0 interface has a direct connection to the dedicated Ethernet management port on the front panel of the device.

- Configuring the Out-of-Band Management Interface with IPv4 Addressing for JDM on page 38
- Configuring the Out-of-Band Management Interface with IPv6 Addressing for JDM on page 38

Configuring the Out-of-Band Management Interface with IPv4 Addressing for JDM

To configure the management interface with IPv4 addressing:

1. Configure the logical interface and the IP address:

```
root@jdm# set interfaces jmgmt0 unit 0 family inet address ipv4-address/mask
```

2. Set the default route:

```
root@jdm# set routing-options static route 0.0.0.0/0 nexthop ipv4-address
```

Configuring the Out-of-Band Management Interface with IPv6 Addressing for JDM

To configure the management interface with IPv6 addressing:

1. Configure the logical interface and the IP address:

```
root@jdm# set interfaces jmgmt0 unit 0 family inet6 address ipv6-address/mask
```

2. Set the default route:

```
root@jdm# set routing-options static route ::/0 nexthop ipv6-address
```

Related Documentation

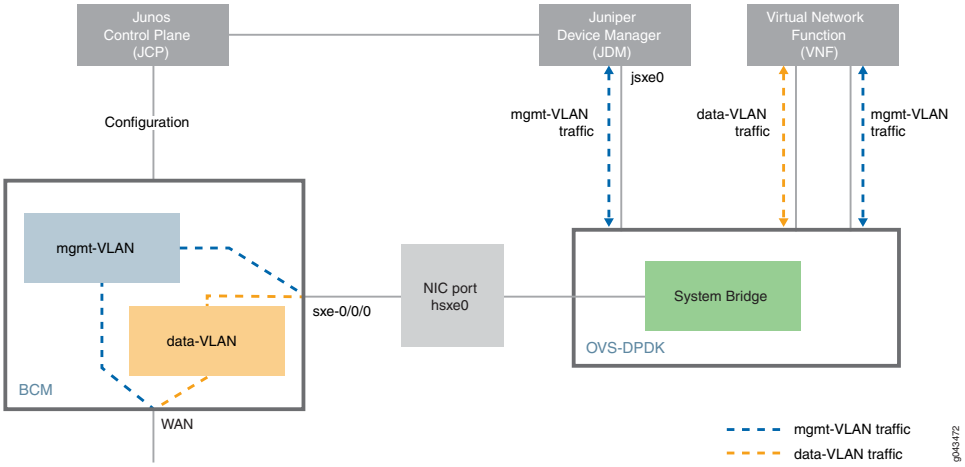
- Configuring the In-Band Management Interface for JDM on page 39

Configuring the In-Band Management Interface for JDM

JDM provides an internal interface—jsxe0—that can be used for in-band management. This internal interface is not directly connected to a physical interface. You must link jsxe0 to a physical interface through VLAN bridging—that is, you must configure both the physical interface and jsxe0 to be in the same management VLAN. See Figure 10 on page 37.

JCP, and not JDM manages the physical network interfaces and the service interfaces; therefore, you must first configure the sxe-0/0/0 and sxe-0/0/1 internal interfaces using the JCP CLI before you can manage the jsxe0 interface using the JDM CLI.

Figure 11: In-Band Management Interface Example



To configure jsxe0 as an in-band management interface:

1. Log in to the JCP CLI and enter configuration mode:

```
root@jdm> ssh vjunos0

root@:RE:0% cli
{master:0}
{master:0}[edit]
```

2. Configure the physical network port as a trunk port:

```
[edit]
root# set interfaces interface-name unit 0 family ethernet-switching
interface-mode trunk
```

For example:

```
[edit]
root# set interfaces ge-0/0/0 unit 0 family ethernet-switching interface-mode
 trunk
```

3. Configure a JCP service port as a trunk port:

```
root# set interfaces service-interface-name unit 0 family ethernet-switching
 interface-mode trunk
```

For example:

```
[edit]
root# set interfaces sxe-0/0/0 unit 0 family ethernet-switching
interface-mode trunk
```

4. Configure the management VLAN and add the physical network interface and the service interface as members of the VLAN:.

```
[edit]
root@jcp# set interfaces sxe-0/0/0 unit 0 family ethernet-switching vlan
members mgmt-vlan
root@jcp# set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan
members mgmt-vlan
```

For example:

```
[edit]
root@jcp# set vlans mgmt-vlan vlan-id vlan-id
```

5. Exit the JCP CLI to return to the JDM CLI (*hostname*@jdm> prompt):

```
[edit]
root# exit
Exiting configuration mode
root> exit
root% exit
logout
Connection to vjunos0 closed.
root@jdm>
```

6. Configure the jsxe0 interface as a trunk interface with membership in the management VLAN, and configure the management IP address on the interface:

```
root@jdm# set interfaces jsxe0 vlan-tagging
root@jdm# set interfaces jsxe0 unit logical-unit-number vlan-id mgmt-vlan-id
 family inet address mgmt-ip-address/prefix-length
```

Related Documentation

- Configuring the Out-of-Band Management Interface for JDM on page 37

Configuring the Out-of-Band Management Interface for Hypervisor

This topic discusses how to configure an out-of-band management interface for Hypervisor in a disaggregated Junos OS platform.

On a disaggregated Junos OS platform, the out-of-band management interface for Hypervisor is named eth0br. The eth0br interface has a direct connection to the dedicated Ethernet management port on the front panel of the device.

- Configuring the Out-of-Band Management Interface with IPv4 Addressing for Hypervisor on page 41
- Configuring the Out-of-Band Management Interface with IPv6 Addressing for Hypervisor on page 41

Configuring the Out-of-Band Management Interface with IPv4 Addressing for Hypervisor

To configure the management interface with IPv4 addressing:

1. Configure the logical interface and the IP address:

```
root@jdm# set host-os interfaces eth0br unit 0 family inet address
ipv4-address/mask
```

2. Set the default route:

```
root@jdm# set host-os routing-options static route 0.0.0.0/0 next-hop
gateway-ipv4-address
```

Configuring the Out-of-Band Management Interface with IPv6 Addressing for Hypervisor

To configure the management interface with IPv6 addressing:

1. Configure the logical interface and the IP address:

```
root@jdm# set host-os interfaces eth0br unit 0 family inet6 address
ipv6-address/mask
```

2. Set the default route:

```
root@jdm# set host-os routing-options static route ::/0 next-hop
gateway-ipv6-address
```

See Also

- Configuring the Out-of-Band Management Interface for JDM on page 37

Configuring SSH Service and NETCONF-Over-SSH Connections for Remote Access to the Disaggregated Junos OS Platform

You can configure the disaggregated Junos OS platform to accept NETCONF sessions over SSH as an access service.

To do so:

1. Access the system services SSH configuration:

```
[edit]
user@jdm#set system services netconf ssh
```

2. Configure the TCP port used for NETCONF-over-SSH connections:

```
[edit system services netconf ssh]
user@jdm#user@jdm# port port-number
```

The configured port only accepts NETCONG-over-SSH connections. Regular SSH connections to the port are ignored.

Related Documentation

- Understanding the JDM CLI on page 32
- Accessing the JDM Shell, JDM CLI, and JCP Prompts in a Disaggregated Junos OS Platform on page 32

Configuring HTTP Access to the Disaggregated Junos OS Platform

You can configure HTTP access to the disaggregated Junos OS platform.

To do so:

1. Access the system services HTTP configuration:

```
[edit]
user@jdm#edit system services http
```

2. Set the HTTP port for incoming connections:

```
[edit system services http]
user@jdm# set port port
```

Related Documentation

- Configuring HTTPS Access to the Disaggregated Junos OS Platform on page 42
- Understanding the JDM CLI on page 32
- Accessing the JDM Shell, JDM CLI, and JCP Prompts in a Disaggregated Junos OS Platform on page 32

Configuring HTTPS Access to the Disaggregated Junos OS Platform

You can configure HTTPS access to the disaggregated Junos OS platform.

To do so:

1. Access the HTTPS configuration:

```
[edit]
user@jdm#edit system services https
```

2. Set the HTTPS port for incoming connections:

```
[edit system services https]
user@jdm# set port port-number
```

Related Documentation

- Configuring HTTP Access to the Disaggregated Junos OS Platform on page 42
- Understanding the JDM CLI on page 32
- Accessing the JDM Shell, JDM CLI, and JCP Prompts in a Disaggregated Junos OS Platform on page 32

Configuring SNMP on JDM

There are several SNMP-enabled components in NFX (JDM, hypervisor, and so on). This topic discusses the SNMP implementation of JDM and hypervisor. For JCP, see the Junos documentation. On the NFX250 platform, JDM plays the role of the SNMP agent and at the same time it acts as an SNMP proxy for the hypervisor (host OS). When SNMP is configured in JDM, hypervisor also takes the same SNMP configuration. By default, SNMP is disabled on disaggregated Junos OS platforms. To enable SNMP , you must include the SNMP configuration statements at the **[edit snmp]** hierarchy level. This section describes:

- Configuring SNMP Community on page 43
- Configuring SNMP System Parameters on page 43
- Configuring SNMP v3 on page 44
- Configuring SNMP Traps on page 44
- Querying SNMP MIBs on page 44
- Managing Traps on page 45

Configuring SNMP Community

To configure SNMP community:

1. Specify a name for the SNMP community:

```
user@jdm# set snmp community community
```

Configuring SNMP System Parameters

To configure SNMP system parameters:

1. Set the system name:

   ```
   user@jdm# set snmp name name
   ```

2. Enter a description for the system being managed:

   ```
   user@jdm# set snmp description description
   ```

3. Specify the location of the system:

   ```
   user@jdm# set snmp location location
   ```

4. Specify the name of the contact person:

   ```
   user@jdm# set snmp contact contact
   ```

Configuring SNMP v3

In contrast to SNMP version 1 (SNMPv1) and SNMP version 2 (SNMPv2), SNMP version 3 (SNMPv3) supports authentication and encryption. SNMPv3 uses the user-based security model (USM) for message security and the view-based access control model (VACM) for access control. USM specifies authentication and encryption. VACM specifies access-control rules. To configure local engine information for the user-based security model (USM) with Secure Hash Algorithm (SHA) as the authentication type for the SNMPv3 user, enter the command:

```
user@jdm# set snmp v3 usm local-engine user username authentication-sha
authentication-password authentication-password
```

To configure local engine information for the USM with MD5 as the authentication type for the SNMPv3 user, enter the command:

```
user@jdm# set snmp v3 usm local-engine user username authentication-md5
authentication-password authentication-password
```

Configuring SNMP Traps

To configure SNMP traps, create a named group of hosts to receive the specified trap notifications. At least one trap group must be configured for SNMP traps to be sent:

```
user@jdm# set snmp trap-group group-name targets address
```

Querying SNMP MIBs

The NFX 250 platform supports querying SNMP MIBs on both, the JDM and the hypervisor. NFX MIBS are read-only, which means that the values can be read from the MIB but cannot be configured using SNMP.

The commands below are the queries on SNMP v1, SNMP v2 and SNMP v3. :

```
user@jdm# snmpwalk -v 1 -c community-name ip-address oid
```

```
user@jdm# snmpwalk -v 2 -c community-name ip-address oid
```

```
user@jdm# snmpwalk -v3 -u username -l authNoPriv -a SHA -A password ip-address oid
```

To query the hypervisor, you need to provide an additional context name, which is the user name appended by **-host**:

```
user@jdm# snmpwalk -v 1 -c community-name-host ip-address oid
```

```
user@jdm# snmpwalk -v 2 -c community-name-host ip-address oid
```

```
user@jdm# snmpwalk -v3 -u username-host -l authNoPriv -a SHA -A password ip-address oid
```

You can query libvirt MIBs only as a host:

```
user@jdm# snmpwalk -v 2c -c community-name-host ip-address oid
```

Managing Traps

The agent sends traps to notify the manager of significant events that occur on the device. To configure traps:

```
user@jdm# set snmp trap-group group-name targets ip-address
```

JDM traps are assigned the context **jdm**, and hypervisor traps are assigned the context **host**.

Related Documentation

- Understanding Disaggregated Junos OS on page 3

Configuring Enhanced Orchestration in the Disaggregated Junos OS Platform

Enhanced orchestration mode enables you to easily manage VNFs and service chains without requiring the VNF XML descriptor files. By default, this mode is ON and this is the recommended mode.

To enable enhanced orchestration:

```
[edit system services]
user@jdm# set enhanced-orchestration
```

NOTE: Ensure that you reboot the system after enabling the enhanced orchestration mode.

Related Documentation

- Understanding Disaggregated Junos OS on page 3

Configuring IPSec in the Disaggregated Junos OS Platform

The **ipsec-nm** mode allows you to enable or disable the ipsec-nm VNF. To secure the management traffic using ipsec tunnel, you must enable the **ipsec-nm** mode and configure the tunnel appropriately. By default, this mode is enabled.

To enable ipsec-nm:

```
[edit system services]
user@jdm# set system services ipsec-nm
```

To disable ipsec-nm:

```
[edit system services]
user@jdm# delete system services ipsec-nm
```

NOTE: CPU core 7 is available for use after you delete the ipsec-nm.

NOTE: Ensure that you reboot the system after enabling or disabling the ipsec-nm mode for the changes to take effect.

Related Documentation

- Understanding Disaggregated Junos OS on page 3

Viewing and Managing Centralized Log Files in a Disaggregated Junos OS Platform

On a disaggregated Junos OS platform, a centralized logging server collects all system logs for all computing entities in the disaggregated Junos OS. Centralized logging simplifies device management by allowing users to view all log files for all disaggregation Junos entities in a single place.

This topic describes:

- Enabling Centralized Logging on page 46
- Viewing Log Messages on page 47

Enabling Centralized Logging

You can enable centralized logging on vSRX to view the log files on JDM.

1. Log in to vSRX:

   ```
   root@jdm> ssh vsrx-hostname
   ```

2. Ensure that the fxp0 interface is assigned the management IP address:

   ```
   root>show interfaces fxp0 terse
   ```

3. Assign a hostname to the vSRX instance:

   ```
   root# set system host-name hostname
   ```

4. To specify JDM as the host for the central log file:

   ```
   root# set system syslog host 192.168.1.254 any any
   ```

   192.168.1.254 is the internal management IP address of JDM

   You can specify filters for the messages that you want displayed. If you specify **any any** all the messages will be logged.

Viewing Log Messages

To view system log messages logged by VNFs and the system generated logs on JDM, use the command:

```
user@jdm> show log syslog
```

To view other log messages, use the command:

```
user@jdm> show log <filename>
```

NOTE: You cannot view JCP log files on JDM.

A log rotation process runs every 5 minutes. If the log file size is greater than 5 MB, then the log files get rotated. The log files are saved in the /var/log directory.

Related Documentation

- Managing Core Files for a Disaggregated Junos OS Platform on page 47

Managing Core Files for a Disaggregated Junos OS Platform

This topic discusses how to view core files for a disaggregated Junos OS platform.

It contains the following sections:

Viewing Core Files

To view all core files from the JDM CLI, enter the **show system core-dumps** command:

```
/var/tmp/*core* :
drwxr-xr-x 2 root  root   4096 Jan 17 19:01 corefiles
/var/crash/*core* :
-rw-r--r-- 1 root root  382853 Jan  1 00:54 jdm.jdmd.4899.1420073655.core.tgz
```

```
-rw-r--r-- 1 root root  372095 Jan  1 01:11 jdm.jdmd.5697.1420074677.core.tgz
-rw-r--r-- 1 root root 3141405 Jan 17 19:01 jdm.jdmd.6875.1421521308.core.tgz
-rw-r--r-- 1 root root   18440 Jan 17 14:24 jdm.test.19278.1421504682.core.tgz
```

To view all core files displayed in JDM, open the core files using Unix commands. The core files are stored in the **/var/tmp/corefiles/** directory on Hypervisor.

Related Documentation

- Viewing and Managing Centralized Log Files in a Disaggregated Junos OS Platform on page 46

Synchronizing Time Using NTP

You can synchronize time on the following components of the NFX platform using Network Time Protocol (NTP):

- Junos Control Plane (JCP) - JCP runs the NTP server, and synchronizes time using the external NTP servers that are configured. JCP acts as the NTP server for the host.
- Host (hypervisor) - The host runs the NTP server and client, and synchronizes time using the Junos Control Plane (JCP) NTP server through JDM. The host, in turn, acts as the NTP server for the virtual network functions (VNFs).
- VNF - This is optional. VNFs run the NTP client, and synchronize the time using either JCP, hypervisor, or any external server that is configured.

To set the date and time using NTP:

1. Configure the NTP server and set the date on JCP:

   root# set system ntp server <ip-address>
   root# exit
   root> set date ntp

   Commit the configuration.

2. Once the NTP server has been configured on JCP, you can set the date and time on the host using JDM:

   root@jdm> set date ntp

3. (Optional) Set the local time zone to match the location of the device and to present the time in the correct local format. Universal Coordinated Time (UTC) is the default. Many administrators prefer to keep all their devices configured to use the UTC time zone. This approach has the benefit of allowing you to easily compare the time stamps of logs and other events across a network of devices in many different time zones.

   root@jdm# set time zone *time-zone*

NOTE: If the VNFs are not running the NTP clients, reboot the system to synchronize the date and time on all VNFs.

Related Documentation

- *Understanding NTP Time Servers*
- Understanding Virtual Network Functions on page 111

CHAPTER 4

Management Configuration Statements and Operational Commands

- show system visibility network
- show system visibility storage
- show system visibility vnf

enhanced-orchestration

Syntax enhanced-orchestration;

Hierarchy Level [edit system services]

Release Information Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description An option that toggles between set of configuration options used for the existing VNF configuration options and for the VNF orchestration that is based on vlan-aware bridges.

NOTE: By default, the enhanced-orchestration option is enabled and this is the only supported mode.

Required Privilege Level system—To view this statement in the configuration.

http

Syntax http;

Hierarchy Level [edit system services]

Release Information Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description Enable HTTP services.

Required Privilege Level system—To view this statement in the configuration.

https

Syntax https;

Hierarchy Level [edit system services]

Release Information Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description Enable HTTPS services.

Required Privilege Level system—To view this statement in the configuration.

ipsec-nm

Syntax ipsec-nm;

Hierarchy Level [edit system services]

Release Information Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description An option that enables or disables the ipsec docker container if the ipsec-nm option is configured in the system.

Required Privilege Level system—To view this statement in the configuration.

netconf

Syntax

```
netconf {
  ssh {
    port port-number;
  }
  traceoptions {
    flag [all | incoming | outgoing];
      file {
      file-name;
      size file-size;
    }
  }
}
```

Hierarchy Level

```
[edit system services]
```

Release Information Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description Allow NETCONF connections.

Options

ssh —Allow NETCONF connection over SSH.

port-number—Identifier of the service port.

traceoptions —Options that are available for the NETCONF trace operation.

flag —Parameters that can be specified for the NETCONF trace operation.

file-name—Name of file in which the trace information is available.

file-size—Maximum size of a trace file.

Required Privilege Level system—To view this statement in the configuration.

ntp

Syntax

```
ntp {
  server ip-address;
}
```

Hierarchy Level

```
[edit system]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description

Network Time Protocol (NTP) is used to synchronize the system clocks of routers, switches, and other network equipment. It provides the mechanisms to synchronize time and coordinate time distribution in a large, diverse network.

Required Privilege Level

system—To view this statement in the configuration.

Related Documentation

- Synchronizing Time Using NTP on page 48

outbound-ssh

Syntax

```
outbound-ssh {
  client client id {
    address;
      device-id device-id;
      disable-ssh-security-settings;
      keep-alive;
      reconnect-strategy;
      secret;
    services;
  }
  traceoptions {
    flag [all | configuration | connectivity];
    file {
      file-name;
      size file-size;
    }
  }
}
```

Hierarchy Level

```
[edit system services]
```

Release Information Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description Initiate outbound SSH connection.

Options ***client- id***—Identifier of a client application that initiates the SSH connection.

address—Address of the client to which the connection must be established.

device-id—Unique ID used by the client to identify a device.

traceoptions—Options that are available for the outbound SSH trace operation.

flag—Parameters that can be specified for the outbound SSH trace operation.

file-name—Name of file in which the trace information is available.

file-size— Maximum size of a trace file.

Required Privilege Level system—To view this statement in the configuration.

phone-home

Syntax

```
phone-home {
  server;
upgrade-image-before-configuration;
  traceoptions {
  no-remote-trace;
    file {
      file-name;
      size file-size;
    }
    flag {
      all;
      config;
      function;
      misc;
      socket;
      state-machine;
    }
```

Hierarchy Level

```
[edit system services]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 device.
Statement introduced in Junos OS Release 18.1R1 for the NFX150 device.

Description

An option that is used for initial boot up and configuration of the device when the client device is switched on.

The default configuration for phone-home is as follows:

```
user@jdm#  set system phone-home server https://redirect.juniper.net
user@jdm#  set system phone-home upgrade-image-before-configuration
```

Options

server—Name of the server.

upgrade-image-before-configuration—Upgrades the image before applying the configuration received from the Network Activator.

traceoptions—Options that are available for the phone-home trace operation

no-remote-trace—

file-name—Name of file in which the trace information is available

file-size—Maximum size of a trace file.

flag—Parameters that can be specified for the phone-home trace operation.

Required Privilege Level system—To view this statement in the configuration.

rest

Syntax

```
rest {
  control;
    http;
  traceoptions;
}
```

Hierarchy Level [edit system services]

Release Information Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description Allow remote procedure call (RPC) over HTTP or HTTPS connection

Options **control**—Control of the REST API process.

traceoptions—Options that are available for the REST API trace operation.

http—HTTP connection settings that are not encrypted.

Required Privilege Level system—To view this statement in the configuration.

ssh

Syntax

```
ssh;
```

Hierarchy Level [edit system services]

Release Information Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description Allow ssh access.

Required Privilege Level system—To view this statement in the configuration.

system

Syntax

```
system
  ntp {
    server ip-address;
  }
  phone-home {
  upgrade-image-before-configuration;
    traceoptions {
    no-remote-trace;
      file {
        file-name;
        size file-size;
      }
      flag {
        all;
        config;
        function;
        misc;
        socket;
        state-machine;
      }
  services {
    enhanced-orchestration;
    ipsec-nm;
    outbound-ssh {
      client client id {
        address;
          device-id device-id;
          disable-ssh-security-settings;
          keep-alive;
          reconnect-strategy;
          secret;
        services;
      }
      traceoptions {
        flag [all | configuration | connectivity];
        file {
          file-name;
          size file-size;
        }
      }
    }
    http;
    https;
    netconf {
      ssh {
        port port-number;
      }
      traceoptions {
        flag [all | incoming | outgoing];
          file {
          file-name;
```

```
            size file-size;
          }
        }
      }
      rest {
        control;
          http;
        traceoptions;
      }
      ssh;
    }
}
```

Hierarchy Level [edit]

Release Information Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description Configure system management properties.

Options ***client- id***—Identifier of a client application that initiates the SSH connection.

address—Address of the client to which the connection must be established.

device-id—Unique ID used by the client to identify a device.

file-name—Name of file in which the trace information is available.

file-size— Maximum size of a trace file.

port-number—Identifier of the HTTP port.

flag—Parameters that can be specified for the tracing operation.

Required Privilege Level system—To view this statement in the configuration.

traceoptions

Syntax

```
traceoptions {
no-remote-trace;
  file {
    file-name;
    size file-size;
  }
  flag {
    all;
    config;
    function;
    misc;
    socket;
    state-machine;
  }
```

Hierarchy Level

[edit system]

Release Information

Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description

An option that is used for the phone-home trace operations.

Options

traceoptions—Options that are available for the phone-home trace operations.

no-remote-trace—Trace operations for the phone-home is not supported.

file-name—Name of file in which the trace information is available.

file-size—Maximum size of a trace file.

flag—Parameters that can be specified for the phone-home trace operation.

Required Privilege Level

system—To view this statement in the configuration.

upgrade-image-before-configuration

Syntax upgrade-image-before-configuration;

Hierarchy Level [edit system]

Release Information Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description An option to upgrade the image before applying the configuration received from the Network Activator.

Required Privilege Level system—To view this statement in the configuration.

show connections

Syntax **show connections**

Release Information Command introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description Displays information such as network connection, function, interface name, and the connection status for the following types of cross-connect:

- Unconditional cross-connect
- Cross-connect based on the VLAN
- Cross connect with operations such as PUSH, POP, and SWAP on a VLAN tag
- Cross connect of native VLAN traffic

Required Privilege Level view

Related Documentation
- cross-connect on page 140

List of Sample Output show connections on page 63

Output Fields Table 3 on page 63 lists the output fields for the **show connections** command. Output fields are listed in the approximate order in which they appear.

Table 3: show connections Output Fields

Field Name	Field Description
Connection	Displays the type of the cross-connect.
Function	Displays the name of the virtual network function.
Interface	Specifies an interface on which the connection is established.
Status	Displays the status of the connection.

Sample Output

show connections

```
user@jdm> show connections
Connection                  Function                    Interface Vlan  Status
--------------------------------------------------------------------------------
```

```
phy_cc                  system                  hsxe0     200   up
                        centos1                 eth2      500

push_pop_cc             centos1                 eth2      none  down
                        centos2                 eth3      none

swap_cc                 centos1                 eth2      300   up
                        centos2                 eth2      400

uncond_cc               centos1                 eth2      none  up
                        centos2                 eth2      none

vlan_cc                 centos1                 eth2      100   up
                        centos2                 eth2      100
```

show forwarding-options analyzer

Syntax **show forwarding-options analyzer [*analyzer-instance-name*]**

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Displays information about the VNF analyzers that are configured for port mirroring on a disaggregated Junos OS platform.

Options ***analyzer-instance-name***—(Optional) Displays the details of a specific analyzer on the device.

Required Privilege Level view

Related Documentation
- TBD

List of Sample Output show forwarding-options analyzer on page 65

Output Fields Table 4 on page 65 lists the output fields for the **show forwarding-options analyzer** command. Output fields are listed in the approximate order in which they appear.

Table 4: show forwarding-options analyzer Output Fields

Field Name	Field Description
Analyzer instance name	Displays the name of the analyzer instance.
Egress monitored interfaces	Displays interfaces for which traffic leaving the interfaces is mirrored.
Output interface	Specifies an interface to which mirrored packets are sent.
Ingress monitored interfaces	Displays interfaces for which traffic entering the interfaces is mirrored.

Sample Output

show forwarding-options analyzer

```
user@jdm> show forwarding-options analyzer
Analyzer name                    : mon1
  Egress monitored interfaces      : vnf1:eth2
  Output interface                 : analyzer1:eth2

  Analyzer name                    : mon2
```

```
Ingress monitored interfaces     : vnf2:eth2
Output interface                 : analyzer1:eth3
```

show system inventory hardware cpu

Syntax show system inventory hardware cpu

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Display system CPU statistics for a disaggregated Junos OS platform.

Required Privilege Level view

Related Documentation
- show system inventory hardware memory on page 70
- show system inventory hardware network on page 72
- show system inventory hardware storage on page 74

List of Sample Output show system inventory hardware cpu on page 68

Output Fields Table 5 on page 67 lists the output fields for the **show system inventory hardware cpu** command. Output fields are listed in the approximate order in which they appear.

Table 5: show system inventory hardware cpu Output Fields

Field Name	Field Description
Fields for Inventory CPU Information	
No of CPUs	Total number of CPUs.
No of Logical CPUs/Hyper threads	Total number of hyper threads.
No of CPU sockets	Total number of CPU sockets.
No of Cores per socket	Total number of cores per socket.
No of Hyper threads per core	Total number of hyper threads per core.
CPU Vendor	The name of the CPU vendor.
CPU Model	The CPU model.
CPU Architecture	The type of CPU architecture.
CPU Speed	The CPU speed, in GHz.
CPU Cache	The size of the CPU cache.

Table 5: show system inventory hardware cpu Output Fields (continued)

Field Name	Field Description
No of Max vCPUs	The maximum number of allowed virtual CPUs.
Fields for CPU Statistics	
User Time	The amount of user time, in seconds.
System Time	The amount of system time, in seconds.
Idle Time	The amount of time spent in idle mode, in seconds.
I/O Wait Time	The amount of time spent waiting for input/output (I/O) operations, in seconds.
Nice Time	The amount of spent nice time, in seconds.
Interrupt Service Time	The amount of interrupt service time, in seconds.
Fields for CPU Pinning Information	
Virtual Machine	The name of the virtual machine.
vCPU	The ID of virtual CPUs used by the virtual machine.
CPU	The ID of CPUs used by the virtual machine.

Sample Output

show system inventory hardware cpu

```
user@jdm> show system inventory hardware cpu
Inventory CPU Information
-------------------------
No of CPUs:                       6
No of Logical CPUs/Hyper threads: 12
No of CPU sockets:                1
No of Cores per socket:           6
No of Hyper threads per core:     2
CPU Vendor:                       GenuineIntel
CPU Model:                        Intel(R) Xeon(R) CPU D-1528 @ 1.90GHz
CPU Architecture:                 x86_64
CPU Speed:                        1.9000 GHz
CPU Cache:                        9216 KB

CPU Statistics (Time in sec)
----------------------------
No of Max VCPUs:                  16
User Time:              449825
System Time:            0
Idle Time:              4475646
I/O Wait Time:          578
Nice Time:              14325
Interrupt Service Time: 0
```

```
CPU Pinning Information
-----------------------------------
Virtual Machine             vCPU CPU
--------------------------- ---- ---
vjunos0                     0    0
vjunos0                     0    1
vjunos0                     0    6
```

show system inventory hardware memory

Syntax show system inventory hardware memory

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Display hardware memory statistics for a disaggregated Junos OS platform.

Required Privilege Level view

Related Documentation
- show system inventory hardware cpu on page 67
- show system inventory hardware network on page 72
- show system inventory hardware storage on page 74

List of Sample Output show system inventory hardware memory on page 71

Output Fields Table 6 on page 70 lists the output fields for the **show system inventory hardware memory** command. Output fields are listed in the approximate order in which they appear.

Table 6: show system inventory hardware memory Output Fields

Field Name	Field Description
Fields for Virtual Memory	
Total	The total amount of available virtual memory, in kibibytes (KiBs).
Used	The total amount of used virtual memory, in kibibytes (KiBs).
Available	The total amount of available virtual memory, in kibibytes (KiBs).
Free	The total amount of free virtual memory, in kibibytes (KiBs).
Percent Used	The percentage of virtual memory used.
Fields for Swap Memory	
Total	The total amount of available swap space memory, in kibibytes (KiBs).
Used	The total amount of used swap space memory, in kibibytes (KiBs).
Free	The total amount of free swap space memory, in kibibytes (KiBs).
Percent Used	The percentage of swap space memory used.

Sample Output

show system inventory hardware memory

```
user@jdm> show system inventory hardware memory
Inventory Memory Information
----------------------------
Virtual Memory:
---------------
Total  (KiB): 15949116
Used   (KiB): 5542256
Available (KiB): 10437928
Free   (KiB): 10406860
Percent Used: 28.6

Swap Memory:
------------
Total  (KiB): 1249996
Used   (KiB): 0
Free   (KiB): 1249996
Percent Used: 0.0
```

show system inventory hardware network

Syntax show system inventory hardware network

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Display details such as MAC address pool and internal IP address range for VNFs and the number of free Virtual Functions available per Physical Function for VNFs for a disaggregated Junos OS platform.

Required Privilege Level view

Related Documentation

- show system inventory hardware cpu on page 67
- show system inventory hardware memory on page 70
- show system inventory hardware storage on page 74

List of Sample Output show system inventory hardware network on page 73

Output Fields Table 7 on page 72 lists the output fields for the **show system inventory hardware network** command. Output fields are listed in the approximate order in which they appear.

Table 7: show system inventory hardware network Output Fields

Field Name	Field Description
Fields for VNF MAC Address Pool	
Start MAC Address	The first MAC address in the MAC address pool.
Range	The number of MAC addresses available.
Fields for VNF Internal IP Address Range	
Start IP Address	The first IP address in the internal IP address range.
End IP Address	The last IP address in the internal IP address range.
Fields for Virtual Functions Available per Physical Functions	
Physical Function	The names of the Physical Functions available.
Virtual Function	Virtual Functions available for each Physical Function.

Sample Output

show system inventory hardware network

```
user@jdm> show system inventory hardware network

VNF MAC Address Pool
--------------------
Start MAC Address: 30:7c:5e:4c:3f:54
Range:              92

VNF Internal IP Address Range
-----------------------------
Start IP Address: 192.168.1.100
End IP Address:   192.168.1.199

Number of VFs per PF
---------------------------------
Physical Function Virtual Function
----------------- ----------------
hsxe0             16
hsxe1             16
```

show system inventory hardware storage

Syntax show system inventory hardware storage

Release Information Command introduced in Junos OS Release 15.1X53-D40 or the NFX250 Network Services Platform.

Description Display hardware storage details such as the list of partitions, disk usage per partition, and disk I/O statistics for a disaggregated Junos OS platform.

Required Privilege Level view

Related Documentation

- show system inventory hardware cpu on page 67
- show system inventory hardware memory on page 70
- show system inventory hardware network on page 72

List of Sample Output show system inventory hardware storage on page 75

Output Fields Table 8 on page 74 lists the output fields for the **show system inventory hardware storage** command. Output fields are listed in the approximate order in which they appear.

Table 8: show system inventory hardware storage Output Fields

Field Name	Field Description
Fields for List of Partitions	
Device	The device path.
Mount Point	The mount point of the device path.
File System	The file system type.
Options	Options available for the device path.
Fields for Disk Usage Information	
Total	The total amount of disk usage space, in mebibytes (MiB).
Used	The amount of used disk usage space, in mebibytes (MiB).
Free	The amount of free disk usage space, in mebibytes (MiB).
Percentage Used	The percentage of used disk space.
Fields for Disk I/O Information	

Table 8: show system inventory hardware storage Output Fields (continued)

Field Name	Field Description
Read Count	The number of times the disk has been read.
Write Count	The number of times a write operation has happened on the disk.
Read Bytes	The number of bytes used in read operations on the disk.
Write Bytes	The number of bytes used in write operations on the disk.
Read Time	The amount of time the disk has been read, in milliseconds.
Write Time	The amount of time write operations have been performed on the disk, in milliseconds.

Sample Output

show system inventory hardware storage

```
user@jdm> show system inventory hardware storage
Device                                      Mount Point      File System  Options
------------------------------------------- ---------------- -----------
----------------------------------
/dev/sda4                                   /                ext4
rw,relatime,data=ordered
/dev/sda5                                   /var             ext4
rw,relatime,data=ordered
/dev/sda2                                   /boot            vfat
rw,noatime,fmask=0022,dmask=0022,codepage=437,iocharset=iso8859-1,shortname=mixed,errors=remount-ro
/dev/sda1                                   /boot/efi        vfat
rw,noatime,fmask=0022,dmask=0022,codepage=437,iocharset=iso8859-1,shortname=mixed,errors=remount-ro
/dev/sda3                                  /app_disk        ext4         rw,noatime
/dev/mapper/vg0_vjunos-lv_junos             /junos           ext4
rw,noatime,discard,data=ordered
/dev/mapper/vg0_vjunos-lv_var_third_party /third-party     ext4
rw,noatime,discard,data=ordered

Disk Usage Information
-------------------------------------------------------------------------------------------
Disk                                         Total (MiB) Used (MiB)  Free (MiB)
% Used
-------------------------------------------- ----------- ----------- -----------
------
/dev/sda4                                    1409        822         497
58.0
/dev/sda5                                    5639        201         5128
3.0
/dev/sda2                                    976         569         406
58.0
/dev/sda1                                    189         17          172
9.0
/dev/sda3                                    1           0           1
1.0
/dev/mapper/vg0_vjunos-lv_junos              9951        3986        5436
40.0
```

```
/dev/mapper/vg0_vjunos-lv_var_third_party   94849       5734        84274
6.0

Disk I/O Information
--------------------
 Read Count: 304501
Write Count: 1176577
 Read Bytes: 4199641600
Write Bytes: 16493698560
  Read Time: 45528
 Write Time: 1181938
```

show system inventory software vnf

Syntax show system inventory software vnf

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Display the list of the virtual network functions available on a disaggregated Junos OS platform.

Required Privilege Level view

Related Documentation
- show system visibility vnf on page 104

List of Sample Output show system inventory software vnf on page 77

Output Fields Table 9 on page 77 lists the output fields for the **show system inventory software vnf** command. Output fields are listed in the approximate order in which they appear.

Table 9: show system inventory software vnf Output Fields

Field Name	Field Description
Fields for List of Virtual Network Functions	
ID	The ID number of the VNF.
Name	The name of the VNF.
State	The state of the VNF.

Sample Output

show system inventory software vnf

```
user@jdm> show system inventory software vnf
List of VNFs
------------------------------------------------------------
ID   Name                                  State
---- ------------------------------------- ---------------
4    vnf1                                  Running
```

show system services ipsec-nm

Syntax show system services ipsec-nm

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Display details such as status and mode of an ipsec-nm docker container for a disaggregated Junos OS platform.

Required Privilege Level view

Related Documentation

- show system visibility host on page 83
- show system visibility jcp on page 89
- show system visibility jdm on page 92
- show system visibility memory on page 96
- show system visibility network on page 98
- show system visibility storage on page 101
- show system visibility vnf on page 104

List of Sample Output show system services ipsec-nm on page 78

Output Fields Table 10 on page 78 lists the output fields for the **show system services ipsec-nm** command. Output fields are listed in the approximate order in which they appear.

Table 10: show system services ipsec-nm Output Fields

Field Name	Field Description
IPSec-NM Docker-Container Information	
Mode	The mode of the ipsec-nm docker container. Possible values are **Enabled** and **Disabled**.
Status	The status of the ipsec-nm docker container. Possible values are **Running** and **Not Running**.

Sample Output

show system services ipsec-nm

```
user@jdm> show system services ipsec-nm

IPSec-NM Docker-Container Information
-------------------------------------
```

```
Mode:               Enabled
Status:             Running
```

show system visibility cpu

Syntax show system visibility cpu

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Display details such as per CPU statistics, per CPU usage, and CPU pinning for a disaggregated Junos OS platform.

Required Privilege Level view

Related Documentation

- show system visibility host on page 83
- show system visibility jcp on page 89
- show system visibility jdm on page 92
- show system visibility memory on page 96
- show system visibility network on page 98
- show system visibility storage on page 101
- show system visibility vnf on page 104

List of Sample Output show system visibility cpu on page 81

Output Fields Table 10 on page 78 lists the output fields for the **show system visibility cpu** command. Output fields are listed in the approximate order in which they appear.

Table 11: show system visibility cpu Output Fields

Field Name	Field Description
Fields for CPU Statistics	
CPU ID	The CPU ID
User Time	The amount of user time, in seconds.
System Time	The amount of system time, in seconds.
Idle Time	The amount of time spent in idle mode, in seconds.
Nice Time	The amount of spent nice time, in seconds.
I/O Wait Time	The amount of time spent waiting for input/output (I/O) operations, in seconds.
Interrupt Service Time	The amount of interrupt service time, in seconds.

Table 11: show system visibility cpu Output Fields (continued)

Field Name	Field Description
Service Time	The amount of service time, in seconds.
Fields for CPU Usages	
CPU ID	The CPU ID
CPU Usage	The percentage of CPU used.
Fields for CPU Pinning Information	
Virtual Machine	The name of the virtual machine.
vCPU	The ID of virtual CPUs used by the virtual machine.
CPU	The ID of CPUs used by the virtual machine.

Sample Output

show system visibility cpu

```
user@jdm> show system visibility cpu
CPU Statistics (Time in sec)
-------------------------------------------------------------------------------
CPU Id User Time System Time Idle Time Nice Time IOWait Time Intr. Service Time
------ --------- ----------- --------- --------- ----------- ------------------
0      11267     4476        395088    532       0           0
1      14204     5195        392493    28        0           0
2      413638    7           40        0         0           0
3      0         2           413448    15        0           0
4      405       220         412850    1         0           0
5      0         0           413476    2         0           0
6      11908     4470        395821    1         0           0
7      0         0           413678    0         0           0
8      0         0           413679    0         0           0
9      0         0           413680    0         0           0
10     1         0           413677    0         0           0
11     1         1           413675    0         0           0

CPU Usages
----------------
CPU Id CPU Usage
------ ---------
0      6.9000000000000004
1      7.7999999999999998
2      100.0
3      0.0
4      0.0
5      0.0
6      4.9000000000000004
7      0.0
8      0.0
9      0.0
10     0.0
```

```
11     0.0

CPU Pinning Information
-------------------------------------
Virtual Machine             vCPU CPU
--------------------------- ---- ---
vjunos0                     0    0
```

show system visibility host

Syntax show system visibility host

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Display details such as the host uptime, number of tasks, CPU statistics, list of disk partitions, disk usage, disk I/O statistics, list of network interfaces, and per port statistics for a disaggregated Junos OS platform.

Required Privilege Level view

Related Documentation

- show system visibility cpu on page 80
- show system visibility jcp on page 89
- show system visibility jdm on page 92
- show system visibility memory on page 96
- show system visibility network on page 98
- show system visibility storage on page 101
- show system visibility vnf on page 104

List of Sample Output show system visibility host on page 85

Output Fields Table 12 on page 83 lists the output fields for the **show system visibility host** command. Output fields are listed in the approximate order in which they appear.

Table 12: show system visibility host Output Fields

Field Name	Field Description
Field for Host Uptime	
Uptime	The time the host has been operational.
Fields for Host Tasks	
Total	The total number of tasks.
Running	The total number of tasks running.
Sleeping	The total number of tasks in sleeping state.
Stopped	The total number of tasks that are stopped.

Table 12: show system visibility host Output Fields (continued)

Field Name	Field Description
Zombie	The total number of zombie processes.
Fields for Host CPU Information	
User Time	The amount of user time, in seconds.
System Time	The amount of system time, in seconds.
Idle Time	The amount of time spent in idle mode, in seconds.
Nice Time	The amount of spent nice time, in seconds.
I/O Wait Time	The amount of time spent waiting for input/output (I/O) operations, in seconds.
Interrupt Service Time	The amount of interrupt service time, in seconds.
Fields for Host Disk Partitions	
Device	The device path.
Mount Point	The mount point of the device path.
File System	The file system type.
Options	Options available for the device path.
Fields for Host Disk Usage Information	
Total	The total amount of disk usage space, in mebibytes (MiB).
Used	The amount of used disk usage space, in mebibytes (MiB).
Free	The amount of free disk usage space, in mebibytes (MiB).
Percentage Used	The percentage of used disk space.
Fields for Host Disk I/O Information	
Read Count	The number of times the disk has been read.
Write Count	The number of times a write operation has happened on the disk.
Read Bytes	The number of bytes used in read operations on the disk.
Write Bytes	The number of bytes used in write operations on the disk.
Read Time	The amount of time the disk has been read, in milliseconds.
Write Time	The amount of time write operations have been performed on the disk, in milliseconds.

Table 12: show system visibility host Output Fields (continued)

Field Name	Field Description
Fields for List of Host Interfaces	
Interfaces	The name of the interface.
State	The state of the Host Interface.
MAC	The MAC address of the interface.
Fields for List of Host Port Statistics	
Interface	The name of the interface.
Bytes Sent	The number of bytes sent.
Bytes Received	The number of bytes received.
Packets Sent	The number of packets sent.
Packets Received	The number of packets received.
Errors In	The number of errors in.
Errors Out	The number of errors out.
Drops In	The number of drops in.
Drops Out	The number of drops out.

Sample Output

show system visibility host

```
user@jdm> show system visibility host
Host Uptime
-----------
Uptime: 4 days 18:55:15.410000

Host Tasks
----------
Total:    229
Running:  1
Sleeping: 225
Stopped:  0
 Zombie:   3

Host CPU Information (Time in sec)
----------------------------------
User Time:              451464
System Time:            0
Idle Time:              4491938
```

```
I/O Wait Time:          580
Nice Time:              14378
Interrupt Service Time: 0

Host Disk Partitions
-------------------------------------------------------------------------------------------------------------
Device                                      Mount Point      File System  Options
------------------------------------------ ---------------- -----------
----------------------------------
/dev/sda4                                   /                ext4
rw,relatime,data=ordered
/dev/sda5                                   /var             ext4
rw,relatime,data=ordered
/dev/sda2                                   /boot            vfat
rw,noatime,fmask=0022,dmask=0022,codepage=437,iocharset=iso8859-1,shortname=mixed,errors=remount-ro
/dev/sda1                                   /boot/efi        vfat
rw,noatime,fmask=0022,dmask=0022,codepage=437,iocharset=iso8859-1,shortname=mixed,errors=remount-ro
/dev/sda3                                  /app_disk        ext4         rw,noatime
/dev/mapper/vg0_vjunos-lv_junos             /junos           ext4
rw,noatime,discard,data=ordered
/dev/mapper/vg0_vjunos-lv_var_third_party /third-party     ext4
rw,noatime,discard,data=ordered

Host Disk Usage Information
---------------------------
Total (MiB):     1409
Used  (MiB):     822
Free  (MiB):     497
Percentage Used: 58.4

Host Disk I/O Information
-------------------------
 Read Count: 304507
Write Count: 1180545
 Read Bytes: 4199883264
Write Bytes: 16525987328
  Read Time: 45530
 Write Time: 1185740

Host Interfaces
-----------------------------------------------
Interface          State     MAC
------------------ --------- -----------------
hsxe0              active    00:a0:c9:00:00:00
hsxe1              active    34:12:78:56:01:00
ctrlbr0            active    22:ce:d2:99:bf:9e
docker0            inactive  56:84:7a:fe:97:99
eth0br             active    30:7c:5e:4c:78:40
eth1br             inactive  d6:2c:a9:35:1b:b7
ipsec-nm_heth1     active    2a:ec:05:75:b8:2b
ipsec-nm_heth2     active    6e:71:0c:e8:7d:44
jdm_jsxe0          active    a6:a1:89:d7:77:5b
jdm_phc            active    f2:b6:80:39:15:32
lo                 inactive  00:00:00:00:00:00
sit0               inactive  00:00:00:00
virbr0             active    30:7c:5e:4c:78:43

Host Port Statistics
-------------------------------------------------------------------------------------------------------------
Interface Bytes Sent   Bytes Rcvd   Packets Sent Packets Rcvd Errors In Errors
```

```
Out Drops In Drops Out
--------- ------------ ------------ ------------ ------------ --------- ----------
 -------- ---------
ipsec-nm_heth1 4666    508          57           6            0         0
   0        0
ipsec-nm_heth2 4576    508          56           6            0         0
   0        0
ovs-sys-br 0           4526         0            55           0         0
   55       0
vnet0     135899707    32003        956220       561          0         0
   0        0
vnet1     135883353    0            956081       0            0         0
   0        0
vnet2     12145466     6312130      105565       46664        0         0
   0        0
ovs-netdev 0           0            0            0            0         0
   0        0
vnet4     6524         648          99           8            0         0
   0        0
vnet5     136365       0            1025         0            0         0
   0        953940
vnet8     105983497    16071923     325389       251433       0         0
   0        0
ipsecnm-heth0 2519822  648          48484        8            0         0
   0        0
eth0      33431        136998800    579          961948       0         0
   3473     0
iri1      100354502    92371048     1287131      1255994      0         0
   0        0
lo        18460        18460        192          192          0         0
   0        0
irb       0            0            0            0            0         0
   0        0
hsxe1     648          0            8            0            0         0
   0        0
hsxe0     648          1016         8            12           0         0
   0        0
docker0   0            0            0            0            0         0
   0        0
dcapi-tap 0            0            0            0            0         0
   0        0
sit0      0            0            0            0            0         0
   0        0
virbr0-nic 0           0            0            0            0         0
   0        0
virbr0    110199537    18210327     290265       298091       0         0
   0        0
vnet3     53436        0            1024         0            0         0
   0        46845
eth0br    648          122171753    8            955407       0         0
   0        0
ctrlbr0   30250640     70328038     371418       1256002      0         0
   0        0
eth1br    648          0            8            0            0         0
   0        0
jdm_jsxe0 4158         0            51           0            0         0
   0        0
vjunos0_em1 4158       0            51           0            0         0
   0        0
```

```
jdm_phc    4158          0             51            0             0          0
   0          0
```

show system visibility jcp

Syntax show system visibility jcp

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Display details such as CPU statistics, memory usage, internal IP address, list of network interfaces, interface statistics, and the list of disks for Junos VM.

Required Privilege Level view

Related Documentation

- show system visibility cpu on page 80
- show system visibility host on page 83
- show system visibility jdm on page 92
- show system visibility memory on page 96
- show system visibility network on page 98
- show system visibility storage on page 101
- show system visibility vnf on page 104

List of Sample Output show system visibility jcp on page 90

Output Fields Table 13 on page 89 lists the output fields for the **show system visibility jcp** command. Output fields are listed in the approximate order in which they appear.

Table 13: show system visibility jcp Output Fields

Field Name	Field Description
Fields for JCP CPU Statistics	
CPU Time	The total CPU time, in seconds.
System Time	The total system time, in seconds.
User Time	The total user time, in seconds.
Fields for JCP Memory Usage	
Maximum Memory	The maximum amount of memory, in kibibytes (KiBs).
Used Memory	The total amount of used memory, in kibibytes (KiBs).
Fields for JCP Internal IP Addresses	

Table 13: show system visibility jcp Output Fields (continued)

Field Name	Field Description
Interface	The name of the interface.
Address	The IP address of the interface.
Fields for JCP Interfaces	
Interface	The name of the interface.
Type	The type of the interface.
Source	The connectivity source.
Model	The connectivity model.
MAC	The MAC address of the interface.
Fields for JCP Interfaces Statistics	
Port	The name of the port.
Rcvd Bytes	The number of bytes received.
Rcvd Packets	The number of packets received.
Rcvd Error	The number of errors received.
Rcvd Drop	The number of drops received.
Trxd Bytes	The number of bytes transmitted.
Trxd Packets	The number of packets transmitted.
Trxd Error	The number of errors transmitted.
Trxd Drop	The number of drops transmitted.
Fields for JCP Disk Information	
Disk	The type of disk.
File	The path to the disk.

Sample Output

show system visibility jcp

```
user@jdm> show system visibility jcp
```

```
JCP CPU Statistics (Time in sec)
------------------------------
CPU Time:    21435526
System Time: 4981660
User Time:   780770

JCP Memory Usage
----------------
Maximum Memory (KiB): 200089
Used Memory (KiB):    200089

JCP Internal IP Addresses
-------------------------
Interface: em2.32768
Address  : 192.168.1.2

JCP Interfaces
--------------------------------------------------------------
Interface Type      Source       Model      MAC
--------- --------- ------------ ---------- -----------------
vnet1     bridge    virbr0       e1000      52:54:00:7b:28:91
iri1      bridge    ctrlbr0      e1000      52:54:00:db:ba:c2
vnet3     bridge    eth0br       e1000      52:54:00:c6:8e:e9
vjunos0_em1 bridge  ovs-sys-br   e1000      52:54:00:e4:4d:2e

JCP Interfaces Statistics
-------------------------------------------------------------------------------------------------------
Port     Rcvd Bytes   Rcvd Packets Rcvd Error Rcvd Drop Trxd Bytes   Trxd Packets
 Trxd Error Trxd Drop
--------- ------------ ------------ ---------- --------- ------------ ------------
 ---------- ---------
vnet1     107535558    1150988      0          0         57984280     923183
   0          0
iri1      384836814    5242053      0          0         349833594    5250766
   0          0
vnet3     1341546383   6654730      0          125       1169070      27835
   0          0
vjunos0_em1 4656       58           0          0         0            0
   0          0

JCP Disk Information
--------------------------------------------------------------------------
Disk        File
----------- ------------------------------------------------------------
hda         /junos/images/0/vjunos.img
hdb         /junos/images/0/vjunos-data.img
hdc         /junos/images/0/vjunos-config.img
hdd         /junos/images/0/vjunos-platform.img
```

show system visibility jdm

Syntax show system visibility jdm

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Display details such as uptime, number of tasks, CPU statistics, disk usage, disk I/O statistics, memory usage, the list of network interfaces, and internal IP address for JDM container.

Required Privilege Level view

Related Documentation

- show system visibility cpu on page 80
- show system visibility host on page 83
- show system visibility jcp on page 89
- show system visibility memory on page 96
- show system visibility network on page 98
- show system visibility storage on page 101
- show system visibility vnf on page 104

List of Sample Output show system visibility jdm on page 94

Output Fields Table 14 on page 92 lists the output fields for the **show system visibility jdm** command. Output fields are listed in the approximate order in which they appear.

Table 14: show system visibility jdm Output Fields

Field Name	Field Description
Fields for JDM CPU Statistics	
User Time	The amount of user time, in seconds.
System Time	The amount of system time, in seconds.
Idle Time	The amount of time spent in idle mode, in seconds.
I/O Wait Time	The amount of time spent waiting for input/output (I/O) operations, in seconds.
Nice Time	The amount of spent nice time, in seconds.
Interrupt Service Time	The amount of interrupt service time, in seconds.

Table 14: show system visibility jdm Output Fields (continued)

Field Name	Field Description
Fields for JDM Disk Usage Information	
Total	The total amount of disk usage space, in mebibytes (MiB).
Used	The amount of used disk usage space, in mebibytes (MiB).
Free	The amount of free disk usage space, in mebibytes (MiB).
Percentage Used	The percentage of used disk space.
Fields for JDM Disk I/O Information	
Read Count	The number of times the disk has been read.
Write Count	The number of times a write operation has happened on the disk.
Read Bytes	The number of bytes used in read operations on the disk.
Write Bytes	The number of bytes used in write operations on the disk.
Read Time	The amount of time the disk has been read, in milliseconds.
Write Time	The amount of time write operations have been performed on the disk, in milliseconds.
Fields for JDM Memory Information	
Maximum Memory	The maximum virtual memory used, in kilobytes (KiB).
Used	The amount of used virtual memory, in kilobytes (KiB).
Fields for List of JDM Interfaces	
Interface	The name of the interface.
Type	The type of interface.
Source	The connectivity source.
MAC	The MAC address of the interface.
Field for JDM Uptime	
Uptime	The time the JDM has been operational.
Fields for JDM Tasks	
Total	The total number of tasks.
Running	The total number of tasks running.
Sleeping	The total number of tasks in sleeping state.

Table 14: show system visibility jdm Output Fields (continued)

Field Name	Field Description
Stopped	The total number of tasks that are stopped.
Zombie	The number of zombie processes.
Fields for JDM Internal IP Addresses	
Interface	The ID of the interface.
Address	The IP address of the interface.

Sample Output

show system visibility jdm

```
user@jdm> show system visibility jdm
JDM CPU Statistics (Time in sec)
----------------------------
User Time:              451541
System Time:            14381
Idle Time:              4492695
I/O Wait Time:          581
Nice Time:              0
Interrupt Service Time: 0

JDM Disk Usage Information
--------------------------
Total (MiB):      9951
Used  (MiB):      3986
Free  (MiB):      5436
Percentage Used:  40.1

JDM Disk I/O Information
------------------------
 Read Count: 304517
Write Count: 1180759
 Read Bytes: 4200104448
Write Bytes: 16527707648
  Read Time: 45534
 Write Time: 1185936

JDM Memory Information
----------------------

Maximum  (KiB): 1048576
Used   (KiB): 264756

JDM Interfaces
-------------------------------------------------
Interface Type      Source       MAC
--------- --------- ------------ -----------------
vnet0     bridge    eth0br       00:a0:c9:00:00:04
vnet2     bridge    virbr0       52:54:00:20:f7:9d
vnet4     bridge    ctrlbr0      52:54:00:57:8d:ef
```

```
jdm_jsxe0 bridge    ovs-sys-br   52:54:00:e4:f1:3f
jdm_phc   bridge    ovs-sys-br   52:54:00:47:54:47

JDM Uptime
----------
Uptime: 4 days, 18:56:26.180000

JDM Tasks
---------
Total:    44
Running:  1
Sleeping: 42
Stopped:  0
Zombie:   1

JDM Internal IP Addresses
-------------------------
Interface: bme1
Address  : 192.168.1.254
```

show system visibility memory

Syntax

```
show system visibility memory
```

Release Information

Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description

Display the details about virtual memory and shared memory for a disaggregated Junos OS platform.

Required Privilege Level

view

Related Documentation

- show system visibility cpu on page 80
- show system visibility host on page 83
- show system visibility jcp on page 89
- show system visibility jdm on page 92
- show system visibility network on page 98
- show system visibility storage on page 101
- show system visibility vnf on page 104

List of Sample Output

show system visibility memory on page 97

Output Fields

Table 15 on page 96 lists the output fields for the **show system visibility memory** command. Output fields are listed in the approximate order in which they appear.

Table 15: show system visibility memory Output Fields

Field Name	Field Description
Fields for Memory Information—Virtual Memory	
Total	The total amount of available virtual memory, in kibibytes (KiBs).
Used	The total amount of used virtual memory, in kibibytes (KiBs).
Available	The total amount of available virtual memory, in kibibytes (KiBs).
Free	The total amount of free virtual memory, in kibibytes (KiBs).
Percent Used	The percentage of buffer virtual memory used.

Sample Output

show system visibility memory

```
user@jdm> show system visibility memory
Memory Information
------------------
Virtual Memory:
---------------
Total  (KiB): 15949116
Used   (KiB): 5545380
Available (KiB): 10440500
Free   (KiB): 10403736
Percent Used: 28.6

Huge Pages:
------------
Total 1GiB Huge Pages:      1
Free 1GiB Huge Pages:       1
Configured 1GiB Huge Pages: 0
Total 2MiB Huge Pages:      2716
Free 2MiB Huge Pages:       31
Configured 2MiB Huge Pages: 0
```

show system visibility network

Syntax show system visibility network

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Display details such as the list of MAC addresses assigned to VNF interfaces, the list of internal IP addresses for VNFs, the list of VFs used by VNFs, and the list of VNF interfaces for a disaggregated Junos OS platform.

Required Privilege Level view

Related Documentation

- show system visibility cpu on page 80
- show system visibility host on page 83
- show system visibility jcp on page 89
- show system visibility jdm on page 92
- show system visibility memory on page 96
- show system visibility storage on page 101
- show system visibility vnf on page 104

List of Sample Output show system visibility network on page 99

Output Fields Table 16 on page 98 lists the output fields for the **show system visibility network** command. Output fields are listed in the approximate order in which they appear.

Table 16: show system visibility network Output Fields

Field Name	Field Description
Fields for List of VNF MAC Addresses	
VNF	The name of the VNF.
MAC	The MAC address of the VNF.
Fields for List of VNF Internal IP Addresses	
VNF	The name of the VNF.
IP	The IP address of the VNF.
Fields for List of VNF Virtual Functions	
VNF	The name of the VNF.

Table 16: show system visibility network Output Fields (continued)

Field Name	Field Description
PF	The names of the Physical Functions available.
VF	The names of the Virtual Functions available for each Physical Function.
Fields for List of Free Virtual Functions	
PF	The names of the Physical Functions available.
VF	The names of the Virtual Functions available for each Physical Function.
Fields for List of VNF Interfaces	
VNF	The name of the VNF.
Interface	The name of the interface.
Type	The type of interface.
Source	The connectivity source.
Model	The connectivity model.
MAC	The MAC address of the VNF.

Sample Output

show system visibility network

```
user@jdm> show system visibility network
VNF MAC Addresses
------------------------------------------------------------
VNF                                     MAC
---------------------------------------- -----------------
vnf1_ethdef0                            84:C1:C1:A3:39:15
vnf1_ethdef1                            84:C1:C1:A3:39:16
vnf1_eth2                               84:C1:C1:A3:39:17

VNF Internal IP Addresses
------------------------------------------------------------
VNF                                     IP
---------------------------------------- ---------------
vnf1                                    192.168.1.100

VNF Virtual Functions
------------------------------------------------------------------------
VNF                                     PF        VF
---------------------------------------- --------- ------------
sxe0vf9_vfdef0                          hsxe0     0000:03:12:2
sxe0vf10_vfdef0                         hsxe0     0000:03:12:4
sxe0vf11_vfdef0                         hsxe0     0000:03:12:6
sxe0vf12_vfdef0                         hsxe0     0000:03:13:0
```

```
sxe0vf13_vfdef0                          hsxe0     0000:03:13:2
sxe1vf13_vfdef0                          hsxe1     0000:03:13:3
sxe1vf14_vfdef0                          hsxe1     0000:03:13:5

Free Virtual Functions
----------------------
PF        VF
--------- ------------
hsxe0     0000:03:10:0
hsxe0     0000:03:10:2
hsxe0     0000:03:10:4
hsxe0     0000:03:10:6
hsxe0     0000:03:11:0
hsxe0     0000:03:11:2
hsxe0     0000:03:11:4
hsxe0     0000:03:11:6
hsxe0     0000:03:12:0
hsxe0     0000:03:13:4
hsxe1     0000:03:10:1
hsxe1     0000:03:10:3
hsxe1     0000:03:10:5
hsxe1     0000:03:10:7
hsxe1     0000:03:11:1
hsxe1     0000:03:11:3
hsxe1     0000:03:11:5
hsxe1     0000:03:11:7
hsxe1     0000:03:12:1
hsxe1     0000:03:12:3
hsxe1     0000:03:12:5
hsxe1     0000:03:12:7
hsxe1     0000:03:13:1

VNF Interfaces
-------------------------------------------------------------------------------
VNF                  Interface Type      Source       Model      MAC
-------------------- --------- --------- ------------ ---------- -----------------
vnf1                vnet5     network   default      virtio     84:c1:c1:a3:39:15
vnf1                vnet6     bridge    eth0br       virtio     84:c1:c1:a3:39:16
vnf1                --        vhostuser --           virtio     84:c1:c1:a3:39:17
```

show system visibility storage

Syntax show system visibility storage

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Display details such as the list of disk partitions, the list of per disk I/O statistics, and the list of VNF disks for a disaggregated Junos OS platform.

Required Privilege Level view

Related Documentation

- show system visibility cpu on page 80
- show system visibility host on page 83
- show system visibility jcp on page 89
- show system visibility jdm on page 92
- show system visibility memory on page 96
- show system visibility network on page 98
- show system visibility vnf on page 104
- show system visibility cpu on page 80
- show system visibility host on page 83
- show system visibility jcp on page 89
- show system visibility jdm on page 92
- show system visibility memory on page 96
- show system visibility network on page 98
- show system visibility vnf on page 104

List of Sample Output show system visibility storage on page 102

Output Fields Table 17 on page 101 lists the output fields for the **show system visibility storage** command. Output fields are listed in the approximate order in which they appear.

Table 17: show system visibility storage Output Fields

Field Name	Field Description
Fields for Disk Usage Information	
Disk	The name of the disk.

Table 17: show system visibility storage Output Fields (continued)

Field Name	Field Description
Total	The amount of total disk space, in mebibytes (MiB).
Used	The amount of used disk space, in mebibytes (MiB).
Free	The amount of free disk space, in mebibytes (MiB).
Percentage Used	The percentage of used disk space in the disk.
Fields for Disk I/O Information	
Disk	The name of the disk.
Read Count	The number of times the disk has been read.
Write Count	The number of times a write operation has happened on the disk.
Read Bytes	The number of bytes used in read operations on the disk.
Write Bytes	The number of bytes used in write operations on the disk.
Read Time	The amount of time the disk has been read, in milliseconds.
Write Time	The amount of time write operations have been performed on the disk, in milliseconds.
Fields for the VNF Disk Information	
VNF	The name of the VNF.
Disk	The name of the disk.
File	The path to the disk.

Sample Output

show system visibility storage

```
user@jdm> show system visibility storage
Disk Usage Information
-------------------------------------------------------------------------------
Disk                                    Total (MiB) Used (MiB)  Free (MiB)
% Used
---------------------------------------- ----------- ----------- -----------
------
/dev/sda4                               1409        904         416
64.0
/dev/sda5                               5639        320         5009
5.0
/dev/sda2                               976         705         271
72.0
```

```
/dev/sda1                                  189         50          139
26.0
/dev/sda3                                  1           0           1
1.0
/dev/mapper/vg0_vjunos-lv_junos            9951        3889        5534
39.0
/dev/mapper/vg0_vjunos-lv_var_third_party  159317      6888        144313
4.0

Disk I/O Information
-------------------------------------------------------------------------------------------------
Disk                Read Count  Write Count Read Bytes      Write Bytes     Read
 time   Write Time
------------------- ----------- ----------- --------------- ---------------
----------- -----------
sdb3                64          0           249856          0               49
        0
sdb2                201         0           1794048         0               105
        0
sdb1                235         0           1853440         0               118
        0
sdb6                282         0           2209792         0               90
        0
sdb5                201         0           1777664         0               102
        0
sdb4                201         0           1777664         0               106
        0
sda6                10070       555288      713788928       5525233664      4033
       417990
sda7                202         0           1908736         0               68
        0
sda4                4034        1336        188470272       541732864       2114
       3384
sda5                1372        304469      60498944        8484728832      470
        407722
sda2                2639        1           285951488       512             701
        0
sda3                69          4           251904          1941504         47
        7
sda1                406         0           19010560        0               145
        0
dm-2                9524        420264      569390080       3809955840      5284
       447641
dm-3                2236        383054      144124416       1715277824      738
        251792
dm-0                230         0           942080          0               131
        0
dm-1                230         0           942080          0               113
        0

VNF Disk Information
-------------------------------------------------------------------------------------------------
VNF                          Disk        File
---------------------------- -----------
-----------------------------------------------------------------
vnf1                         vda
/var/third-party/images/media-vsrx-vmdisk-15.1X49-D40.6.qcow2.md5
```

show system visibility vnf

Syntax show system visibility vnf *vnf name*

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description If a VNF name is not specified, display the details of the VNFs present on the system. Details include VNF memory usage, CPU statistics, the list of network interfaces, the list of disk files, per disk usage, per port I/O statistics, and media information, which includes details about CD-ROM and USB storage devices.

If a VNF name is specified, display the details of the VNF. Details include VNF memory usage, CPU statistics, the list of network interfaces, the list of disk files, per disk usage, per port I/O statistics, and media information, which includes details about CD-ROM and USB storage devices.

Required Privilege Level view

Related Documentation

- show system visibility cpu on page 80
- show system visibility host on page 83
- show system visibility jcp on page 89
- show system visibility jdm on page 92
- show system visibility memory on page 96
- show system visibility network on page 98
- show system visibility storage on page 101

List of Sample Output show system visibility vnf on page 107

Output Fields Table 18 on page 104 lists the output fields for the **show system visibility vnf** command. Output fields are listed in the approximate order in which they appear.

Table 18: show system visibility vnf Output Fields

Field Name	Field Description
Fields for List of VNFs	
ID	ID of the VNF.
Name	Name of the VNF.
State	State of the VNF.

Table 18: show system visibility vnf Output Fields (continued)

Field Name	Field Description
Fields for VNF Memory Usage	
Name	Name of the VNF.
Maximum Memory	The maximum amount of memory, in kibibytes (KiBs).
Used Memory	The total amount of used memory, in kibibytes (KiBs).
Used 1G Hugepages	The total number of 1G hugepages used.
Used 2M Hugepages	The total number of 2M hugepages used.
Fields for VNF CPU Stats	
Name	Name of the VNF.
CPU Time	The total CPU time, in seconds.
System Time	The amount of system CPU time, in seconds.
User Time	The amount of user CPU time, in seconds.
Fields for List of VNF MAC Addresses	
VNF	Names of the VNFs.
MAC	MAC addresses of the VNFs.
Fields for List of VNF Internal IP Addresses	
VNF	Names of the VNFs.
IP	Internal IP addresses of the VNFs.
Fields for List of Virtual Functions per VNF	
VNF	Names of the VNFs.
PF	The names of the Physical Functions available.
VF	The names of the Virtual Functions available for each Physical Function.
Fields for the VNF Interfaces	
VNF	The name of the VNF.
Interface	The name of the interface.
Type	The type of interface.
Source	The connectivity source.

Table 18: show system visibility vnf Output Fields (continued)

Field Name	Field Description
Model	The connectivity model.
MAC	The MAC address of the VNF.
Fields for List of VNF Disk Information	
VNF	The name of the VNF.
Disk	The name of the disk.
File	The path to the disk.
Fields for List of VNF Disk Usage	
VNF	The name of the VNF.
Disk	The name of the disk.
Read Requests	The number of times a read operation has happened on the disk.
Bytes Read	The number of read bytes on the disk.
Write Requests	The number of times a write operation has happened on the disk.
Bytes Written	The number of bytes written on the disk.
Fields for List of VNF Port Statistics	
VNF	The name of the VNF.
Port	The name of the port.
Rcvd Bytes	The number of bytes received.
Rcvd Packets	The number of packets received.
Rcvd Error	The number of errors received.
Rcvd Drop	The number of drops received.
Trxd Bytes	The number of bytes transferred.
Trxd Packets	The number of packets transferred.
Trxd Error	The number of errors transferred.
Trxd Drop	The number of drops transferred.

Sample Output

show system visibility vnf

```
user@jdm> show system visibility vnf
List of VNFs
-------------------------------------------------------------
ID   Name                                    State
---- --------------------------------------- ---------------
4    vnf1                                    Running

VNF Memory Usage
-------------------------------------------------------------------------------
Name                                 Maximum Memory (KiB) Used Memory (KiB)
Used 1G Hugepages Used 2M Hugepages
------------------------------------ -------------------- -----------------
---------------- -----------------
vnf1                                 104857               104857           0
               256

VNF CPU Statistics (Time in ms)
-------------------------------------------------------------------------------
Name                                 CPU Time     System Time  User Time
------------------------------------ ------------ ------------ ------------
vnf1                                 617598       257020       53510

VNF MAC Addresses
-------------------------------------------------------------
VNF                                     MAC
--------------------------------------- -----------------
vnf1_ethdef0                            84:C1:C1:A3:39:15
vnf1_ethdef1                            84:C1:C1:A3:39:16
vnf1_eth2                               84:C1:C1:A3:39:17

VNF Internal IP Addresses
-------------------------------------------------------------
VNF                                     IP
--------------------------------------- ---------------
vnf1                                    192.168.1.100

VNF Virtual Functions
-------------------------------------------------------------------
VNF                                     PF        VF
--------------------------------------- --------- ------------
sxe0vf9_vfdef0                          hsxe0     0000:03:12:2
sxe0vf10_vfdef0                         hsxe0     0000:03:12:4
sxe0vf11_vfdef0                         hsxe0     0000:03:12:6
sxe0vf12_vfdef0                         hsxe0     0000:03:13:0
sxe0vf13_vfdef0                         hsxe0     0000:03:13:2
sxe1vf13_vfdef0                         hsxe1     0000:03:13:3
sxe1vf14_vfdef0                         hsxe1     0000:03:13:5

VNF Interfaces
-------------------------------------------------------------------------------------------
VNF                  Interface Type      Source       Model      MAC
-------------------- --------- --------- ------------ ---------- -----------------
vnf1                vnet5     network   default      virtio     84:c1:c1:a3:39:15
vnf1                vnet6     bridge    eth0br       virtio     84:c1:c1:a3:39:16
vnf1                --        vhostuser --           virtio     84:c1:c1:a3:39:17
```

```
VNF Disk Information
-------------------------------------------------------------------------------------------------
VNF                           Disk        File
------------------------------ -----------
-----------------------------------------------------------------
vnf1                          vda
/var/third-party/images/media-vsrx-vmdisk-15.1X49-D40.6.qcow2.md5

VNF Disk Usage
-------------------------------------------------------------------------------
VNF                  Disk      Read Req   Read Bytes   Write Req  Write Bytes
-------------------- --------- ---------- ------------ ---------- ------------
vnf1                 vda       0          0            0          0

VNF Port Statistics
_______________________________________________________________________________________________
VNF                  Port      Rcvd Bytes   Rcvd Packets Rcvd Error Rcvd Drop
Trxd Bytes   Trxd Packets Trxd Error Trxd Drop
-------------------- --------- ------------ ------------ ---------- ---------
------------ ------------ ---------- ---------
vnf1                 vnet5     0            0            0          252654    0
           0            0          0
vnf1                 vnet6     0            0            0          8893085   0
           0            0          0
```

PART 4

Virtual Network Functions

CHAPTER 5

Virtual Network Functions

Understanding Virtual Network Functions

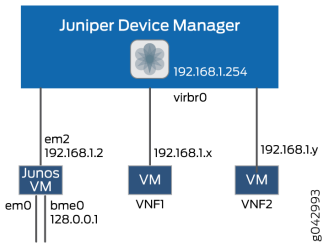
Virtualized network functions (VNFs) include all virtual entities that can be launched and managed from the Juniper Device Manager (JDM). Currently, virtual machines (VMs) are the only VNF type that is supported.

There are several components in a JDM environment:

- **JDM**—Manages the life cycle for all service VMs. JDM also provides a CLI with configuration persistence or the ability to use NETCONF for scripting and automation.
- **Primary Junos OS VM**—A *system VM* that is the primary virtual device. This VM is always present when the system is running.
- **Other Junos OS VMs**—These VMs are *service VMs* and are activated dynamically by an external controller. A typical example of this type of VM is a vSRX instance.
- **Third-party VNFs**—JDM supports the creation and management of third-party VMs such as Ubuntu Linux VMs.

The JDM architecture provides an internal network that connects all VMs to the JDM as shown in Figure 12 on page 112.

Figure 12: Network Connections Between JDM and the VMs



The JDM can reach any VNF using an internal network (192.168.1.0/24).

A VNF can own or share management ports and NIC ports in the system.

All VMs run in isolation and a state change in one VM does not affect another VM. When the system restarts, the service VMs are brought online as specified in the persistent configuration file. When you gracefully shut down the system, all VMs including the Junos VMs are shut down.

Table 19 on page 112 provides a glossary of commonly used VNF acronyms and terms.

Table 19: VNF Glossary

Term	Definition
JCP	Junos Control Plane (also known as the primary Junos OS VM)
JDM	Juniper Device Manager
NFV	Network Functions Virtualization
VM	Virtual Machine
VNF	Virtualized Network Function

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- Disaggregated Junos OS VMs on page 6

Managing the VNF Life Cycle

You can use the JDM CLI to manage the VNF. Additionally, *libvirt* software offers extensive virtualization features. To ensure that you are not limited by the CLI, JDM provides an option to operate VNF using an XML descriptor file. Network Configuration Protocol (NETCONF) supports all VNF operations. Multiple VNFs can co-exist in a system and you can configure multiple VNFs using either an XML descriptor file or an image.

NOTE: Ensure that VNF resources that are specified in the XML descriptor file do not exceed the available system resources.

This topic covers the life-cycle management of a VNF.

Planning Resources for a VNF

Purpose Before launching a VNF, it is important to check the system inventory and confirm that the resources required by the VNF are available. The VNF must be designed and configured properly so that its resource requirements do not exceed the available capacity of the system.

NOTE:

- The output of the **show system inventory** command displays only the current snapshot of system resource usage. When you start a VNF, the resource usage might be less than what was available when you installed the VNF package.
- Before starting a VNF, you must check the system resource usage.

NOTE: Some of the physical CPUs are reserved by the system. Except for the following physical CPUs, all others are available for user-defined VNFs:

Table 20 on page 114 provides the list of physical CPUs that are reserved for NFX250-LS1.

Table 20: Physical CPU Allocation for NFX250-LS1

CPU Core	Allocation
0	Host, JDM, and JCP
4	Host bridge
7	IPSec

Table 21 on page 114 provides the list of physical CPUs that are reserved for NFX250.

Table 21: Physical CPU Allocation for NFX250

CPU Core	Allocation
0	Host, JDM, and JCP
6	Host bridge
7	IPSec

For more information, see the following:

- show system inventory hardware cpu
- show system inventory hardware memory
- show system inventory hardware network
- show system inventory hardware storage
- show system inventory software vnf

Managing the VNF Image

To load a VNF image on the device from a remote location, use the **file-copy** command. Alternatively, you can use the NETCONF command **file-put**, to load a VNF image.

NOTE: You must save the VNF image in the */var/third-party/images* directory.

Preparing the Bootstrap Configuration

You can bootstrap a VNF by attaching either a CD or a USB storage device that contains a bootstrap-config ISO file.

A bootstrap configuration file must contain an initial configuration that allows the VNF to be accessible from an external controller, and accepts SSH, HTTP, or HTTPS connections from an external controller for further runtime configurations.

An ISO disk image must be created offline for the bootstrap configuration file as follows:

```
user@jdm>request genisoimage bootstrap-config-filename iso-filename
```

Launching a VNF

You can launch a VNF by configuring the VNF name, and specifying either the path to an XML descriptor file or to an image.

While launching a VNF with an image, two VNF interfaces are added by default. These interfaces are required for management and internal network. For those two interfaces, the target Peripheral Component Interconnect (PCI) addresses, such as 0000:00:03:0 and 0000:00:04:0 are reserved.

To launch a VNF using an XML descriptor file:

```
user@jdm#  set virtual-network-functions vnf-name init-descriptor file-path
user@jdm#  commit
```

To launch a VNF using an image:

```
user@jdm#  set virtual-network-functions vnf-name image file-path
user@jdm#  commit
```

To specify a UUID for the VNF:

```
user@jdm#  set virtual-network-functions vnf-name [uuid vnf-uuid]
```

uuid is an optional parameter, and it is recommended to allow the system to allocate a UUID for the VNF.

NOTE:

- You cannot change the init-descriptor or image configuration after saving and committing the init-descriptor and image configuration. To change the init-descriptor or image for a VNF, you must delete and create a VNF again.
- Commit checks are applicable only for VNF configurations that are based on image specification through JDM CLI, and not for VNF configurations that are based on init-descriptor XML file.

NOTE: For creating VNFs using image files, ensure the following:

- You must use unique files for image, disk, USB that are used within a VNF or across VNFs except for an iso9660 type file, which can be attached to mutiple VNFs.
- A file specified as image in raw format should be a block device with a partition table and a boot partition.
- A file specified as image in qcow2 format should be a valid qcow2 file.

Allocating Resources for a VNF

This topic covers the process of allocating various resources to a VNF.

Specifying CPU for VNF

To specify the number of virtual CPUs that are required for a VNF, type the following command:

```
user@jdm#  set virtual-network-functions vnf-name virtual-cpu count 1-4
```

To pin a virtual CPU to a physical CPU, type the following command:

```
user@jdm#  set virtual-network-functions vnf-name virtual-cpu vcpu-number physical-cpu
pcpu-number
```

The physical CPU number can either be a number or a range. By default, a VNF is allocated with one virtual CPU that is not pinned to any physical CPU.

NOTE: You cannot change the CPU configuration of a VNF when the VNF is in running state. Restart the VNF for changes to take effect.

To enable hardware-virtualization or hardware-acceleration for VNF CPUs, type the following command:

```
user@jdm#  set virtual-network-functions vnf-name virtual-cpu features hardware-virtualization
```

Allocating Memory for a VNF

To specify the maximum primary memory that the VNF can use, enter the following command:

```
user@jdm#  set virtual-network-functions vnf-name memory size size
```

By default, 1 GB of memory is allocated to a VNF.

NOTE: You cannot change the memory configuration of a VNF if the VNF is in running state. Restart the VNF for changes to take effect.

To allocate hugepages for a VNF, type the following command:

```
user@jdm#  set virtual-network-functions vnf-name memory features hugepages [page-size
page-size]
```

page-size is an optional parameter. Possible values are 1024 for a page size of 1 GB and 2 for a page size of 2 MB. Default value is 1024 hugepages.

NOTE: Configuring hugepages is recommended only if the enhanced orchestration mode is enabled. If the enhanced orchestration mode is disabled and if VNF requires hugepages, the VNF XML descriptor file should contain the XML tag with hugepages configuration.

NOTE: For VNFs that are created using image files, there is a maximum limit of the total memory that can be configured for all user-defined VNFs including memory based on hugepages and memory not based on hugepages. The maximum limit is 5GB for all SKUs with 8GB RAM and 13GB for all SKUs with 16GB RAM.

Configuring VNF Storage Devices

To add a virtual CD or to update the source file of a virtual CD, enter the following command:

```
user@jdm# set virtual-network-functions vnf-name storage device-name type cdrom source file file-name
```

To add a virtual USB storage device, enter the following command:

```
user@jdm# set virtual-network-functions vnf-name storage device-name type usb source file file-name
```

To attach an additional hard disk, enter the following command:

```
user@jdm# set virtual-network-functions vnf-name storage device-name type disk [bus-type virtio | ide] [file-type raw | qcow2] source file file-name
```

To delete a virtual CD, USB storage device, or a hard disk from the VNF, enter the following command:

```
user@jdm# delete virtual-network-functions vnf-name storage device-name
```

NOTE:

- After attaching or detaching a CD from a VNF, you must restart the device for changes to take effect. The CD detach operation fails if the device is in use within the VNF.
- VNF supports one virtual CD, one virtual USB storage device, and multiple virtual hard disks.
- You can update the source file in a CD or USB storage device while the VNF is in running state.
- You must save the source file in the /var/third-party directory and the file must have read and write permission for all users.

NOTE: For VNFs created using image files, ensure the following:

- A file specified as a hard disk in raw format should be a block device with a partition table.
- A file specified as a hard disk in qcow2 format should be a valid qcow2 file.
- A file specified as USB should be a block device with a partition table, or an iso9660 type file.
- A file specified as CD-ROM should be a block device of type iso9660.
- If a VNF has an image specified with bus-type=ide, it should not have any device attached with name hda.
- If a VNF has an image specified with bus-type=virtio, then it should not have any device attached with name vda.

Configuring VNF Interfaces and VLANs

You can create a VNF interface and attach it to a physical NIC port, management interface, or VLANs.

1. To attach a VNF interface to a physical interface by using the SR-IOV virtual function:

   `user@jdm#` **set virtual-network-functions** ***vnf-name*** **interfaces** ***interface-name*** **mapping** ***physical-interface-name*** **virtual-function [vlan-id** ***vlan-id*****]**

   vlan-id is optional and it is the port VLAN ID.

2. To create VLAN:

   `user@jdm#` **set host-name vlan** ***vlan-name***

3. To attach a VNF interface to a VLAN:

```
user@jdm#  set virtual-network-functions vnf-name interfaces interface-name mapping vlan
members list-of-vlans [mode trunk|access]
```

NOTE:

- The interfaces attached to the VNF are persistent across VNF restarts.
- If the VNF supports hot-plugging, you can attach the interfaces when the VNF is in running state. Otherwise, add the interfaces, and then restart the VNF.
- To map interfaces to VLAN, you must enable the memory features hugepages command option.
- You cannot change the mapping of VNF interface when the VNF is in running state.

4. To map virtual interfaces with physical interfaces:

   ```
   user@jdm#  set virtual-network-functions vnf-name interfaces interface-name mapping
   peer-interfaces
   ```

   Mapping of virtual interfaces and physical interfaces (ge-0/0/n and xe-0/0/n) makes sure that the state of the virtual interface matches with the state of the physical interface to which it is mapped. For example, if a physical interface is down and virtual interface is up, the virtual interface will be brought down within 5 seconds of detection. One or more virtual interface can be mapped to one or more physical interfaces.

5. To connect VNF interfaces to the internal management network:

   ```
   user@jdm#  set virtual-network-functions vnf-name interfaces interface-name management
   internal
   user@jdm#  set virtual-network-functions vnf-name interfaces interface-name management
   out-of-band
   ```

   Any of the VNF interfaces including eth0 and eth1 can have internal or out-of-band attribute management. However, only one VNF interface out of all interfaces that are connected can have either out-of-band-management or internal-management. You cannot specify both the attribute values to the same VNF interface. For example, eth5 can have management internal while eth0 can have management out-of-band.

6. To specify the target PCI address for a VNF interface:

   ```
   user@jdm#  set virtual-network-functions vnf-name interfaces interface-name pci-address
   target-pci-address
   ```

   You can use the target PCI address to rename or reorganize interfaces within the VNF.

   For example, a Linux-based VNF can use udev rules within the VNF to name the interface based on the PCI address.

NOTE:

- The target PCI-address string should be in the following format:

  0000:00:<slot:>:0, which are the values for domain:bus:slot:function. The slot should be different for each VNF interface. The values for domain, bus, and function should be zero.

- You cannot change the target PCI-address of VNF interface when the VNF is in running state.

7. To delete a VNF interface:

```
user@jdm# delete virtual-network-functions vnf-name interfaces interface-name
user@jdm# commit
```

NOTE:

- To delete an interface, you must stop the VNF, delete the interface, and start the VNF.
- After attaching or detaching a virtual function, you must restart the VNF for changes to take effect.
- eth0 and eth1 are reserved for default VNF interfaces that are connected to the internal network and out-of-band management network. Therefore, the configurable VNF interface names start from eth2.
- Within a VNF, the interface names can be different, based on guest OS naming convention. VNF interfaces that are configured in JDM might not appear in the same order within the VNF.
- You must use the target PCI addresses to map to the VNF interfaces that are configured in JDM and name them accordingly.

Managing VNF States

By default, the VNF is autostarted on committing the VNF config.

1. To disable an autostart of a VNF on a VNF config commit:

```
user@jdm# set virtual-network-functions vnf-name no-autostart
```

2. To manually start a VNF:

```
user@jdm> request virtual-network-functions vnf-name start
```

3. To stop a VNF:

```
user@jdm> request virtual-network-functions vnf-name stop
```

4. To restart a VNF:

```
user@jdm> request virtual-network-functions vnf-name restart
```

Managing VNF MAC Addresses

VNF interfaces that are defined, either using a CLI or specified in an init-descriptor XML file, are assigned a globally-unique and persistent MAC address. A common pool of 64 MAC addresses is used to assign MAC addresses. You can configure a MAC address other than that available in the common pool, and this address will not be overwritten.

1. To configure a specific MAC address for a VNF interface:

   ```
   user@jdm# set virtual-network-functions vnf-name interfaces interface-name mac-address mac-address
   ```

2. To delete the MAC address configuration of a VNF interface:

   ```
   user@jdm# delete virtual-network-functions vnf-name interfaces interface-name mac-address mac-address
   ```

NOTE:

- To delete or modify the MAC address of a VNF interface, you must stop the VNF, make the necessary changes, and then start the VNF.
- The MAC address specified for a VNF interface can be either a system MAC address or a user-defined MAC address.
- The MAC address specified from the system MAC address pool must be unique for VNF interfaces.

Managing MTU

The maximum transmission unit (MTU) is the largest data unit that can be forwarded without fragmentation. You can configure either 1500 bytes or 2048 bytes as the MTU size. The default MTU value is 1500 bytes.

NOTE: MTU configuration is supported only on VLAN interfaces.

1. To configure MTU on a VNF interface:

   ```
   user@jdm# set virtual-network-functions vnf-name interfaces interface-name mtu size
   ```

   NOTE: You must restart the VNF after configuring MTU if the VNF does not support hot-plugging functionality.

2. To delete MTU of a VNF interface:

```
user@jdm# delete virtual-network-functions vnf-name interfaces interface-name mtu
```

NOTE: After the deletion of MTU, the MTU of VNF interface is reset to 1500 bytes.

NOTE:

- MTU size can be either 1500 bytes or 2048 bytes.
- The maximum number of VLAN interfaces on the OVS that can be configured in the system is limited to 25.
- The maximum size of the MTU for a VNF interface is 2048 bytes.

Configuring Cross-connect

The **Cross-connect** feature enables traffic switching between any two OVS interfaces such as VNF interfaces or physical interfaces such as hsxe0 and hsxe1 that are connected to the OVS. You can bidirectionally switch either all traffic or traffic belonging to a particular VLAN between any two OVS interfaces.

NOTE: This feature does not support unidirectional traffic flow.

The **Cross-connect** feature supports the following:

- Unconditional cross-connect between two VNF interfaces for all network traffic.
- VLAN-based traffic forwarding between VNF interfaces support the following functions:
  - Provides an option to switch traffic based on a VLAN ID.
  - Supports network traffic flow from trunk to access port.
  - Supports network traffic flow from access to trunk port.
  - Supports VLAN PUSH, POP, and SWAP operations.

To configure cross-connect:

1. Configure VLANs:

```
[edit]
user@jdm#set host-os vlans vlan-name vlan-id vlan-id
user@jdm#set host-os vlans v10 vlan-id 10
user@jdm#set host-os vlans v20 vlan-id 20
user@jdm#set host-os vlans v30 vlan-id 30
user@jdm#set host-os vlans v40 vlan-id 40
```

2. Configure VNFs:

```
user@jdm# set virtual-network-functions vnf-name interfaces interface-name mapping
  vlan members list
user@jdm# set virtual-network-functions vnf1 interfaces eth1 mapping vlan members
  v10
user@jdm# set virtual-network-functions vnf1 interfaces eth1 mapping vlan members
  v30
user@jdm# set virtual-network-functions vnf1 interfaces eth1 mapping vlan members
  v40
user@jdm# set virtual-network-functions vnf1 interfaces eth1 mapping vlan mode
  trunk
user@jdm# set virtual-network-functions vnf2 interfaces eth2 mapping vlan members
  v10
user@jdm# set virtual-network-functions vnf2 interfaces eth2 mapping vlan members
  v20
user@jdm# set virtual-network-functions vnf2 interfaces eth2 mapping vlan members
  v30
user@jdm# set virtual-network-functions vnf2 interfaces eth2 mapping vlan mode
  trunk
user@jdm# set virtual-network-functions vnf2 interfaces eth3 mapping vlan members
  v10
user@jdm# set virtual-network-functions vnf2 interfaces eth3 mapping vlan members
  v20
user@jdm# set virtual-network-functions vnf2 interfaces eth3 mapping vlan members
  v30
user@jdm# set virtual-network-functions vnf2 interfaces eth3 mapping vlan mode
  trunk
user@jdm# set virtual-network-functions vnf4 interfaces eth4 mapping vlan members
  v30
user@jdm# set virtual-network-functions vnf5 interfaces eth5 mapping vlan members
  v50
user@jdm# set virtual-network-functions vnf5 interfaces eth6
user@jdm# set virtual-network-functions vnf6 interfaces eth7
```

3. Configure cross-connect:

   - Configure VLAN-based cross-connect:

     ```
     user@jdm# set cross-connect vlan-cross-connect-name virtual-network-functions
       vnf-name interfaces interface-name vlan-id vlan-id
     user@jdm# set cross-connect c1 virtual-network-functions vnf1 interfaces eth2
       vlan-id 10
     user@jdm# set cross-connect c1 virtual-network-functions vnf4 interfaces eth2
       vlan-id 10
     ```

   - Configure unconditional cross-connect

     ```
     user@jdm# set cross-connect unconditional-cross-connect-name
       virtual-network-functions vnf-name interfaces interface-name
     user@jdm# set cross-connect c2 virtual-network-functions vnf1 interfaces eth1
     user@jdm# set cross-connect c2 virtual-network-functions vnf5 interfaces eth5
     ```

   - Configure cross-connect with VLAN SWAP operation enabled:

```
user@jdm# set cross-connect swap-cross-connect-name virtual-network-functions
  vnf-name interfaces interface-name vlan-id vlan-id
user@jdm# set cross-connect c1 virtual-network-functions vnf1 interfaces eth1
  vlan-id 10
user@jdm# set cross-connect c1 virtual-network-functions vnf4 interfaces eth4
  vlan-id 30
```

- Configure cross-connect with VLAN PUSH or POP operation enabled:

```
user@jdm# set cross-connect push-pop-cross-connect-name
  virtual-network-functions vnf-name interfaces interface-name vlan-id vlan-id
user@jdm# set cross-connect push-pop-cross-connect-name
  virtual-network-functions vnf-name interfaces interface-name
user@jdm# set cross-connect c3 virtual-network-functions vnf1 interfaces eth1
  vlan-id 10
user@jdm# set cross-connect c3 virtual-network-functions vnf6 interfaces eth7
```

- Configure native VLAN traffic on cross-connect

```
user@jdm# set cross-connect push-pop cross-connect-name
  virtual-network-functions vnf-name interfaces interface-name vlan-id vlan-id
user@jdm# set cross-connect c2 virtual-network-functions vnf1 interfaces eth1
  vlan-id none
user@jdm# set cross-connect c2 virtual-network-functions vnf5 interfaces eth5
  vlan-id none
```

See Also

- cross-connect on page 140

Configuring Analyzer VNF and Port-mirroring

The **Port-mirroring** feature allows you to monitor network traffic. If the feature is enabled on a VNF interface, the OVS system bridge sends a copy of all network packets of that VNF interface to the analyzer VNF for analysis. You can use the port-mirroring or analyzer JDM commands for analyzing the network traffic.

NOTE:

- Port-mirroring is supported only on VNF interfaces that are connected to an OVS system bridge.
- VNF interfaces must be configured before configuring port-mirroring options.
- If the analyzer VNF is active after you configure, you must restart the VNF for changes to take effect.
- You can configure up to four input ports and only one output port for an analyzer rule.
- Output ports must be unique in all analyzer rules.
- After changing the configuration of the input VNF interfaces, you must de-activate and activate the analyzer rules referencing to it along with the analyzer VNF restart.

To configure the analyzer VNF and enable port-mirroring:

1. Configure the analyzer VNF:

```
[edit]
user@jdm#set virtual-network-functions analyzer-vnf-name image file-path
user@jdm#set virtual-network-functions analyzer-vnf-name memory features
   hugepages page-size page-size
user@jdm#set virtual-network-functions analyzer-vnf-name interfaces interface-name
   analyzer
```

2. Enable port-mirroring of the network traffic in the input and output ports of the VNF interface and analyzer VNF:

```
user@jdm# set host-os forwarding-options analyzer analyzer-instance-name input
   [ingress | egress] virtual-network-function vnf-name interface interface-name
user@jdm# set host-os forwarding-options analyzer analyzer-rule-name output
   virtual-network-function analyzer-vnf-name interface interface-name
```

See Also

- host-os forwarding-options analyzer on page 143

Accessing a VNF from JDM

You can access a VNF from JDM using either SSH or a VNF console.

1. To access a VNF using SSH:

   ```
   user@jdm> ssh vnf-name
   ```

2. To access a VNF using a virtual console:

   ```
   user@jdm> request virtual-network-functions vnf-name console
   ```

NOTE:

- Use ctrl-] to exit the virtual console.
- Do not use Telnet session to run the command.

Viewing List of VNFs

1. To view the list of VNFs:

   ```
   user@jdm> show virtual-network-functions
   ID        Name                    State            Liveliness
   ------------------------------------------------------------------
   3         vjunos0                 running          alive
   -         vsrx                    shut off         down
   ```

The **Liveliness** output field of a VNF indicates whether the IP address of the VNF is reachable or not reachable from JDM. The default IP address of the liveliness bridge 192.0.2.1/24.

Displaying the VNF Details

To display VNF details:

```
user@jdm> show virtual-network-functions vnf-name
Virtual Machine Information
---------------------------
Name:               vsrx
IP Address:         192.0.2.4
Status:             Running
Liveliness:         Up
VCPUs:              1
Maximum Memory:     2000896
Used Memory:        2000896
Virtual Machine Block Devices
-----------------------------
Target   Source
---------------
hda
```

```
/var/third-party/images/vsrx/media-srx-ffp-vsrx-vmdisk-15.1-2015-05-29_X_151_X49.qcow2
hdf      /var/third-party/test.iso
```

Deleting a VNF

To delete a VNF:

```
user@jdm# delete virtual-network-functions vnf-name
```

NOTE: The VNF image remains in the disk even after you delete the VNF.

Virtual Route Reflector on NFX250 Overview

The virtual Route Reflector (vRR) feature allows you to implement route reflector capability using a general purpose virtual machine that can be run on a 64-bit Intel-based blade server or appliance. Because a route reflector works in the control plane, it can run in a virtualized environment. A virtual route reflector on an Intel-based blade server or appliance works the same as a route reflector on a router, providing a scalable alternative to full mesh internal BGP peering.

Starting in Junos OS Release 17.3R1, you can implement the virtual route reflector (vRR) feature on the NFX250 Network Services platform. The Juniper Networks NFX250 Network Services Platform comprises the Juniper Networks NFX250 devices, which are Juniper Network's secure, automated, software-driven customer premises equipment (CPE) devices that deliver virtualized network and security services on demand. NFX250 devices use the Junos Device Manager (JDM) for virtual machine (VM) lifecycle and device management, and for a host of other functions. The JDM CLI is similar to the Junos OS CLI in look and provides the same added-value facilities as the Junos OS CLI.

Benefits of vRR

vRR has the following benefits:

- Scalability: By implementing the vRR feature, you gain scalability improvements, depending on the server core hardware on which the feature runs. Also, you can implement virtual route reflectors at multiple locations in the network, which helps scale the BGP network with lower cost. The maximum routing information base (RIB) scale with IPv4 routes on NFX250 is 20 million.
- Faster and more flexible deployment: You install the vRR feature on an Intel server, using open source tools, which reduces your router maintenance.
- Space savings: Hardware-based route reflectors require central office space. You can deploy the virtual route reflector feature on any server that is available in the server infrastructure or in the data centers, which saves space.

Software Requirements for vRR on NFX250

The following software components are required to support vRR on NFX250:

- Juniper Device Manager: The Juniper Device Manager (JDM) is a low-footprint Linux container that supports Virtual Machine (VM) lifecycle management, device management, Network Service Orchestrator module, service chaining, and virtual console access to VNFs including vSRX, vjunos, and now vRR as a VNF.
- Junos Control Plane: Junos Control Plane (JCP) is the Junos VM running on the hypervisor. You can use JCP to configure the network ports of the NFX250 device, and JCP runs by default as vjunos0 on NFX250. You can log on to JCP from JDM using the SSH service and the command-line interface (CLI) is the same as Junos.

vRR VM Memory Allocation Limitations for NFX250

vRR has the following memory allocation limitations:

- Default mode of memory allocation can give flexibility to create up to 26GB size of VRR VM with 32GB available memory in NFX250-S2.
- 26GB out of 32GB is available for vRR VM in NFX250-S2. By default, JDM requires 2 GB, JCP requires 2GB, and the base Linux OS requires 2GB. The rest is available for vRR.
- Use default non-huge page mode of allocation for vRR VM creation. Page-size and page-count values depends on the size and total number of hugepages required by all the VNFs that will be launched in the system. vRR VM can be created with huge pages' allocation with the limit of 16 huge page size limitation of 16GB max for VRR.

Release History Table

Release	Description
17.3R1	Starting in Junos OS Release 17.3R1, you can implement the virtual route reflector (vRR) feature on the NFX250 Network Services platform.

Related Documentation

- Configuring vRR as a VNF on NFX250 on page 128

Configuring vRR as a VNF on NFX250

Configuring Junos Device Manager (JDM) for vRR

By default, the Junos Device Manager (JDM) virtual machine comes up after NFX250 is powered on. By default, enhanced orchestration mode is enabled on JDM. While configuring vRR, disable enhanced orchestration mode, remove the interfaces configuration, and reboot the NFX device.

To configure the Junos Device Manager (JDM) virtual machine for vRR, perform the following steps:

1. In configuration mode, at the [**edit**] hierarchy level, disable enhanced orchestration. By default, enhanced orchestration mode is enabled on JDM.

```
[edit ]
user@jdm# delete system services enhanced-orchestration
```

2. Delete interface configuration.

```
[edit ]
user@jdm# delete interfaces
```

3. Set the JDM root password.

```
[edit ]
user@jdm# set system root-authentication plain-text-password
```

4. Commit the configuration using the **commit** command and reboot the system for the configuration to take effect.

```
[edit ]
user@jdm#commit
user@jdm#run request reboot
```

5. After the system reboots, the default bridges configuration is available on JDM. Configure the JDM root password, management port IP, and add default routes.

NOTE: After the system reboots, if the groups **groups1604-configs** is not present in the configuration, include it so the default bridges configuration is available on JDM.

```
[edit ]
user@jdm# set system root-authentication encrypted-password
user@jdm# set system services ssh
user@jdm# set interface eth0 vlan-id valn-id family inet address address
user@jdm# set route destination address next-hop address
```

Verifying that the Management IP is Configured

Purpose Ensure that the management IP address has been configured accurately.

Action From configuration mode, enter the **show interface** command.

```
user@jdm# show interface eth0
```

```
vlan-id 0 {
    family {
        inet {
            address 10.48.14.18/22;
        }
    }
}
```

Verifying that the Default Routes are Configured

Purpose Ensure that the default routes are configured for DNS and gateway access.

Action From configuration mode, enter the **show route** command.

```
user@jdm# show route
```

```
destination 172.16.0.0/12 next-hop 10.48.15.254;
destination 192.168.0.0/16 next-hop 10.48.15.254;
destination 207.17.136.0/24 next-hop 10.48.15.254;
destination 10.0.0.0/10 next-hop 10.48.15.254;
destination 10.64.0.0/10 next-hop 10.48.15.254;
destination 10.128.0.0/10 next-hop 10.48.15.254;
destination 10.192.0.0/11 next-hop 10.48.15.254;
destination 10.224.0.0/12 next-hop 10.48.15.254;
destination 10.240.0.0/13 next-hop 10.48.15.254;
destination 10.248.0.0/14 next-hop 10.48.15.254;
destination 10.252.0.0/15 next-hop 10.48.15.254;
destination 10.254.0.0/16 next-hop 10.48.15.254;
destination 66.129.0.0/16 next-hop 10.48.15.254;
destination 10.48.0.0/15 next-hop 10.48.15.254;
```

Configuring Junos Control Plane(JCP) for vRR

By default, the Junos Control Plane (JCP) VM comes up after NFX250 is powered on. The JCP virtual machine controls the front panel ports in the NFX250 device. VLANs provide the bridging between the virtual route reflector VM interfaces and the JCP VMs using **sxe** ports. The front panel ports are configured as part of the same VLAN bridging of the VRR ports. As a result, packets are transmitted or received using these bridging ports between JCP instead of the vRR VM ports.

To configure JCP for vRR, perform the following steps:

1. In the operational mode, connect to the JCP virtual machine.

```
user@jdm>ssh vjunos0
user@jcp> configure
user@jcp#
```

2. Configure the front panel ports with the same VLAN bridging of vRR VM ports. In this example, the front panel ports, **ge-0/0/1**, **ge-0/0/10**, and **xe-0/0/12** are mapped with vRR VM interfaces, **em1**, **em2**, and **em3**. The **ge-0/0/1** (front panel port) maps to the **sxe-0/0/0** (internal interface) which maps to **em1** (vRR VM interface). They are all part of the same VLAN (VLAN ID 100).

```
root#set interfaces ge-0/0/1 unit 0 family ethernet-switching interface-mode trunk
root# set interfaces ge-0/0/10 unit 0 family ethernet-switching interface-mode trunk
root# set interfaces xe-0/0/12 unit 0 family ethernet-switching interface-mode trunk
```

```
root# set interfaces sxe-0/0/0 unit 0 family ethernet-switching interface-mode trunk
root# set interfaces sxe-0/0/1 unit 0 family ethernet-switching interface-mode trunk
```

3. Configure the VLANs and add the physical interface and the service interface as members of the same VLAN. In this example, we have 3 VLANs (100, 101, and 102).

```
root@jcp# set vlans vlan100 vlan-id 100
root@jcp# set vlans vlan101 vlan-id 101
root@jcp# set vlans vlan102 vlan-id 102
root@jcp#set interfaces ge-0/0/1 unit 0 family ethernet-switching interface-mode
  trunk vlan members vlan100
root@jcp# set interfaces ge-0/0/10 unit 0 family ethernet-switching interface-mode
  trunk vlan members vlan101
root@jcp# set interfaces xe-0/0/12 unit 0 family ethernet-switching interface-mode
  trunk vlan members vlan102
root@jcp# set interfaces sxe-0/0/0 unit 0 family ethernet-switching interface-mode
  trunk vlan members 100
root@jcp# set interfaces sxe-0/0/1 unit 0 family ethernet-switching interface-mode
  trunk vlan members 101-102
```

4. Verify that you have configured the mapping of interfaces accurately.

```
sxe-0/0/0 {
       ether-options {
            no-flow-control;
          }
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
                vlan {
                    members [ 100 default vlan-2 ];
                }
            }
        }
    }
ge-0/0/1 {
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
                vlan {
                    members [ default vlan100 ];
                }
                storm-control default;
```

```
            }
        }
    }
    sxe-0/0/1 {
        ether-options {
            no-flow-control;
        }
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
                vlan {
                    members [ 101-102 vlan-3 ];
                }
            }
        }
    }
ge-0/0/10 {
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
                vlan {
                    members [ default vlan101 ];
                }
                storm-control default;
            }
        }
    }
    xe-0/0/12 {
        unit 0 {
            family ethernet-switching {
                interface-mode trunk;
                vlan {
                    members [ vlan-2 vlan102 ];
                }
                storm-control default;
            }
        }
    }
vlans{
..
        vlan100 {
        vlan-id 100;
    }
    vlan101 {
        vlan-id 101;
    }
    vlan102 {
        vlan-id 102;
    }
  }
```

Launching vRR

You can launch the vRR VM as a virtualized network function (VNF) using the XML configuration templates that are part of the vRR image archive.

1. To launch the vRR VM, use the **vrish** command and specify the VM name.

```
root$ virsh create vrr.xml
```

2. To create a vRR VM of size 26 GB with 2 virtual CPU(s) and 3 vRR VM interfaces (**em1**, **em2**, and **em3**) as shown in this example, you can use this sample configuration.

```
<domain type= 'kvm'id='10'>
<name> vrr </name>
<uuid> bf4b183b-1a41-43e3-90ea-895feb43f63c </uuid>
<memory unit='KiB'>27262976</currentMemory>
<vcpu placement= 'static' cpuset='0-1'> 2 </vcpu>
<cputune>
<vcpupin vcpu='0' cpuset='0'/>
<vcpupin vcpu='1' cpuset='1'/>
</cputune>
<resource>
<partition>/machine </partition>
</resource>
<sysinfo type='smbios'>
<bios>
<entry name='vendor'>Juniper </entry>
</bios>
<system>
<entry name='manufacturer'>Juniper </entry>
<entry name='product'>VRR </entry>
<entry name='version'>17.2</entry>
</system>
</sysinfo>
<os>
<type arch='x86_64' machine='pc-i440fx-1.7'>hvm</type>
<boot dev='hd'/>
<smbios mode='sysinfo'/>
</os>
<features>
<acpi/>
<apic/>
<pae/>
</features>
<cpu mode='custom' match='exact'>
<model fallback='allow'> SandyBridge</model>
</cpu>
<clock offset='utc'/>
<on_poweroff>destroy </on_poweroff>
<on_reboot>restart </on_reboot>
<on_crash>restart </on_crash>
<devices>
<emulator>/usr/bin/kvm</emulator>
<disk type='file' device='disk'>
<driver name='qemu' type='qcow2' cache='none'/>
<source file='/var/third-party/junos-x86-64-17.3-20170405_dev_common.0.img'/>
<backingstore/>
<target dev='had' bus='ide'/>
<alias name='ide0-0-0' />
<address type='drive' controller='0' bus='0' target='0' unit='0'/>
</disk>
<controller type='pci' index='0' model='pci-root'>
<alias name='pci.0'/>
</controller>
<interface type='bridge'>
```

```
<mac address='2c:21:31:5f:2e:0b'/>
<source bridge= 'eth0br'/>
<target dev='vnet5'/>
<model type='e1000'/>
<alias name='net0'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x0'/>
</interface>
<interface type='bridge'>
<mac address='52:54:00:d0:c5:a3'/>
<source bridge= 'sxe1br'/>
<target dev='vnet7'/>
<model type='e1000'/>
<alias name='net2'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x0'/>
</interface>
<interface type='bridge'>
<mac address='52:54:00:85:7c:40'/>
<source bridge= 'sxe1br'/>
<target dev='vnet8'/>
<model type='e1000'/>
<alias name='net3'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x05' function='0x0'/>
</interface>
<serial type='pty'>
<source path='/dev/pts/0'/>
<target port='0'/>
<alias name='serial0'/>
</serial>
<console type='pty' tty='/dev/pts/0'>
<source path='/dev/pts/0'/>
<target type='serial'port='0'/>
<alias name='serial0'/>
</console>
<input type='mouse' bus='ps2'/>
<input type='keyboard' bus='ps2'/>
<graphics type='vnc' port='5900' autoport='yes' listen='127.0.0.1'>
<listen type='address' address='127.0.0.1'/>
</graphics>
<sound model='ich6'>
<alias name='sound0'>
<address type='pci' domain='0x0000' bus='0x00' slot='0x08' function='0x0'/>
</sound>
<video>
<model type='cirrus' vram='16384' heads='1'/>
<alias name='video0'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x08' function='0x0'/>
</video>
<memballoon model='virtio'>
<alias name='balloon0'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x09' function='0x0'/>
</memballoon>
</devices>
<seclabel type='none' model='none'/>
</domain>
```

Related Documentation

- Virtual Route Reflector on NFX250 Overview on page 127
- Enabling Liveliness Detection of vRR VM from JDM on page 135

Enabling Liveliness Detection of vRR VM from JDM

Liveliness of a virtual machine (VM) indicates if the IP address of the VM is accessible to the Junos Device Manager (JDM). If the liveliness of the VM is down, it implies that the VM is not reachable from JDM. You can view the liveliness of VMs using the **show virtual-machines** command. By default, the liveliness of vRR VM is shown as down. Before creating the vRR VM, it is recommended that you enable liveliness detection in JDM.

To enable liveliness detection of the vRR VM from JDM, perform the following steps:

1. To verify that liveliness detection of vRR VM from JDM, issue the following command:

   ```
   user@jdm> show virtual-machines
   ID       Name                                 State      Liveliness
   ----------------------------------------------------------------------
   2        vjunos0                              running    alive
   18       VRR                                  running    down
   8366     jdm                                  running    alive
   ```

   By default, the liveliness of vRR VM is shown as down. You must enable liveliness detection of vRR VM from the JDM.

2. Create a dummy interface with internal bridge, **virbr0**, by modifying the network interface settings for the vRR VM interface. stanza of the VM template like as shown below. The PCI details like **bus**, **slot**, **function** information can be based on your existing interfaces arbitrary running next number, especially the slot number.

   This is a sample network interface setting:

   ```
   <interface type='bridge'>
       <source bridge='eth0br'/>
   <model type='e1000'/>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x0'/>
   </interface>
   ```

   You must change the settings as follows:

   ```
   <interface type='bridge'>
       <source bridge='virbr0'/>
   <model type='e1000'/>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x06' function='0x0'/>
   </interface>
   ```

   When you are modifying the settings make sure that:

   - The interface type is **'bridge'**.
   - The model type is **e1000** to prevent problems with VLAN subinterfaces.
   - The PCI resource for the address is unique for this VM.

3. To identify the MAC address associated with the **virbr0** interface, use the **virsh dumpxml** ***vrr-vm-name*** command.

```
user@jdm:/var/third-party# virsh dumpxml VRR

...
<interface type='bridge'>
<mac address='52:54:00:c4:fe:8d'/>
    <source bridge='virbr0'/>
<model type='e1000'/>
 <address type='pci' domain='0x0000' bus='0x00' slot='0x06' function='0x0'/>
</interface>
...
```

This is the MAC address assigned by the VRR VM to the **virbr0** interface.

4. To assign an IP address to the vRR VM interface connected to **virbr0** interface, you must use an IP, that is part of the internal network. In this example, the MAC address, **52:54:00:c4:fe:8d**, assigned by the vRR VM is associated with the **em4** interface of the vRR VM. So, we must configure the **em4** interface with the IP address as shown in this step.

```
user@> set interfaces em4 unit 0 family inet address 192.168.1.10/24
user@# run show interfaces em4
Physical interface: em4, Enabled, Physical link is Up
  Interface index: 130, SNMP ifIndex: 154
  Type: Ethernet, Link-level type: Ethernet, MTU: 1514, Speed: 1000mbps
  Device flags   : Present Running
  Interface flags: SNMP-Traps
  Link type      : Full-Duplex
  Current address: 52:54:00:c4:fe:8d, Hardware address: 52:54:00:c4:fe:8d
  Last flapped   : 2017-07-11 07:42:38 UTC (00:04:42 ago)
    Input packets : 40
    Output packets: 4

  Logical interface em4.0 (Index 69) (SNMP ifIndex 155)
    Flags: Up SNMP-Traps 0x4000000 Encapsulation: ENET2
    Input packets : 37
    Output packets: 4
    Protocol inet, MTU: 1500
    Max nh cache: 100000, New hold nh limit: 100000, Curr nh cnt: 0,
    Curr new hold cnt: 0, NH drop cnt: 0
      Flags: Sendbcast-pkt-to-re, Is-Primary
      Addresses, Flags: Is-Default Is-Primary
        Local: 192.168.1.10
```

The MAC address assigned by the vRR VM in this example is associated with **em4** interface.

5. In Junos Device Manager (JDM), update the file **/etc/hosts** with the IP address and vRR VM name.

NOTE: When you update the **/etc/hosts** file, include space between the IP address and the vRR VM name. Do not include tab spaces.

```
user@jdm> cat /etc/hosts
```

```
127.0.0.1       localhost
::1             localhost ip6-localhost ip6-loopback
ff02::1         ip6-allnodes
ff02::2         ip6-allrouters
192.168.1.1     hypervisor
192.168.1.2     vjunos0
192.168.1.3     vjunos1
192.168.1.4     vsrx
192.168.1.254   jdm nfx-vrr-jdm
128.0.0.254     jdm nfx-vrr-jdm
192.168.1.5     ipsec-nm
192.168.1.10    VRR
```

6. Ping the IP address of the vRR VM from JDM to verify that the internal bridge **virbr0** is accessible from JDM.

```
user@jdm> ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
64 bytes from 192.168.1.10: icmp_seq=1 ttl=64 time=0.316 ms
64 bytes from 192.168.1.10: icmp_seq=2 ttl=64 time=0.379 ms
^C
--- 192.168.1.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.316/0.347/0.379/0.036 ms
```

7. To verify that liveliness detection of vRR VM from JDM, issue the following command:

```
user@jdm> show virtual-machines
ID       Name                                State      Liveliness
------------------------------------------------------------------
2        vjunos0                             running    alive
18       VRR                                 running    alive
8366     jdm                                 running    alive
```

Now, the liveliness status of vRR VM is shown as alive.

Related Documentation

- Virtual Route Reflector on NFX250 Overview on page 127
- Configuring vRR as a VNF on NFX250 on page 128

CHAPTER 6

Virtual Network Functions Configuration Statements and Operational Commands

cross-connect

Syntax

```
cross-connect {
  cross-connect-name {
    physical-interface {
      hsxe0 {
        vlan-id vlan-id;
      }
      hsxe1 {
        vlan-id vlan-id;
      }
    }
    virtual-network-function vnf-name {
      interfaceinterface-name;
      vlan-id vlan-id;
    }
  }
  push-pop-cross-connect-name {
    virtual-network-function vnf-name {
      interfaceinterface-name;
      vlan-id vlan-id;
    }
    virtual-network-function vnf-name {
      interfaceinterface-name;
    }
  }
  swap-cross-connect-name {
    virtual-network-function vnf-name {
      interfaceinterface-name;
      vlan-id vlan-id;
    }
    virtual-network-function vnf-name {
      interfaceinterface-name;
      vlan-id vlan-id;
    }
  }
  unconditional-cross-connect-name {
    virtual-network-function vnf-name {
      interfaceinterface-name;
    }
    virtual-network-function vnf-name {
      interfaceinterface-name;
    }
  }
  vlan-based-cross-connect-name{
    virtual-network-function vnf-name {
      interfaceinterface-name;
      vlan-id vlan-id;
    }
    virtual-network-function vnf-name {
      interfaceinterface-name;
      vlan-id vlan-id;
    }
```

```
  }
}
```

Hierarchy Level [edit]

Release Information Statement introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description Connects any two VNF interfaces, VLANs on physical interfaces such as hsxe0 and hsxe1, and also provides features such as unconditional cross-connect, VLAN-based cross-connect, cross-connect of native VLAN traffic, and cross-connect with operations such as PUSH, POP, and SWAP.

Options **virtual-network-functions *vnf-name***—Name of the VNF instance.

cross-connect-name—Name of the cross-connect.

vlan-id—Virtual LAN identifier.

Required Privilege Level set—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Configuring Cross-connect*

features

Syntax

```
features {
  hugepages;
  }
```

Hierarchy Level

```
[edit virtual-network-functions]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description

Displays the supported features of a VNF.

Options

features—Features of a VNF.

hugepages—Option to support memory pages with a size of 2 MB and 1 GB.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

host-os forwarding-options analyzer

Syntax

```
forwarding-options {
  analyzer analyzer-instance-name {
    input {
      ingress {
        virtual-network-function vnf-name {
          interface interface-name;
        }
      }
      egress {
        virtual-network-function vnf-name {
          interface interface-name;
        }
      }
    }
    output {
      virtual-network-function analyzer-vnf-name;
        interface interface-name;
    }
  }
```

Hierarchy Level

[edit]

Release Information

Statement introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description

Configures an analyzer for port mirroring and configures port mirroring for either ingress or egress traffic of a VNF interface to an analyzer VNF.

Options

virtual-network-functions *vnf-name*—Name of the VNF instance.

analyzer-instance-name—Name of the analyzer.

ingress—Traffic that enters the port.

egress—Traffic that leaves the port.

Required Privilege Level

set host-os—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

•

hugepages

Syntax hugepages;

Hierarchy Level [edit virtual-network-functions]

Release Information Statement introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description An option to support of 2 MB and 1 GB size memory pages.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

image

Syntax

```
image {
  file-path;
  bus-type [ide | virtio];
  image-type [qcow2 | raw];
}
```

Hierarchy Level

```
[edit virtual-network-functions]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description

Specify the VNF image source file. VNF image is virtual hard disk, which contains the bootable file-system for the VNF.

Options

file-path—Path of the image source file.

image-type—Format of the image. Default value is qcow2.

bus-type—Type of the system bus. Default value is virtio.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

init-descriptor

Syntax init-descriptor *file-path*;

Hierarchy Level [edit virtual-network-functions]

Release Information Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description Create an XML descriptor file to launch a VNF. You can launch a VNF by configuring the VNF name, and specifying either the path to the XML descriptor file or to an image.

Options ***file-path***—Path of the init-descriptor XML file.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

interfaces

Syntax

```
interfaces interface-name {
  pci-address pci-address;
    mac-address mac-address;
    mtu size;
    mapping {
    hsxe0 {
      virtual-function {
        vlan-id vlan-id;
      }
    }
    hsxe1 {
      virtual-function {
        vlan-id vlan-id;
      }
    }
    vlan {
      members vlan-name;
        mode [access | trunk];
      native-vlan-id vlan-id;
    }
  }
}
```

Hierarchy Level

```
[edit virtual-network-functions]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D50 for the NFX250 Network Services Platform.

Description

Configure Virtual Network Functions (VNF) interfaces on platforms running disaggregated Junos OS.

Options

interface-name—Name of the VNF interfaces.

mac-address—MAC address of the VNF interfaces.

mtu—Maximum transfer unit (MTU) size of packets in bytes.

pci-address—Target PCI address of the VNF interfaces.

vlan-id—SR-IOV virtual function to use to attach a VNF to a physical interface.

vlan *members*—Membership for this interface.

native-vlan-id—Virtual LAN identifier for untagged frames. For example, 1...4095

vlan-name—Name of the VLAN members.

Required Privilege Level
interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- show virtual-network-functions on page 165

ipsec-nm

Syntax

```
ipsec-nm {
  interfaces {
    heth1 | heth2 {
      mapping {
        vlan {
          members vlan-name;
        }
      }
    }
  }
}
```

Hierarchy Level

```
[edit virtual-network-functions]
```

Release Information
Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description
An option that enables or disables the ipsec virtual network function if the ipsec-nm option is configured in the system.

Options
interfaces—Name of the interface. For example, heth1 and heth2

vlan members *vlan-name*—Name of the VLAN members.

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- show virtual-network-functions on page 165

mac-address

Syntax mac-address *mac-address*;

Hierarchy Level [edit virtual-network-functions]

Release Information Statement introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description MAC address for the VNF interfaces.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

mapping

Syntax

```
mapping {
  hsxe0 {
    virtual-function {
      vlan-id vlan-id;
    }
  }
  hsxe1 {
    virtual-function {
      vlan-id vlan-id;
    }
  }
  vlan {
    members vlan-name;
      mode [access | trunk];
    native-vlan-id vlan-id;
  }
  peer-interfaces peer-interface-name;
}
```

Hierarchy Level

```
[edit virtual-network-functions]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D50 for the NFX250 Network Services Platform.

Description

Mapping Virtual Network Functions (VNF) interfaces on platforms running disaggregated Junos OS.

Options

vlan-id—SR-IOV virtual function to use to attach a VNF to a physical interface.

vlan *members*—Membership for this interface.

native-vlan-id—Virtual LAN identifier for untagged frames. For example, 1...4095

vlan-name—Name of the VLAN members.

peer-interface-name—Name of the virtual peer interfaces that is mapped to a physical interface.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

memory

Syntax

```
memory {
  size size;
  features {
  hugepages;
  }
}
```

Hierarchy Level

```
[edit virtual-network-functions]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description

Configure memory parameters for VNFs on a platform that is running a disaggregated Junos OS.

Options

memory *size*—Amount of memory allocated to a VNF in kilobytes. The default size is 1 GB.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

mtu

Syntax mtu *size*;

Hierarchy Level [edit interfaces *interface-name*]

Release Information Statement introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description Specify the maximum transmission unit (MTU) size for the media in bytes. MTU size can be either 1500 bytes or 2048 bytes.

Options ***size***—Size of the MTU

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

no-autostart

Syntax no-autostart;

Hierarchy Level [edit virtual-network-functions]

Release Information Statement introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Disable auto-start of VNF on the VNF configuration commit.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

offloads

Syntax

```
offloads {
  disable;
}
```

Hierarchy Level

```
[edit virtual-network-functions]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D471 for the NFX250 Network Services Platform.

Description

Offloads configuration for the VNF interface.

NOTE: This feature is available only in enhanced-orchestration mode.

Options

disable—Disable the **offloads** option that offloads the configuration for the VNF interface.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

pci-address

Syntax pci-address *pci-address*;

Hierarchy Level [edit virtual-network-functions]

Release Information Statement introduced in Junos OS Release 15.1X53-D50 for the NFX250 Network Services Platform.

Description PCI address for the VNF interfaces.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

size

Syntax size *size*;

Hierarchy Level [edit virtual-network-functions]

Release Information Statement introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Size of the memory in kilobytes.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

storage

Syntax

```
storage device-name {
  type {
    cdrom {
      source {
        file filename;
      }
    }
    disk {
      bus-type [ide | virtio];
        file-type [qcow2 | raw];
        source {
        file filename;
      }
    }
    usb {
      source {
        file filename;
      }
    }
  }
}
```

Hierarchy Level

```
[edit virtual-network-functions]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description

Configure storage parameters on VNFs.

Options

storage *device-name*—Name of the storage device. For example, hda, hdb, sdb, or vdb.

type *disk-type*—Type of disk. For example, cdrom, usb, or disk.

source *file-name*—Path of the source file of the storage device.

NOTE: The source file to be attached to the VNF must be located in the /var/third-party/ directory and must have read and write permissions for all.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- show virtual-network-functions on page 165

type

Syntax

```
type {
  linux-container | virtual-machine;
  }
```

Hierarchy Level

```
[edit virtual-network-functions]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description

Type of the VNF.

Options

linux-container—The VNF type is Linux container.

virtual-machine—The VNF type is virtual machine.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- show virtual-network-functions on page 165

virtual-cpu

Syntax

```
virtual-cpu {
  virtual-cpu-number {
    physical-cpu number | range;
  }
  count number;
  features {
    hardware-virtualization;
  }
}
```

Hierarchy Level

```
[edit virtual-network-functions]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description

Specify the number of virtual CPUs the VNF can use. By default, a VNF is assigned one virtual CPU, which is independent of any specific physical CPU.

Options

virtual-cpu count *number*—Number of virtual CPUs. For example, 2.

virtual-cpu features—Features of the virtual cpu. For example, hardware-virtualization.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

virtual-network-functions

Syntax

```
virtual-network-functions vnf-name;
```

Hierarchy Level

```
virtual-network-functions {
  ipsec-nm {
    interfaces {
      heth1 | heth2 {
        mapping {
          vlan {
            members vlan-name;
          }
        }
      }
    }
  }
  vjunos0 {
    interfaces {
      em1 {
        mapping {
          vlan {
            members vlan-name;
          }
        }
      }
    }
  }
  vmx {
    image filepath;
      virtual-cpu {
      count virtual-cpu-count;
      features;
    }
    memory size size;
    storage {
    vdc {
      type type;
      source file-path;
    }
    vdb {
      type type;
      source file-path;
    }
  }
  vsrx {
  vnf-name  {
    type {
      linux-container | virtual-machine;
    }
    image {
      file-path;
      bus-type [ide | virtio];
```

```
    image-type [qcow2 | raw];
}
init-descriptor file-path;
memory {
    size size;
    features {
        hugepages ;
    }
}
no-autostart;
storage device-name {
    type {
        cdrom {
            source {
                file filename;
            }
        }
        disk {
            bus-type [ide | virtio];
                file-type [qcow2 | raw];
                source {
                file filename;
            }
        }
        usb {
            source {
                file filename;
            }
        }
    }
}
virtual-cpu {
    virtual-cpu-number {
        physical-cpu number | range;
    }
    count number;
    features {
        hardware-virtualization;
    }
}
interfaces interface-name {
    pci-address pci-address;
    mapping {
        hsxe0 {
            virtual-function {
                vlan-id vlan-id;
            }
        }
        hsxe1 {
            virtual-function {
                vlan-id vlan-id;
            }
        }
        vlan {
            members vlan-name;
```

```
            mode [access | trunk];
            native-vlan-id vlan-id;
          }
          peer-interfaces peer-interface-name;
        }
        management {
          internal;
          out-of-band;
        }
      }
    }
```

Release Information Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description Create an instance of a virtual network function (VNF) on platforms that run disaggregated Junos OS software.

NOTE:

- Creating a VNF instance fails if the resources required by the VNF are not available in the system.
- If you use init-descriptor to define a VNF by specifying and setting different values for the virtual CPU count or memory and later if you delete the virtual cpu count, the system restores the value to a default value of 1 for vCPU and 1GB for memory.
- You can enable the VNF options such virtual-cpu-hardware-virtualization (vmx), hugepages, image-type, and image-bus-type only when you define the VNF initially. You cannot enable or disable the VNF options after committing the VNF configuration. To enable or disable the VNF options, you must delete the VNF configuration and re-configure with the VNF options.

Options **virtual-network-functions** ***vnf-name***—Name of the VNF instance. It is mandatory to provide one of the options: init-descriptor or image.

file-path—Path of the source file.

number | range—Number or a range of physical CPUs. For example, 2 or 2...5.

interface-name—Name of the VNF interface, which can range from eth0 to eth9. You can configure eth0 and eth1 interfaces and can assign VLAN IDs. To configure eth0 and eth1 interfaces, you must configure **no-default-interfaces** option.

physical-interface-name—Name of the physical interface to which the VNF interface is attached.

vlan-id—SR-IOV virtual function to use to attach a VNF to a physical interface.

native-vlan-id—Virtual LAN identifier for untagged frames. For example, 1...4095

vlan-name—Name of the VLAN members.

size—Amount of memory allocated to a VNF in kilobytes. The default size is 1 GB.

device-name—Name of the storage device.

file-name—Name of the source file of the storage device.

peer-interface-name—Name of the virtual peer interfaces that is mapped to a physical interface.

management—VNF interface management configuration.

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- show virtual-network-functions on page 165

vjunos0

Syntax

```
vjunos0 {
  interfaces {
    em1 {
      mapping {
        vlan {
          members vlan-name;
        }
      }
    }
  }
}
```

Hierarchy Level

```
[edit virtual-network-functions]
```

Release Information
Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description
An option that enables or disables the vjunos virtual network function if the vjunos0 option is configured in the system.

Options
interfaces—Name of the interface. For example, em1.

vlan members *vlan-name*—Name of the VLAN members.

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- show virtual-network-functions on page 165

vnf-name

Syntax

```
vnf-name {
  type {
    linux-container | virtual-machine;
  }
  image {
    file-path;
    bus-type [ide | virtio];
    image-type [qcow2 | raw];
  }
  init-descriptor file-path;
  memory {
    size size;
    features {
      hugepages;
    }
  }
  no-autostart;
  storage device-name {
    type {
      cdrom {
        source {
          file filename;
        }
      }
      disk {
        bus-type [ide | virtio];
          file-type [qcow2 | raw];
          source {
          file filename;
        }
      }
      usb {
        source {
          file filename;
        }
      }
    }
  }
  virtual-cpu {
    virtual-cpu-number {
      physical-cpu number | range;
    }
    count number;
    features {
      hardware-virtualization;
    }
  }
  interfaces interface-name {
    pci-address pci-address;
    mapping {
      hsxe0 {
```

```
        virtual-function {
          vlan-id vlan-id;
        }
      }
      hsxe1 {
        virtual-function {
          vlan-id vlan-id;
        }
      }
      vlan {
        members vlan-name;
        mode [access | trunk];
        native-vlan-id vlan-id;
      }
    }
  }
}
```

Hierarchy Level

```
[edit virtual-network-functions]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description

Name of the virtual network function.

Options

interfaces—Name of the interface. For example, em1.

no-autostart—An option to disable auto-start of VNF on the VNF configuration commit.

pci-address—An option to specify PCI address for the VNF interfaces.

mapping—An option to map VNF interfaces on platforms running disaggregated Junos OS.

virtual-cpu—An option to specify the number of virtual CPUs the VNF can use. By default, a VNF is assigned one virtual CPU, which is independent of any specific physical CPU.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112
- **show virtual-network-functions on page 165**

show virtual-network-functions

Syntax

```
show virtual-network-functions
show virtual-network-functions vnf-name
show virtual-network-functions ipsec-nm
show virtual-network-functions vjunos0
```

Release Information Command introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.

Description Display Virtual Network Function (VNF) information.

Options ***vnf-name***—(Optional) Display information for a specific VNF.

ipsec-nm—(Optional) Display information of the system VNF IPsec.

vjunos0—(Optional) Display information of the system VNF vjunos0.

brief—(Optional) Display brief output.

NOTE: **This is the default option.**

detail—(Optional) Display detailed output.

Required Privilege Level view

Related Documentation

- Understanding Virtual Network Functions on page 111
- Managing the VNF Life Cycle on page 112

List of Sample Output

Output Fields Table 22 on page 166 describes the output fields for the **show virtual-network-functions** command. Output fields are listed in the approximate order in which they appear.

Table 22: show virtual-network functions Output Fields

Field Name	Field Description
ID	ID of the VNF
Name	Name of the VNF
State	Status of the VNF. Possible values are Running, Shutdown, or Undefined.
Liveliness	Indicates whether or not the IP address of the VNF is reachable.
IP Address	IP address of the VNF
VCPUs	Number of virtual CPUs
Maximum Memory	Maximum amount of memory available to the VNF
Used Memory	Amount of memory used by the VNF
Used 1G Hugepages	The number of used 1G hugepages.
Used 2M Hugepages	The number of used 2M hugepages

Sample Output

show virtual-network-functions vjunos0

```
user@host> show virtual-network-functions vjunos0
ID       Name                                                State      Liveliness
------------------------------------------------------------------------------------
2        vjunos0                                             Running    alive
```

show virtual-network-functions vjunos0 detail

```
user@host> show virtual-network-functions vjunos0 detail
Virtual Network Function Information
------------------------------------

Id:                 3
Name:               vjunos0
State:              Running
Liveliness:         Up
IP Address:         192.0.2.2
VCPUs:              1
Maximum Memory:     1000448 KiB
Used Memory:        1000448 KiB
Used 1G Hugepages:  0
Used 2M Hugepages:  0

     Block Devices
     ----------------------------
```

```
    Target   | Source
    ---------------------------
    vda      | /junos/images/0/vjunos.img
    vdb      | /junos/images/0/vjunos-data.img
    hdc      | /junos/images/shared/swap-disk.img
    vdd      | /junos/images/0/vjunos-platform.img
    vde      | /junos/images/0/metadata-usb-re.img
```

Sample Output

show virtual-network-functions vsrx

```
user@host> show virtual-network-functions vsrx
ID       Name                                                State      Liveliness
-------------------------------------------------------------------------------
67       vsrx                                                Running    alive
```

Sample Output

show virtual-network-functions vmx

```
user@host> show virtual-network-functions vmx
ID       Name                                                State      Liveliness
-------------------------------------------------------------------------------
3       vjunos0                                               Running   alive
10      vm x                 Running  alive
11341  jdm                              Running  alive
```

show virtual-network-functions

```
user@host> show virtual-network-functions
ID       Name                                                State      Liveliness
-------------------------------------------------------------------------------
3        vjunos0                                             Running    alive
1        LTE-VM                                              Running    down
```

Sample Output

show virtual-network-functions detail

```
user@host> show virtual-network-functions detail
Virtual Network Function Information
------------------------------------

Id:                3
Name:              vjunos0
State:             Running
Liveliness:        Up
IP Address:        192.0.2.2
VCPUs:             1
Maximum Memory:    1000448 KiB
Used Memory:       1000448 KiB
Used 1G Hugepages:  0
Used 2M Hugepages:  0
```

```
    Block Devices
    ----------------------------
    Target   | Source
    ----------------------------
    vda      | /junos/images/0/vjunos.img
    vdb      | /junos/images/0/vjunos-data.img
    hdc      | /junos/images/shared/swap-disk.img
    vdd      | /junos/images/0/vjunos-platform.img
    vde      | /junos/images/0/metadata-usb-re.img

Virtual Network Function Information
-----------------------------------

Id:                 1
Name:               LTE-VM
State:              Running
Liveliness:         Down
IP Address:         -
VCPUs:              2
Maximum Memory:     122880 KiB
Used Memory:        122880 KiB
Used 1G Hugepages:  0
Used 2M Hugepages:  0

    Block Devices
    ----------------------------
    Target   | Source
    ----------------------------
    vdb      | /usr/share/juniper/LTE/lte_vm.latest
```

show vlans

Syntax show vlans *vlan-name*

Release Information Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.

Description Display the details about the VLANs.

Options ***vlan-name***—Display information for a specific VLAN.

brief | detail | extensive | terse—(Optional) Display the specified level of output.

Required Privilege Level view

List of Sample Output show vlans on page 169

Output Fields Table 22 on page 166 describes the output fields for the **show virtual-network-functions** command. Output fields are listed in the approximate order in which they appear.

Table 23: show virtual-network functions Output Fields

Field Name	Field Description
vlan-name	Display information for a specified VLAN
brief	Display brief output
detail	Display detailed output
extensive	Display extensive output
instance	Display information for a specified instance
interface	Name of interface for which to display table
logical-system	Name of logical systems

Sample Output

show vlans

```
root@jdm> show vlans

Routing instance        VLAN name             Tag         Interfaces
host-os                 vlan100               100
```

```
                                                        vsrx1_eth6.0
                                                        vsrx2_eth6.0
host-os                 vlan200-202-vlan-0200 200
                                                        vsrx1_eth7.0
                                                        vsrx2_eth7.0
host-os                 vlan200-202-vlan-0202 202
                                                        vsrx1_eth7.0
                                                        vsrx2_eth7.0
```

PART 5

Service Chaining

CHAPTER 7

Service Chaining

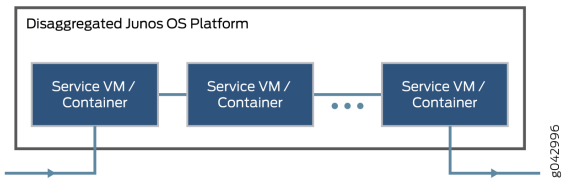
Understanding Service Chaining on Disaggregated Junos OS Platforms

In many network environments, it is common for traffic to flow through several *network services* on the way to its destination. These services—firewalls, Network Address Translators (NAT), load balancers, and so on—are generally spread across multiple network elements. Each device is a separate piece of hardware, providing a different service, and requiring separate operation and management. This method of linking together multiple network functions could be thought of as *physical service chaining*.

A more efficient model for service chaining is to virtualize and consolidate network functions onto a single device.

Platforms running the disaggregated Junos OS software support virtualized service chaining. These devices enable virtual network functions (VNFs) by supporting the installation and instantiation of VNFs. VNFs can be linked together to provide network services for traffic flowing through the device, as shown in "Virtual Network Functions" on page 109.

Figure 13: Virtual Network Functions on a Disaggregated Junos OS Platform



Related Documentation

- Understanding Disaggregated Junos OS on page 3
- Disaggregated Junos OS VMs on page 6
- Understanding Virtio Usage on page 8
- Understanding SR-IOV Usage on page 10
- Understanding Virtual Network Functions on page 111

Configuring Service Chaining Using VLANs

You can achieve service chaining using VLANs.

- Ensure that connectivity to the host is not lost during the configuration process.

To configure service chaining:

1. Create a VLAN. Use one of the following commands:
   - Create a VLAN without a VLAN ID. You can add only access ports to this VLAN:

     set host-os vlans *vlan-name* vlan-id none

   - Create a VLAN with a VLAN ID:

     set host-os vlans *vlan-name* vlan-id *vlan-id*

   - Create a VLAN using a list of VLAN IDs:

     set host-os vlans *vlan-name* vlan-id-list *vlan-id range | comma-separated list*

2. Attach an interface on the VNF to the VLAN:

   set virtual-network-functions *vnf-name* interfaces eth*x* mapping vlan mode [access|trunk]
   set virtual-network-functions *vnf-name* interfaces eth*x* mapping vlan members *list*

3. Attach a native VLAN ID to the VNF interface:

   set virtual-network-functions *vnf-name* interfaces eth*x* mapping vlan native-vlan-id *vlan-id*

Related Documentation

- Understanding Service Chaining on Disaggregated Junos OS Platforms on page 173
- Example: Configuring Service Chaining Using VLANs on NFX250 Network Services Platform on page 176

Configuring Service Chaining Using DHCP Services on VLANs

Using DHCP services, you need not manually configure the IP addresses on the VNF interfaces to achieve service-chaining. Enable DHCP clients on the glue bridge interfaces within the VNF for an IP address to be assigned from the DHCP pool. Based on the IP subnet, the IRB interface on the VLAN is automatically mapped to the corresponding subnet dhcp pool.

To configure service chaining:

1. Create a VLAN with a VLAN ID **none**.

   set host-os vlans *vlan-name* vlan-id none

   NOTE: To use the DHCP pooling feature, the VLAN ID must be set to none.

2. Create IRB interfaces on the hypervisor

   set host-os interfaces irb unit *logical-unit-number* family inet address *inet-address*
   set host-os vlans *vlan-name* l3-interface irb. *logical-interface-number*

3. Specify the IP address pool to be used:

   set access address-assignment pool p4 family inet network *network-address*
   set access address-assignment pool p4 family inet range r4 low *start-IP-address*
   set access address-assignment pool p4 family inet range r4 high *end-IP-address*

4. Attach an interface on the VNF to the VLAN to complete the service chain:

   set virtual-network-functions *vnf-name* interfaces eth*x* mapping vlan mode [access|trunk]
   set virtual-network-functions *vnf-name* interfaces eth*x* mapping vlan members *list*

5. Enable the DHCP client on the VNF.

To check the assigned IP address, use the **show system visibility vnf** command.

Related Documentation

- Understanding Service Chaining on Disaggregated Junos OS Platforms on page 173
- Example: Configuring Service Chaining Using VLANs on NFX250 Network Services Platform on page 176

Example: Configuring Service Chaining Using VLANs on NFX250 Network Services Platform

This example shows how to configure service chaining using VLANs on the host bridge.

Requirements

This example uses the following hardware and software components:

- NFX250 running Junos OS Release 15.1X53-D45

Before you configure service chaining, be sure you have:

- Installed and launched the relevant VNFs, assigned the corresponding interfaces, and configured the resources.

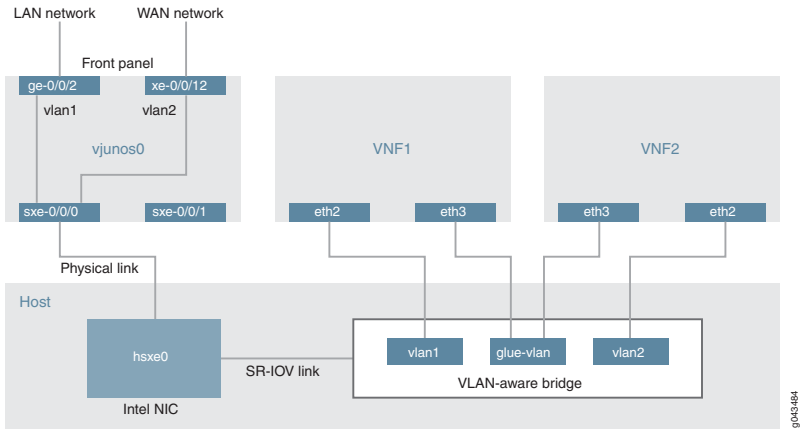
Overview

Service chaining on a device running the disaggregated Junos OS allows multiple services, or virtual network functions (VNFs), to be applied to traffic as it flows through the device. This example explains how to configure the various layers of the device to enable traffic to enter the device, flow through two service VNFs, and exit the device.

Topology

This example uses a single device running the disaggregated Junos OS, as shown in Figure 14 on page 176.

Figure 14: Service Chaining Using VLANs



This example is configured using the Juniper Device Manager (JDM) and Junos Control Plane (JCP). The key configuration elements include:

- The Packet Forwarding Engine's front panel ports.
- The Packet Forwarding Engine's internal-facing ports.
- A routing instance named *host-os*. The *host-os* routing instance is the CLI construct that provides the ability to configure host OS elements from the JDM.
- NIC ports. As these interfaces are not directly configurable, they are abstracted in the host OS. Using the JDM CLI, NIC interfaces (sxe ports) are configured in the *host-os* routing instance as "hsxe" interfaces.
- The VM interfaces. In the JDM, VNF interfaces must use the format eth#, where # is from 2 through to 9.
- VLANs, to provide bridging between the sxe and VM interfaces.

Configuration

Configuring the Packet Forwarding Engine Interfaces

CLI Quick Configuration

To quickly configure the Packet Forwarding Engine interfaces, enter the following configuration statements from the JCP:

```
[edit]
user@jcp#
```

```
set vlans vlan1 vlan-id 77
set interfaces ge-0/0/2.0 family ethernet-switching vlan members vlan1
set interfaces sxe-0/0/0.0 family ethernet-switching interface-mode trunk
set interfaces sxe-0/0/0.0 family ethernet-switching vlan members vlan1
set vlans vlan2 vlan-id 1177
set interfaces xe-0/0/12.0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/12.0 family ethernet-switching vlan members vlan2
set interfaces sxe-0/0/0.0 family ethernet-switching vlan members vlan2
```

Step-by-Step Procedure

To configure the Packet Forwarding Engine interfaces:

1. Connect to the JCP.

```
user@jdm> ssh vjunos0
user@jcp> configure
[edit]
user@jcp#
```

2. Configure a VLAN for the LAN-side interfaces.

```
user@jcp# set vlans vlan1 vlan-id 77
```

3. Configure the Packet Forwarding Engine's LAN-side front panel port and add it to the LAN-side VLAN.

   The LAN-side port is typically an access port, but could be a trunk port if appropriate.

```
user@jcp# set interfaces ge-0/0/2.0 family ethernet-switching vlan members vlan1
```

4. Configure the Packet Forwarding Engine's LAN-side internal-facing interface as a trunk port and add it to the LAN-side VLAN.

   The internal-facing interfaces are typically trunk ports, as they must support traffic from multiple front panel ports and VLANs.

```
user@jcp# set interfaces sxe-0/0/0.0 family ethernet-switching interface-mode trunk
user@jcp# set interfaces sxe-0/0/0.0 family ethernet-switching vlan members vlan1
```

5. Configure a VLAN for the WAN-side interfaces.

```
user@jcp# set vlans vlan2 vlan-id 1177
```

6. Configure the Packet Forwarding Engine's WAN-side front panel port as a trunk port and add it to the WAN-side VLAN.

   The WAN-side front panel port is typically a trunk port, as it might be required to support multiple VLANs.

```
user@jcp# set interfaces xe-0/0/12.0 family ethernet-switching interface-mode trunk
user@jcp# set interfaces xe-0/0/12.0 family ethernet-switching vlan members vlan2
```

7. Configure the Packet Forwarding Engine's WAN-side internal-facing interface as a trunk port and add it to the WAN-side VLAN.

   The internal-facing interfaces are typically trunk ports, as they must support traffic from multiple front panel ports and VLANs.

```
user@jcp# set interfaces sxe-0/0/0.0 family ethernet-switching vlan members vlan2
```

8. Commit the configuration and return to the JDM.

```
user@jcp# commit and-quit
user@jcp> exit
```

```
user@jdm>
```

Results From configuration mode, check the results of your configuration by entering the following **show** commands:

```
[edit]
user@jcp# show interfaces ge-0/0/2
unit 0 {
   family ethernet-switching {
      vlan {
         members vlan1;
      }
   }
}

[edit]
user@jcp# show interfaces xe-0/0/12
unit 0 {
   family ethernet-switching {
      interface-mode trunk;
      vlan {
         members vlan2;
      }
   }
}

[edit]
user@jcp# show interfaces sxe-0/0/0
unit 0 {
   family ethernet-switching {
      interface-mode trunk;
      vlan {
         members [vlan1 vlan2];
      }
   }
}

[edit]
user@jcp# show vlans
vlan2 {
   vlan-id 1177;
}
vlan1 {
   vlan-id 77;
}
```

Configuring the VNF Interfaces and Creating the Service Chain

Step-by-Step Procedure

Once you have completed the configuration on JCP, you need to:

1. Configure the host-os instance with either with LAN, WAN, or glue-vlan to be used for service chaining

```
user@jdm# set host-os vlans vlan1 vlan-id 77
user@jdm# set host-os vlans vlan2 vlan-id 1177
user@jdm# set host-os vlans glue-vlan vlan-id 123
```

2. Bring up the VM1 with one virtio interface mapped to VLAN, and another interface mapped to glue-vlan.

```
user@jdm# set virtual-network-functions VM1 interfaces eth2 mapping vlan members vlan1
user@jdm# set virtual-network-functions VM1 interfaces eth3 mapping vlan members
glue-vlan
```

3. Similarly bring up VM2 with one interface with one interface mapped to VLAN2, and the second interface mapped to the same glue-vlan.

```
user@jdm# set virtual-network-functions VM2 interfaces eth2 mapping vlan members vlan2
user@jdm# set virtual-network-functions VM2 interfaces eth3 mapping vlan members
glue-vlan
```

4. Finally, configure the IP addresses and static routes for each interface of the VMs as shown in Figure 14 on page 176.

Related Documentation

- Understanding Service Chaining on Disaggregated Junos OS Platforms on page 173
- Disaggregated Junos OS VMs on page 6
- Understanding Virtio Usage on page 8

Example: Configuring Service Chaining Using SR-IOV on NFX250 Network Services Platform

This example shows how to configure service chaining using SR-IOV on platforms running the disaggregated Junos OS software.

Requirements

This example uses the following hardware and software components:

- NFX250 running Junos OS Release 15.1X53-D45

Before you configure service chaining, be sure you have:

- Installed and launched the relevant VNFs

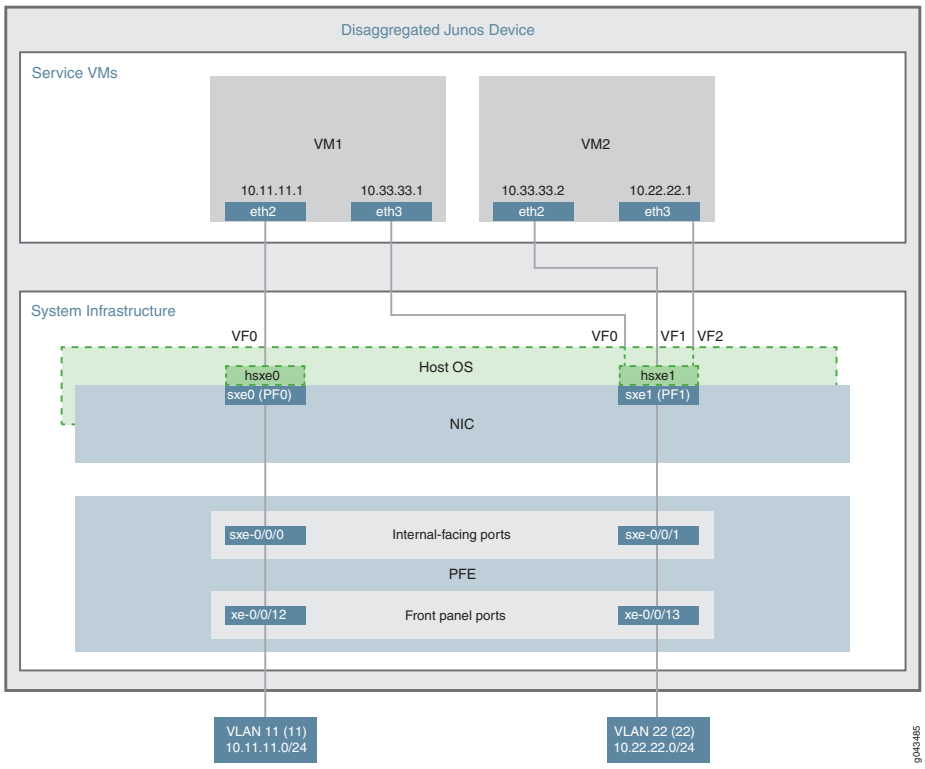
Overview

Service chaining on a device running the disaggregated Junos OS allows multiple services, or virtual network functions (VNFs), to be applied to traffic as it flows through the device. This example explains how to configure the various layers of the device to enable traffic to enter the device, flow through two service VNFs, and exit the device.

Topology

This example uses a single device running the disaggregated Junos OS, as shown in Figure 15 on page 181.

Figure 15: Service Chaining Using SR-IOV—Device Infrastructure



This example uses the Packet Forwarding Engine's front panel ports xe-0/0/12 and xe-0/0/13, and its internal-facing ports, sxe-0/0/0 and sxe-0/0/1. The internal NIC's two ports (sxe0 and sxe1) are not configured directly; instead, they are abstracted at the

host OS layer and configured as interfaces hsxe0 and hsxe1. The VMs use two interfaces each (eth2 and eth3).

These elements are generally separated into two parts: a *LAN side* and a *WAN side*.

As this example uses SR-IOV, the NIC ports' virtual functions (VFs) are used to bypass the host OS and provide direct NIC-to-VM connectivity. Given this setup, it might seem unusual to see host OS interfaces (hsxe0 and hsxe1) included in this scenario. However, as there is no direct configuration method for the NIC ports, it is necessary to use their abstracted versions, hsxe0 and hsxe1.

This example is configured using the Juniper Device Manager (JDM) and Junos Control Plane (JCP). The key configuration elements include:

- The Packet Forwarding Engine's front panel ports.
- The Packet Forwarding Engine's internal-facing ports.
- NIC ports. Because NIC interfaces (sxe ports) cannot be configured directly, the host OS construct for these interfaces (hsxe) must be used.
- The VNF interfaces. In the JDM, VNF interfaces must use the format eth#, where # is from 2 through to 9.
- The virtual function setting, to indicate SR-IOV is being used to provide direct access between hsxe and VNF interfaces.

Configuration

This example describes:

- Configuring the Packet Forwarding Engine Interfaces on page 182
- Creating the Service Chain on page 185

Configuring the Packet Forwarding Engine Interfaces

CLI Quick Configuration

To quickly configure the Packet Forwarding Engine interfaces, enter the following configuration statements from the JCP:

```
[edit]
user@jcp#
```

```
set vlans Vlan11 vlan-id 11
set interfaces xe-0/0/12.0 family ethernet-switching vlan member Vlan11
set interfaces sxe-0/0/0.0 family ethernet-switching interface-mode trunk
set interfaces sxe-0/0/0.0 family ethernet-switching vlan member Vlan11
set vlans Vlan22 vlan-id 22
set interfaces xe-0/0/13.0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/13.0 family ethernet-switching vlan member Vlan22
set interfaces sxe-0/0/1.0 family ethernet-switching interface-mode trunk
set interfaces sxe-0/0/1.0 family ethernet-switching vlan member Vlan22
```

Step-by-Step Procedure

To configure the Packet Forwarding Engine interfaces:

1. Connect to the JCP.

```
user@jdm> ssh vjunos0
user@jcp> configure
[edit]
user@jcp#
```

2. Configure a VLAN for the LAN-side interfaces.

```
user@jcp# set vlans Vlan11 vlan-id 11
```

3. Configure the Packet Forwarding Engine's LAN-side front panel port, and add it to the LAN-side VLAN.

   The LAN-side port is typically an access port, but could be a trunk port if appropriate.

```
user@jcp# set interfaces xe-0/0/12.0 family ethernet-switching vlan members Vlan11
```

4. Configure the Packet Forwarding Engine's LAN-side internal-facing interface as a trunk port, and add it to the LAN-side VLAN.

   The internal-facing interfaces are typically trunk ports, as they must support traffic from multiple front panel ports and VLANs.

```
user@jcp# set interfaces sxe-0/0/0.0 family ethernet-switching interface-mode trunk
user@jcp# set interfaces sxe-0/0/0.0 family ethernet-switching vlan member Vlan11
```

5. Configure a VLAN for the WAN-side interfaces.

```
user@jcp# set vlans Vlan22 vlan-id 22
```

6. Configure the Packet Forwarding Engine's WAN-side front panel port as a trunk port, and add it to the WAN-side VLAN.

   The WAN-side front panel port is typically a trunk port, as it might be required to support multiple VLANs.

```
user@jcp# user@jcp# set interfaces xe-0/0/13.0 family ethernet-switching interface-mode
trunk
user@jcp# user@jcp# set interfaces xe-0/0/13.0 family ethernet-switching vlan members
Vlan22
```

7. Configure the Packet Forwarding Engine's WAN-side internal-facing interface as a trunk port, and add it to the WAN-side VLAN.

   The internal-facing interfaces are typically trunk ports, as they must support traffic from multiple front panel ports and VLANs.

   ```
   user@jcp# set interfaces sxe-0/0/1.0 family ethernet-switching interface-mode trunk
   user@jcp# set interfaces sxe-0/0/1.0 family ethernet-switching vlan members Vlan22
   ```

8. Commit the configuration and return to the JDM.

   ```
   user@jcp# commit and-quit
   user@jcp> exit
   user@jdm>
   ```

Results From configuration mode, check the results of your configuration by entering the following **show** commands:

```
[edit]
user@jcp# show interfaces xe-0/0/12
unit 0 {
   family ethernet-switching {
      vlan {
         members Vlan11;
      }
   }
}

[edit]
user@jcp# show interfaces xe-0/0/13
unit 0 {
   family ethernet-switching {
      interface-mode trunk;
      vlan {
         members Vlan22;
      }
   }
}

[edit]
user@jcp# show interfaces sxe-0/0/0
unit 0 {
   family ethernet-switching {
      interface-mode trunk;
      vlan {
         members Vlan11;
      }
   }
}
```

```
[edit]
user@jcp# show interfaces sxe-0/0/1
unit 0 {
   family ethernet-switching {
      interface-mode trunk;
      vlan {
         members Vlan22;
      }
   }
}

[edit]
user@jcp# show vlans
Vlan11 {
   vlan-id 11;
}
Vlan22 {
   vlan-id 22;
}
```

Creating the Service Chain

Step-by-Step Procedure

To configure the VNF interfaces and create the service chain:

1. Configure VM1's LAN-side interface as a Layer 3 interface, and map it to the LAN-side NIC interface. Include the virtual function (VF) setting to specify direct NIC-to-VM connectivity. VNF must use the interfaces from eth2 through to eth9.

   The hsxe interface is the configurable representation of the related NIC (sxe) interface.

   ```
   user@jdm> configure
   [edit]
   user@jdm# set virtual-network-functions vnf1 interfaces eth2 mapping hsxe0 virtual-function
   ```

2. Configure VM1's WAN-side interface from sxe1 NIC as shown in Figure 15 on page 181.

   ```
   user@jdm# set virtual-network-functions vnf1 interfaces eth3 mapping hsxe1 virtual-function
   ```

3. Similarly bring up VM2 with both interfaces eth2 and eth3 on sxe1 NIC.

   ```
   user@jdm# set virtual-network-functions vnf2 interfaces eth2 mapping hsxe1 virtual-function
   user@jdm# set virtual-network-functions vnf2 interfaces eth3 mapping hsxe1 virtual-function
   ```

4. Finally, configure the IP addresses and static routes for each interface of the VNFs, and add routes to achieve the complete bidirectional path for the service chain.

Related Documentation

- Understanding Service Chaining on Disaggregated Junos OS Platforms on page 173
- Disaggregated Junos OS VMs on page 6

- Understanding SR-IOV Usage on page 10

PART 6

IPSec-NM

CHAPTER 8

Understanding IPSec-NM

Overview of IP Security

IP Security (IPSec) provides a secure way to authenticate senders and encrypt IP version 4 (IPv4) and version 6 (IPv6) traffic between network devices, such as routers and hosts. IPSec offers network administrators and their users the benefits of data confidentiality, data integrity, sender authentication, and anti-replay services. IPSec is increasingly becoming a critical component in today's contemporary IP networks.

IPSec is a framework for ensuring secure private communication over IP networks and is based on standards developed by the International Engineering Task Force (IETF). IPSec provides security services at the network layer of the Open Systems Interconnection (OSI) model by enabling a system to select required security protocols, determine the algorithms to use for the security services, and implement any cryptographic keys required to provide the requested services. You can use IPSec to protect one or more paths between a pair of hosts, between a pair of security gateways (such as routers), or between a security gateway and a host.

The native IPSec virtual private network (VPN) supported on JUNOS is used in various Juniper products to provide secure VPN connectivity. To address certain use cases, the IPSec VPN functionality depends on various JUNOS components and interworks across the modules. With the emergence of advanced technologies such as software-defined networking (SDN), network functions virtualization (NFV), and cloud services, Juniper IPSec VPN needed to be flexible with more efficient security solutions. To address such use cases, Juniper Networks introduced containerized SRX (cSRX) support and IPSec is also added to cSRX. Additionally, Juniper Networks introduced IP Security Network Manager (IPSec-NM) , which offers a security management solution by using IPSec in cSRX to protect management traffic flowing into Juniper VM.

The following features are supported on IPSec:

- Anti-replay services
- Internet Key Exchange (IKE) gateway
- Internet Key Exchange (IKE) v1 policy in Aggressive and Main mode with pre-shared key (PSK).
- One IKE security associations (SA) with multiple IPSec SA based on traffic selector.
- Traffic selector based tunnel establishment (not route based and no routing protocol over tunnel).
- Xauth client with config mode for internal IP attribute.
- key id, hostname, distinguished name, user@hostname, inet, and inet6 support as local and remote identity.
- Initiator to establish IPSec VPN tunnels immediately.
- IPv4 and IPv6 addresses for IPSec VPN tunnel source and destination.
- Encryption algorithms such as DES, 3DES, AES-128, and AES-256.
- Authentication algorithms such as MD5, SHA1, and SHA-256.
- Diffie-Hellman groups (dh-groups) such as 2, 5, 14, and 19.
- Dead peer detection (DPD)
- Perfect Forward Secrecy (PFS)
- NAT-T
- Tunnel mode
- Traffic selector based tunnel establishment

The terminology and components of IPSec can be intimidating to first-time users. However, if you learn a few key concepts, you can quickly master and deploy IPSec in your network. The main concepts you need to understand are as follows:

- *Authentication Algorithms*
- *Encryption Algorithms*
- *IPsec Protocols*
- *IPsec Security Associations*
- *IPSec Modes*

Configuring IP Security Network Manager

IP Security Network Manager (IPSec-NM) is a network management system that offers confidentiality, security, and authentication of data that is shared within a network. It provides data security at the IP layer of the network.

The following features are supported on IPSec-NM:

- *Understanding Address Books*
- *Understanding IKE and IPsec Packet Processing*
- *Hostname*
- *Static Routing*
- *Media MTU*
- *Root Password Encryption*
- *Security Zones Overview*
- *Host Inbound Traffic*

Configuring IPSec-NM Interfaces

To enable IPSec-NM on a LAN or WAN, you must configure interfaces to provide network connectivity and data flow.

NOTE: Ensure that connectivity to the host is not lost during the configuration process.

To configure IPSec-NM interface, complete the following steps:

1. Create a logical interface with a VLAN ID:

   `root@ipsec-nm#` **set interfaces *interface-name* unit *interface-logical-unit-number* vlan-id *vlan-id***

2. Assign an IPv4 address to the logical interface:

   `root@ipsec-nm#` **set interfaces *interface-name* unit *interface-logical-unit-number* family inet address *interface-address***

3. Assign an IPv6 address to the logical interface:

   `root@ipsec-nm#` **set interfaces *interface-name* unit *interface-logical-unit-number* family inet6 address *interface-address***

4. Enable VLAN tagging support on the logical interface:

   `root@ipsec-nm#` **set interfaces *interface-name* vlan-tagging**

Configuring AutoKey Internet Key Exchange

IPSec-NM supports the automated generation and negotiation of keys and security associations (SAs) using the Internet Key Exchange (IKE) protocol. This automation is termed as AutoKey IKE. Juniper Networks supports AutoKey IKE with pre-shared keys and certificates.

Dynamic SAs require IKE configuration. With dynamic SAs, you can configure IKE and then the SA. IKE creates the dynamic SAs and negotiates them for IPSec. The IKE configuration defines the algorithms and keys used to establish the secure IKE connection with the peer security gateway.

NOTE:

- Ensure that connectivity to the host is not lost during the configuration process.
- Ensure that the IPSec-NM interfaces are configured.

Configuring IKE Proposals

You can configure one or more IKE proposals. Each proposal is a list of IKE attributes to protect the IKE connection between the IKE host and its peer.

To configure IKE proposal, complete the following steps:

1. Define an IKE proposal:

   ```
   root@ipsec-nm# set security ike proposal ike-proposal-name authentication-method pre-shared-keys
   ```

2. Define a Diffie-Hellman group (dh-group) for the IKE proposal:

   ```
   root@ipsec-nm# set security ike proposal ike-proposal-name dh-group group2
   ```

3. Define an authentication algorithm for the IKE proposal:

   ```
   root@ipsec-nm# set security ike proposal ike-proposal-name authentication-algorithm sha1
   ```

4. Define an encryption algorithm for the IKE proposal:

   ```
   root@ipsec-nm# set security ike proposal ike-proposal-name encryption-algorithm aes-192-cbc
   ```

5. Set a lifetime for the IKE proposal in seconds:

   ```
   root@ipsec-nm# set security ike proposal ike-proposal-name lifetime-seconds 180 to 86400 seconds
   ```

Configuring IKE Policies

An IKE policy defines a combination of security parameters (IKE proposals) to be used during IKE negotiation. It defines a peer address and the proposals needed for that connection. Depending on which authentication method is used, it defines the preshared key for the given peer. During the IKE negotiation, IKE looks for an IKE policy that is the same on both peers. The peer that initiates the negotiation sends all its policies to the remote peer, and the remote peer tries to find a match.

A match is made when both policies from the two peers have a proposal that contains the same configured attributes. If the lifetimes are not identical, the shorter lifetime between the two policies (from the host and peer) is used. The configured preshared key must also match its peer.

The key management process (kmd) daemon determines which version of IKE is used in a negotiation. If kmd is the IKE initiator, it uses IKEv1 by default and retains the configured version for negotiations. If kmd is the IKE responder, it accepts connections from IKEv1.

You can create multiple, prioritized proposals at each peer to ensure that at least one proposal matches the proposal of a remote peer.

Initially, you must configure one or more IKE proposals and associate these proposals with an IKE policy. You can also prioritize a list of proposals used by IKE in the policy statement by listing the proposals you want to use, from first to last.

To configure IKE policy, complete the following steps:

1. Define an IKE policy with first phase mode:

   ```
   root@ipsec-nm# set security ike policy ike-policy-name mode aggressive
   ```

2. Define a set of IKE proposals:

   ```
   root@ipsec-nm# set security ike policy ike-policy-name proposals proposal-name
   ```

3. Define a pre-shared key for IKE:

   ```
   root@ipsec-nm# set security ike policy ike-policy-name pre-shared-key ascii-text text-format
   ```

Configuring IKE Gateway

An IKE gateway initiates and terminates network connections between a firewall and a security device.

To configure IKE gateway, complete the following steps:

1. Configure an IKE gateway with an IKE policy:

   ```
   root@host# set security ike gateway gateway-name ike-policy ike-policy-name
   ```

2. Configure an IKE gateway with an address or hostname of the peer:

```
root@host# set security ike gateway gateway-name address address-or-hostname-of-peer
```

3. Enable dead peer detection (DPD) feature to send DPD messages periodically:

```
root@host# set security ike gateway gateway-name dead-peer-detection always-send
```

4. Configure username of the xauth client:

```
root@ipsec-nm# set security ike gateway gateway-name xauth client username
xauth-client-username
```

5. Configure password of the xauth client:

```
root@ipsec-nm# set security ike gateway gateway-name xauth client password
xauth-client-password
```

6. Enable dead peer detection (DPD) feature to send DPD messages at a regular interval:

```
root@ipsec-nm# set security ike gateway gateway-name dead-peer-detection interval
10-to-60-seconds
```

7. Configure the maximum number of DPD retransmissions:

```
root@ipsec-nm# set security ike gateway gateway-name dead-peer-detection threshold
1-to-5
```

8. Configure an external interface for IKE negotiations:

```
root@ipsec-nm# set security ike gateway gateway-name external-interface ge-0/0/2
```

9. Configure the local IKE address:

```
root@ipsec-nm# set security ike gateway gateway-name local-address local-address
```

10. Configure the local IKE identity:

```
root@ipsec-nm# set security ike gateway gateway-name local-identity <inet | inet6 | key-id
| hostname | user-at-hostname | distinguished-name>
```

11. Set the version of the IKE protocol:

```
root@ipsec-nm# set security ike gateway gateway-name version v1-only
```

Configuring IKE Trace Options

Trace options is used for debugging and managing the IPSec IKE.

To configure IPSec IKE trace options, complete the following steps:

1. Provide the name of the file in which trace information has to be written:

```
root@ipsec-nm# set security ike traceoptions file file-name
```

2. Specify the maximum size of the trace file:

```
root@ipsec-nm# set security ike traceoptions file size file-size
```

3. Specify the parameters to trace information for IKE:

```
root@ipsec-nm# set security ike traceoptions flag all
```

Related Documentation

- ike on page 210

Configuring IPSec

IPSec is a suite of related protocols for cryptographically securing communications at the IP Packet Layer. IPSec also provides methods for the manual and automatic negotiation of security associations (SAs) and key distribution, all the attributes for which are gathered in a domain of interpretation (DOI). The IPSec DOI is a document containing definitions for all the security parameters and attributes required for SA and IKE negotiations. See RFC 2407 and RFC 2408 for more information.

Ensure that connectivity to the host is not lost during the configuration process.

Configuring IPSec Proposals

An IPSec proposal lists protocols and algorithms or security services to be negotiated with the remote IPSec peer.

To configure IPSec proposals, complete the following steps:

1. Define an IPSec proposal and protocol for the proposal:

```
root@ipsec-nm# set security ipsec proposal ipsec-proposal-name protocol esp
```

2. Define an authentication algorithm for the IPSec proposal:

```
root@ipsec-nm# set security ipsec proposal ipsec-proposal-name authentication-algorithm
hmac-sha1-96
```

3. Define an encryption algorithm for the IPSec proposal:

```
root@ipsec-nm# set security ipsec proposal ipsec-proposal-name encryption-algorithm
aes-256-cbc
```

4. Set a lifetime for the IPSec proposal in seconds:

```
root@ipsec-nm# set security ipsec proposal ipsec-proposal-name lifetime-seconds 180..86400
seconds
```

Configuring IPSec Policies

An IPSec policy defines a combination of security parameters (IPSec proposals) used during IPSec negotiation. It defines Perfect Forward Secrecy (PFS) and the proposals needed for the connection. During the IPSec negotiation, IPSec searches for a proposal

that is the same on both peers. The peer that initiates the negotiation sends all its policies to the remote peer, and the remote peer tries to find a match.

A match is made when both policies from both the peers have a proposal that contains the same configured attributes. If the lifetime is not identical, the shorter lifetime between the two policies (from the host and peer) is used.

You can create multiple, prioritized IPSec proposals at each peer to ensure that at least one proposal matches the proposal of the remote peer.

Initially, you must configure one or more IPSec proposals and then associate these proposals with an IPSec policy. You can prioritize a list of proposals used by IPSec in the policy statement by listing the proposals you want to use, from first to last.

To configure IPSec policies, complete the following steps:

1. Define an IPSec policy, a perfect forward secrecy, and a Diffie-Hellman group for the policy:

   ```
   root@ipsec-nm# set security ipsec policy ipsec-policy-name perfect-forward-secrecy keys group2
   ```

2. Define a set of IPSec proposals for the policy:

   ```
   root@ipsec-nm# set security ipsec policy ipsec-policy-name proposals proposal-name
   ```

Configuring IPSec Virtual Private Network

A virtual private network (VPN) provides a means for securely communicating among remote computers across a public WAN such as the Internet. A VPN connection can link two LANs (site-to-site VPN) or a remote dial-up user and a LAN. The traffic that flows between these two points passes through shared resources such as routers, switches, and other network equipment that make up the public WAN. To secure VPN communication while passing through the WAN, the two participants create an IP Security (IPsec) tunnel. For more information, see *IPsec VPN Overview*.

To configure IPSec VPN, complete the following steps:

1. Define an IKE-keyed IPSec VPN:

   ```
   root@ipsec-nm# set security ipsec vpn vpn-name ike gateway remote-gateway-name
   ```

2. Define an IPSec policy for the IPSec VPN:

   ```
   root@ipsec-nm# set security ipsec vpn vpn-name ike ipsec-policy ipsec-policy-name
   ```

3. Define a local traffic selector for the IPSec VPN:

   ```
   root@ipsec-nm# set security ipsec vpn vpn-name traffic-selector traffic-selector-name local-ip local-traffic-selector-ip-address
   ```

4. Define a remote traffic selector for the IPSec VPN:

   ```
   root@ipsec-nm# set security ipsec vpn vpn-name traffic-selector traffic-selector-name
   remote-ip remote-traffic-selector-ip-address
   ```

5. Define a criteria to establish IPSec VPN tunnels:

   ```
   root@ipsec-nm# set security ipsec vpn vpn-name establish-tunnels immediately
   ```

6. Configure default action and permit all traffic if the user-defined policy does not match:

   ```
   root@ipsec-nm# set security policies default-policy permit-all
   ```

Related Documentation

- *IPsec VPN Overview*

Example: Configuring IKE, IPSec, and Security Zones

The master administrator is responsible for assigning an interface to a user logical system and configuring IKE, IPSec SAs, and security zones. This example shows how to assign an interface to a user logical system and configure IKE, IPSec SAs, and security zone parameters.

Requirements

Before you begin:

- Log in to the master logical system as the master administrator. See "*Understanding the Master Logical Systems and the Master Administrator Role*.
- Read "Overview of IP Security" on page 189 and "Configuring IP Security Network Manager" on page 190 topics.

Overview

In this example you configure IKE, IPSec SAs, and security zones. This example configures the parameters that are described in Table 24 on page 198.

Table 24: IKE, IPSec SAs, and Security Zones Configuration

Feature	Name	Configuration Parameters
IKE traceoptions	ike traceoptions	• file kmd • file size 10m • flag all • level 15
IKE proposal	IKE_PROP	• authentication-method pre-shared-keys • dh-group group14 • authentication-algorithm sha-256 • authentication-algorithm sha-256 • lifetime-seconds 3600
IKE policy	IKE_POL	• mode aggressive • proposals IKE_PROP • pre-shared-key ascii-text <enter psk>
IKE gateway	GW1	• ike-policy IKE_POL • address 2.2.2.2 • local-identity user-at-hostname "r0r2_store1@juniper.net" • external-interface ge-0/0/0 • local-address 3.3.3.2 • version v1-only
IPSec traceoptions	ipsec traceoptions	flag all
IPSec proposal	IPSEC_PROP	• protocol esp • authentication-algorithm hmac-sha-256-128 • encryption-algorithm aes-256-cbc • lifetime-seconds 2600
IPSec policy	IPSEC_POL	• perfect-forward-secrecy keys group14l • proposals IPSEC_PROP
IPSec VPN	VPN1	• ike gateway GW1 • ike ipsec-policy IPSEC_POL • traffic-selector VPN1_TS1 local-ip 51.0.1.0/24 • traffic-selector VPN1_TS1 remote-ip 41.0.1.0/24 • establish-tunnels immediately
flow	tcp-mss	all-tcp mss 1300
policies	default-policy	permit-all
zones	security-zone	• trust • untrust

Table 24: IKE, IPSec SAs, and Security Zones Configuration (continued)

Feature	Name	Configuration Parameters
interfaces	ge-0/0/0	• unit 0 vlan-id 100 • unit 0 family inet address 3.3.3.2/24 • unit 0 family inet6 address 3000::1/64 • vlan-tagging
	ge-0/0/1	• unit 0 vlan-id 4088 • unit 0 family inet address 51.0.1.1/24 • unit 0 family inet6 address 5000::1/64 • vlan-tagging
Routing options	routing-options	static route 2.2.2.0/24 next-hop 21.1.1.2

Configuration

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.

```
set security ike traceoptions file kmd
set security ike traceoptions file size 10m
set security ike traceoptions flag all
set security ike traceoptions level 15
set security ike proposal IKE_PROP authentication-method pre-shared-keys
set security ike proposal IKE_PROP dh-group group14
set security ike proposal IKE_PROP authentication-algorithm sha-256
set security ike proposal IKE_PROP encryption-algorithm aes-256-cbc
set security ike proposal IKE_PROP lifetime-seconds 3600
set security ike policy IKE_POL mode aggressive
set security ike policy IKE_POL proposals IKE_PROP
set security ike policy IKE_POL pre-shared-key ascii-text <enter psk>
set security ike gateway GW1 ike-policy IKE_POL
set security ike gateway GW1 address 2.2.2.2
set security ike gateway GW1 local-identity user-at-hostname "r0r2_store1@juniper.net"
set security ike gateway GW1 external-interface ge-0/0/0
set security ike gateway GW1 local-address 3.3.3.2
set security ike gateway GW1 version v1-only
set security ipsec traceoptions flag all
set security ipsec proposal IPSEC_PROP protocol esp
set security ipsec proposal IPSEC_PROP authentication-algorithm hmac-sha-256-128
set security ipsec proposal IPSEC_PROP encryption-algorithm aes-256-cbc
set security ipsec proposal IPSEC_PROP lifetime-seconds 2600
set security ipsec policy IPSEC_POL perfect-forward-secrecy keys group14
set security ipsec policy IPSEC_POL proposals IPSEC_PROP
set security ipsec vpn VPN1 ike gateway GW1
set security ipsec vpn VPN1 ike ipsec-policy IPSEC_POL
set security ipsec vpn VPN1 traffic-selector VPN1_TS1 local-ip 51.0.1.0/24
set security ipsec vpn VPN1 traffic-selector VPN1_TS1 remote-ip 41.0.1.0/24
```

```
set security ipsec vpn VPN1 establish-tunnels immediately
set security flow tcp-mss all-tcp mss 1300
set security policies default-policy permit-all
set security zones security-zone trust
set security zones security-zone untrust
set interfaces ge-0/0/0 unit 0 vlan-id 100
set interfaces ge-0/0/0 unit 0 family inet address 3.3.3.2/24
set interfaces ge-0/0/0 unit 0 family inet6 address 3000::1/64
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/1 unit 0 vlan-id 4088
set interfaces ge-0/0/1 unit 0 family inet address 51.0.1.1/24
set interfaces ge-0/0/1 unit 0 family inet6 address 5000::1/64
set interfaces ge-0/0/1 vlan-tagging
set routing-options static route 2.2.2.0/24 next-hop 21.1.1.2
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure IKE, IPSec SAs, and security zones:

1. Log in to an IPSec-NM device and enter configuration mode.

```
[edit]
root@ipsec-nm> configure
root@ipsec-nm#
```

2. Configure IKE traceoptions:

```
[edit security ike]
root@ipsec-nm# set security ike traceoptions file kmd
root@ipsec-nm# set security ike traceoptions file size 10m
root@ipsec-nm# set security ike traceoptions flag all
root@ipsec-nm# set security ike traceoptions level 15
```

3. Configure an IKE proposal:

```
[edit security ike]
root@ipsec-nm# set security ike proposal IKE_PROP authentication-method
  pre-shared-keys
root@ipsec-nm# set security ike proposal IKE_PROP dh-group group14
root@ipsec-nm# set security ike proposal IKE_PROP authentication-algorithm
  sha-256
root@ipsec-nm# set security ike proposal IKE_PROP encryption-algorithm
  aes-256-cbc
root@ipsec-nm# set security ike proposal IKE_PROP lifetime-seconds 3600
```

4. Configure an IKE policy:

```
[edit security ike]
```

```
root@ipsec-nm# set security ike policy IKE_POL mode aggressive
root@ipsec-nm# set security ike policy IKE_POL proposals IKE_PROP
root@ipsec-nm# set security ike policy IKE_POL pre-shared-key ascii-text <enter
  psk>
```

5. Configure an IKE gateway.

```
[edit security ike]
root@ipsec-nm# set security ike gateway GW1 ike-policy IKE_POL
root@ipsec-nm# set security ike gateway GW1 address 2.2.2.2
root@ipsec-nm# set security ike gateway GW1 local-identity user-at-hostname
  "r0r2_store1@juniper.net"
root@ipsec-nm# set security ike gateway GW1 external-interface ge-0/0/0
root@ipsec-nm# set security ike gateway GW1 local-address 3.3.3.2
root@ipsec-nm# set security ike gateway GW1 version v1-only
```

6. Configure IPSec traceoptions:

```
[edit security ipsec]
root@ipsec-nm# set security ipsec traceoptions flag all
```

7. Configure an IPSec proposal.

```
[edit security ipsec]
root@ipsec-nm# set security ipsec proposal IPSEC_PROP protocol esp
root@ipsec-nm# set security ipsec proposal IPSEC_PROP authentication-algorithm
  hmac-sha-256-128
root@ipsec-nm# set security ipsec proposal IPSEC_PROP encryption-algorithm
  aes-256-cbc
root@ipsec-nm# set security ipsec proposal IPSEC_PROP lifetime-seconds 2600
```

8. Configure an IPSec policy.

```
[edit security ipsec]
root@ipsec-nm# set security ipsec policy IPSEC_POL perfect-forward-secrecy keys
  group14
root@ipsec-nm# set security ipsec policy IPSEC_POL proposals IPSEC_PROP
```

9. Configure the IPSec VPN.

```
[edit security ipsec]
root@ipsec-nm# set security ipsec vpn VPN1 ike gateway GW1
root@ipsec-nm# set security ipsec vpn VPN1 ike ipsec-policy IPSEC_POL
root@ipsec-nm# set security ipsec vpn VPN1 traffic-selector VPN1_TS1 local-ip
  51.0.1.0/24
root@ipsec-nm# set security ipsec vpn VPN1 traffic-selector VPN1_TS1 remote-ip
  41.0.1.0/24
root@ipsec-nm# set security ipsec vpn VPN1 establish-tunnels immediately
```

10. Configure security flow:

```
[edit security]
root@ipsec-nm# set security flow tcp-mss all-tcp mss 1300
```

11. Configure security policies:

```
[edit security]
root@ipsec-nm# set security policies default-policy permit-all
```

12. Configure security zones:

```
[edit security]
root@ipsec-nm# set security zones security-zone trust
root@ipsec-nm# set security zones security-zone untrust
```

13. Configure interfaces for IPSec-NM:

```
[edit]
root@ipsec-nm# set interfaces ge-0/0/0 unit 0 vlan-id 100
root@ipsec-nm# set interfaces ge-0/0/0 unit 0 family inet address 3.3.3.2/24
root@ipsec-nm# set interfaces ge-0/0/0 unit 0 family inet6 address 3000::1/64
root@ipsec-nm# set interfaces ge-0/0/0 vlan-tagging
root@ipsec-nm# set interfaces ge-0/0/1 unit 0 vlan-id 4088
root@ipsec-nm# set interfaces ge-0/0/1 unit 0 family inet address 51.0.1.1/24
root@ipsec-nm# set interfaces ge-0/0/1 unit 0 family inet6 address 5000::1/64
root@ipsec-nm# set interfaces ge-0/0/1 vlan-tagging
```

14. Configure routing options:

```
[edit]
root@ipsec-nm# set routing-options static route 2.2.2.0/24 next-hop 21.1.1.2
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show security ike**, and **show security ipsec** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
root@ipsec-nm# show security ike
   traceoptions {
      file kmd;
      file size 10m;
      flag all;
      level 15;
   }
      proposal IKE_PROP {
         authentication-method pre-shared-keys;
         dh-group group14;
```



```
        authentication-algorithm sha-256;
        encryption-algorithm aes-256-cbc;
        lifetime-seconds 3600;
      }
      policy IKE_POL {
        mode aggressive;
        proposals IKE_PROP;
        pre-shared-key ascii-text <enter psk>;
      }
      gateway GW1 {
        ike-policy IKE_POL;
        address 2.2.2.2;
        local-identity user-at-hostname "r0r2_store1@juniper.net";
        external-interface ge-0/0/0;
        local-address 3.3.3.2;
        version v1-only;
      }
[edit]
root@ipsec-nm# show security ipsec
  traceoptions {
    flag all;
  }
  proposal IPSEC_PROP {
    protocol esp;
    authentication-algorithm hmac-sha-256-128;
    encryption-algorithm aes-256-cbc;
    lifetime-seconds 2600;
  }
  policy IPSEC_POL {
    perfect-forward-secrecy keys group14;
    proposals IPSEC_PROP;
  }
  vpn VPN1 {
    ike gateway GW1;
    ike ipsec-policy IPSEC_POL;
    traffic-selector VPN1_TS1 {
      local-ip 51.0.1.0/24;
      remote-ip 41.0.1.0/24;
    }
    establish-tunnels immediately;
  }
[edit]
root@ipsec-nm# show security flow
  tcp-mss {
    all-tcp mss 1300;
  }
[edit]
root@ipsec-nm# show security policies
  default-policy {
    permit-all;
  }
[edit]
root@ipsec-nm# show security zones
  security-zone {
    trust;
```

```
      untrust;
    }
[edit]
root@ipsec-nm# show interfaces
    ge-0/0/0 unit 0 {
      vlan-id 100;
        family inet address 3.3.3.2/24;
        family inet6 address 3000::1/64;
      vlan-tagging;
    }
    ge-0/0/1 unit 0 {
      vlan-id 4088;
        family inet address 51.0.1.1/24;
        family inet6 address 5000::1/64;
      vlan-tagging;
    }
[edit]
root@ipsec-nm# show routing-options
    static route 2.2.2.0/24 {
      next-hop 21.1.1.2;
    }
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying the Configuration

Purpose Verify that the IKE, IPSec SA, and security zones configuration is correct.

Action From operational mode, enter the **show security ike**, **show security ipsec**, **show security flow**, **show security policies**, **show security zones**, **show interfaces**, and **show routing-options** commands.

Related Documentation

- Configuring IPSec-NM Interfaces on page 191
- Configuring AutoKey Internet Key Exchange on page 192
- Configuring IPSec on page 195

CHAPTER 9

IPSec-NM Configuration Statements and Operational Commands

ipsec-nm

Syntax

```
ipsec-nm {
  configuration {
    security {
      ike {
        proposal ike-proposal-name {
          authentication-method {
            pre-shared-keys;
          }
          authentication-algorithm {
            md5;
              sha-256;
              sha-384;
            sha1;
          }
          dh-group {
            group1;
              group14;
              group2;
            group5;
          }
          encryption-algorithm {
            3des-cbc;
              aes-128-cbc;
              aes-192-cbc;
              aes-256-cbc;
            des-cbc;
          }
          lifetime-seconds lifetime-in-seconds;
        }
        policy ike-policy-name {
          mode {
            aggressive;
          }
          proposals proposal-name;
          pre-shared-key {
            ascii-text;
            hexadecimal;
          }
        }
        gateway gateway-name {
          ike-policy ike-policy-name;
            address address-or-hostname-of-peer;
            dead-peer-detection {
            always-send;
              interval interval-in-seconds;
            threshold maximum-number-of-DPD-retransmissions;
          }
          external-interface external-interface;
            local-address local-ip-address;
            version {
            v1-only;
```

```
        }
      }
    }
    ipsec {
      proposal ipsec-proposal-name {
        protocol {
          ah;
          esp;
        }
        authentication-algorithm {
          hmac-md5-96;
            hmac-sha-256-128;
          hmac-sha1-96;
        }
        encryption-algorithm {
          3des-cbc;
            aes-128-cbc;
            aes-192-cbc;
            aes-256-cbc;
          des-cbc;
        }
        lifetime-seconds lifetime-in-seconds;
      }
      policy ipsec-policy-name {
        perfect-forward-secrecy {
          keys {
            group1;
              group14;
              group2;
            group5;
          }
        }
        proposals proposal-name;
      }
      vpn vpn-name {
        ike {
          gateway remote-gateway-name;
          ipsec-policy ipsec-policy-name;
        }
        traffic-selector {
          traffic-selector-name1 {
            local-ip local-traffic-selector-ip-address;
            remote-ip remote-traffic-selector-ip-address;
          }
          traffic-selector-name2 {
            local-ip local-traffic-selector-ip-address;
            remote-ip remote-traffic-selector-ip-address;
          }
        }
        establish-tunnels immediately;
      }
    }
    policies {
      from-zone {
        from-zone-name {
```

```
            to-zone to--zone-name;
          }
          trust {
            to-zone to--zone-name;
              trust;
            untrust;
          }
          untrust {
            to-zone to--zone-name;
              trust;
            untrust;
          }
        }
      }
      zones {
        security-zone {
          security-zone-name {
            trust {
              host-inbound-traffic {
                system-services {
                  all;
                }
                protocols {
                  all;
                }
              }
            }
            untrust {
              host-inbound-traffic {
                system-services {
                  all;
                }
                protocols {
                  all;
                }
              }
            }
          }
        }
      }
    }
    interfaces {
      ge-0/0/0 {
        mtu maximum-transmit-packet-size;
          unit interface-unit-number;
        vlan-tagging;
      }
      ge-0/0/1 {
        mtu maximum-transmit-packet-size;
          unit interface-unit-number;
        vlan-tagging;
      }
    }
  }
}
```

Hierarchy Level [ipsec-nm configuration security]

Release Information Statement introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description Provides confidentiality, security, and authentication of data that is shared within a network. It also provides data security at the IP layer of the network.

Options

proposal ***ike-proposal-name***—Name of the IKE proposal.

lifetime-seconds ***lifetime-in-seconds***—Lifetime in seconds.

policy ***ipsec-policy-name***—Name of the IPSec policy.

gateway ***remote-gateway-name***—Name of the remote gateway.

ike-policy ***ike-policy-name***—Name of the IKE policy.

mtu ***maximum-transmit-packet-size***—Packet size of maximum transmit.

unit ***interface-unit-number***—Interface unit number.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring IP Security Network Manager on page 190

ike

Syntax

```
ike {
  proposal ike-proposal-name {
    authentication-method {
      dsa-signatures;
        pre-shared-keys;
      rsa-signatures;
    }
    authentication-algorithm {
      md5;
        sha-256;
        sha-384;
      sha1;
    }
    dh-group {
      group1;
        group14;
        group2;
      group5;
    }
    encryption-algorithm {
      3des-cbc;
        aes-128-cbc;
        aes-128-gcm;
        aes-192-cbc;
        aes-256-cbc;
        aes-256-gcm;
      des-cbc;
    }
    lifetime-seconds lifetime-in-seconds;
  }
  policy ike-policy-name {
    mode {
      aggressive;
      main;
    }
    proposals proposal-name;
    pre-shared-key {
      ascii-text;
      hexadecimal;
    }
  }
  gateway gateway-name {
    ike-policy ike-policy-name;
      address address-or-hostname-of-peer;
      dead-peer-detection {
      always-send;
        interval interval-in-seconds;
      threshold maximum-number-of-DPD-retransmissions;
    }
    external-interface external-interface;
      local-address local-ip-address;
```

```
        version {
        v1-only;
        v2-only;
      }
    }
    traceoptions {
      file {
        filename;
        size file-size;
      }
      flag all;
    }
}
```

Hierarchy Level [ipsec-nm configuration security]

Release Information Statement introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description IPSec-NM supports the automated generation and negotiation of keys and security associations (SAs) using the Internet Key Exchange (IKE) protocol. This automation is termed as AutoKey IKE. Juniper Networks supports AutoKey IKE with pre-shared keys and certificates. For more information, see "Configuring AutoKey Internet Key Exchange" on page 192.

Options

proposal *ike-proposal-name*—Name of the IKE proposal.

lifetime-seconds *lifetime-in-seconds*—Lifetime in seconds.

proposals *ipsec-policy-name*—Name of the proposals.

gateway *remote-gateway-name*—Name of the remote gateway.

ike-policy *ike-policy-name*—Name of the IKE policy.

traceoptions—Option to trace information for the IPSec key management.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring AutoKey Internet Key Exchange on page 192

ipsec

Syntax

```
ipsec {
  proposal ipsec-proposal-name {
    protocol {
      ah;
      esp;
    }
    authentication-algorithm {
      hmac-md5-96;
        hmac-sha-256-128;
      hmac-sha1-96;
    }
    lifetime-seconds lifetime-in-seconds;
  }
  policy ipsec-policy-name {
    perfect-forward-secrecy {
      keys {
        group1;
          group14;
          group2;
        group5;
      }
    }
    proposals proposal-name;
  }
  vpn vpn-name {
    ike {
      gateway remote-gateway-name;
      ipsec-policy ipsec-policy-name;
    }
    traffic-selector {
      traffic-selector-name1 {
        local-ip local-traffic-selector-ip-address;
        remote-ip remote-traffic-selector-ip-address;
      }
      traffic-selector-name2 {
        local-ip local-traffic-selector-ip-address;
        remote-ip remote-traffic-selector-ip-address;
      }
    }
    establish-tunnels immediately;
  }
}
```

Hierarchy Level

```
[ipsec-nm configuration security]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description IPSec is a suite of related protocols for cryptographically securing communications at the IP Packet Layer. IPSec also provides methods for the manual and automatic negotiation of security associations (SAs) and key distribution, all the attributes for which are gathered in a domain of interpretation (DOI). The IPSec DOI is a document containing definitions for all the security parameters and attributes required for SA and IKE negotiations. See RFC 2407 and RFC 2408 for more information. For more information, see "Configuring IPSec" on page 195.

Options

proposal *ipsec-proposal-name*—Name of the IPSec proposal.

lifetime-seconds *lifetime-in-seconds*—Lifetime in seconds.

policy *ipsec-policy-name*—Name of the IPSec policy.

gateway *remote-gateway-name*—Name of the remote gateway.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring IPSec on page 195

policies

Syntax

```
policies {
  from-zone {
    from-zone-name {
      to-zone to--zone-name;
    }
    trust {
      to-zone to--zone-name;
        trust;
      untrust;
    }
    untrust {
      to-zone to--zone-name;
        trust;
      untrust;
    }
  }
}
```

Hierarchy Level

```
[ipsec-nm configuration security]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description

You can configure network security policies for IPSec-NM.

Options

from-zone —Define a policy context from this zone.

from-zone-name—Name of the source zone.

to-zone —Destination zone.

to-zone—Name of the source zone.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- ipsec-nm on page 206

interfaces

Syntax

```
interfaces {
  ge-0/0/0 {
    mtu maximum-transmit-packet-size;
      unit interface-unit-number;
    vlan-tagging;
  }
  ge-0/0/1 {
    mtu maximum-transmit-packet-size;
      unit interface-unit-number;
    vlan-tagging;
  }
}
```

Hierarchy Level

```
[ipsec-nm configuration security]
```

Release Information

Statement introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description

You can configure interfaces for IPSec-NM.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- ipsec-nm on page 206

show security ike sa

Syntax

```
show security ike sa
show security ike sa detail
```

Release Information Command introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description Display information about the Internet Key Exchange (IKE) Security Association (SA).

Required Privilege Level view

Related Documentation

- ike on page 210

List of Sample Output

show security ike sa on page 219
show security ike sa detail on page 219

Output Fields Table 25 on page 216 lists the output fields for the **show security ike sa** command and Table 26 on page 217 lists the output fields for the **show security ike sa detail** command. Output fields are listed in the approximate order in which they appear.

Table 25: show security ike sa Output Fields

Field Name	Field Description
Index	Index number of an SA. This number is an internally generated number you can use to display information about a single SA.
State	State of the IKE SAs: • **DOWN** - SA has not been negotiated with the peer. • **UP** - SA has been negotiated with the peer.
Initiator cookie	Random number, called a cookie, which is sent to the remote node when the IKE negotiation is triggered.
Responder cookie	Random number generated by the remote node and sent back to the initiator as a verification that the packets were received. A cookie is aimed at protecting the computing resources from attack without spending excessive CPU resources to determine the cookie's authenticity.
Mode	Mode determines the number of messages and the payload types that are contained in each message that is exchanged by the two IPsec endpoints, or peers.
Remote Address	Address of the remote peer.

Table 26: show security ike sa detail Output Fields

Field Name	Field Description
IKE peer	IP address of the destination peer with which the local peer communicates.
Index	Index number of an SA. This number is an internally generated number you can use to display information about a single SA.
Gateway Name	Name of the IKE gateway.
Role	Part played in the IKE session. The device triggering the IKE negotiation is the initiator, and the device accepting the first IKE exchange packets is the responder.
State	State of the IKE SAs: • **DOWN** - SA has not been negotiated with the peer. • **UP** - SA has been negotiated with the peer.
Initiator cookie	Random number, called a cookie, which is sent to the remote node when the IKE negotiation is triggered.
Responder cookie	Random number generated by the remote node and sent back to the initiator as a verification that the packets were received. A cookie is aimed at protecting the computing resources from attack without spending excessive CPU resources to determine the cookie's authenticity.
Exchange type	Negotiation method agreed on by the two IPsec endpoints, or peers, used to exchange information between one another. Each exchange type or mode determines the number of messages and the payload types that are contained in each message. The modes are: • **main** - The exchange is done with six messages. This mode encrypts the payload, protecting the identity of the neighbor. • **aggressive** - The exchange is done with three messages. This mode does not encrypt the payload, leaving the identity of the neighbor unprotected.
Authentication method	Method used to authenticate the source of IKE messages, which can be either Pre-shared-keys or digital certificates, such as DSA-signatures, ECDSA-signatures-256, ECDSA-signatures-384, or RSA-signatures.
Local	Address of the local peer.
Remote	Address of the remote peer.
Lifetime	Number of seconds remaining until the IKE SA expires.
Reauth Lifetime	When enabled, number of seconds remaining until re-authentication triggers a new IKEv2 SA negotiation.

Table 26: show security ike sa detail Output Fields (continued)

Field Name	Field Description
IKE Fragmentation	**Enabled** means that both the IKEv2 initiator and responder support message fragmentation and have negotiated the support during the IKE_SA_INIT message exchange. **Size** shows the maximum size of an IKEv2 message before it is fragmented.
Remote Access Client Info	Information of the remote access client.
Peer ike-id	ID of the IKE peer.
Algorithms	
Authentication	Types of authentication algorithm used to encrypt and secure exchanges between the peers during the IPsec Phase 2 process: • **sha1** – Secure Hash Algorithm 1 authentication. • **md5** - MD5 authentication. • **sha-256** - Secure Hash Algorithm 256 authentication. • **sha-384** - Secure Hash Algorithm 384 authentication.
Encryption	Types of encryption algorithm used to encrypt and secure exchanges between the peers during the IPsec Phase 2 process: • **aes-256-cbc** - Advanced Encryption Standard (AES) 256-bit encryption. • **aes-192-cbc** - AES192-bit encryption. • **aes-128-cbc** - AES 128-bit encryption. • **3des-cbc** - 3 Data Encryption Standard (DES) encryption. • **des-cbc** - DES encryption.
Diffie-Hellman group	Specifies the IKE Diffie-Hellman group.
Traffic Statistics	
Input bytes	Number of bytes received.
Output bytes	Number of bytes transmitted.
Input packets	Number of packets received.
Output packets	Number of packets transmitted.
Input fragmentated packets	Number of IKEv2 fragmented packets received.
Output fragmentated packets	Number of IKEv2 fragmented packets transmitted.
IPSec security associations	• *number* created: The number of SAs created. • *number* deleted: The number of SAs deleted.

Table 26: show security ike sa detail Output Fields (continued)

Field Name	Field Description

Phase 2 negotiations in progress	Number of Phase 2 IKE negotiations in progress and status information:
	• **Negotiation type** - Type of Phase 2 negotiation. Junos OS currently supports quick mode.
	• **Message ID** - Unique identifier for a Phase 2 negotiation.
	• **Local identity** - Identity of the local Phase 2 negotiation. The format is *id-type-name (proto-name:port-number,[0..id-data-len] = iddata-presentation)*.
	• **Remote identity** - Identity of the remote Phase 2 negotiation. The format is *id-type-name (proto-name:port-number,[0..id-data-len] = iddata-presentation)*.
	• **Flags** - Notification to the key management process of the status of the IKE negotiation:
	  ◦ **caller notification sent** - Caller program notified about the completion of the IKE negotiation.
	  ◦ **waiting for done** - Negotiation is done. The library is waiting for the remote end retransmission timers to expire.
	  ◦ **waiting for remove** - Negotiation has failed. The library is waiting for the remote end retransmission timers to expire before removing this negotiation.
	  ◦ **waiting for policy manager** - Negotiation is waiting for a response from the policy manager.

Sample Output

show security ike sa

```
user@jdm> show security ike sa
Index   State  Initiator cookie  Responder cookie  Mode           Remote Address

7796166 UP     a1a6b1516bc43d54  f0846e4239c817f8  Aggressive     2.2.2.2
```

show security ike sa detail

```
user@jdm> show security ike sa detail
IKE peer 2.2.2.2, Index 7796166, Gateway Name: GW1
  Role: Initiator, State: UP
  Initiator cookie: a1a6b1516bc43d54, Responder cookie: f0846e4239c817f8
  Exchange type: Aggressive, Authentication method: Pre-shared-keys
  Local: 3.3.3.2:500, Remote: 2.2.2.2:500
  Lifetime: Expires in 3585 seconds
  Reauth Lifetime: Disabled
  IKE Fragmentation: Disabled, Size: 0
  Remote Access Client Info: Unknown Client
  Peer ike-id: 2.2.2.2
  AAA assigned IP: 0.0.0.0
  Algorithms:
   Authentication        : hmac-sha256-128
   Encryption            : aes256-cbc
   Pseudo random function: hmac-sha256
   Diffie-Hellman group  : DH-group-14
  Traffic statistics:
   Input  bytes  :                1056
```

```
 Output bytes  :                1311
 Input  packets:                   2
 Output packets:                   4
 Input  fragmentated packets:      0
 Output fragmentated packets:      0
IPSec security associations: 1 created, 0 deleted
Phase 2 negotiations in progress: 1

  Negotiation type: Quick mode, Role: Initiator, Message ID: 0
  Local: 3.3.3.2:500, Remote: 2.2.2.2:500
  Local identity: r0r2_store1@juniper.net
  Remote identity: 2.2.2.2
  Flags: IKE SA is created
```

show security ike active-peer

Syntax show security ike active-peer

Release Information Command introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description Display information about IKE active peers.

Required Privilege Level view

Related Documentation
- ike on page 210

List of Sample Output show security ike active-peer on page 221

Output Fields Table 27 on page 221 lists the output fields for the **show security ike active-peer** command. Output fields are listed in the approximate order in which they appear.

Table 27: show security ike active-peer Output Fields

Field Name	Field Description
Remote Address	Address of the remote peer.
Port	Network post assigned to the IKE active peer.
Peer IKE-ID	ID of the IKE peer.
AAA username	Authentication, Authorization, and Accounting (AAA) username.
Assigned IP	IP address assigned to the IKE active peer.

Sample Output

show security ike active-peer

```
user@jdm> show security ike active-peer
Remote Address        Port     Peer IKE-ID      AAA username         Assigned IP
2.2.2.2               500      2.2.2.2          not available        0.0.0.0
```

show security ipsec sa

Syntax

```
show security ipsec sa
show security ike sa detail
```

Release Information Command introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description Display information about the IPSec Security Association (SA).

Required Privilege Level view

Related Documentation

- ipsec on page 212

List of Sample Output

show security ipsec sa on page 225
show security ipsec sa detail on page 225

Output Fields Table 28 on page 222 lists the output fields for the **show security ipsec sa** command and Table 29 on page 223 lists the output fields for the **show security ipsec sa detail** command. Output fields are listed in the approximate order in which they appear.

Table 28: show security ipsec sa Output Fields

Field Name	Field Description
Total active tunnels	Total number of active IPsec tunnels.
ID	Index number of the SA. You can use this number to get additional information about the SA.
Algorithm	Cryptography used to secure exchanges between peers during the IKE Phase 2 negotiations includes: • An authentication algorithm used to authenticate exchanges between the peers. Options are **hmac-md5-96**, **hmac-sha-256-128**, or **hmac-sha1-96**. • An encryption algorithm used to encrypt data traffic. Options are **3des-cbc**, **aes-128-cbc**, **aes-192-cbc**, **aes-256-cbc**, or **des-cbc**.
SPI	Security parameter index (SPI) identifier. An SA is uniquely identified by an SPI. Each entry includes the name of the VPN, the remote gateway address, the SPIs for each direction, the encryption and authentication algorithms, and keys. The peer gateways each have two SAs, one resulting from each of the two phases of negotiation: Phase 1 and Phase 2.
Life:sec/kb	The lifetime of the SA, after which it expires, expressed either in seconds or kilobytes.

Table 28: show security ipsec sa Output Fields (continued)

Field Name	Field Description
Mon	The Mon field refers to VPN monitoring status. If VPN monitoring is enabled, then this field displays U (up) or D (down). A hyphen (-) means VPN monitoring is not enabled for this SA. A V means that IPSec datapath verification is in progress.
lsys	The root system.
Port	If Network Address Translation (NAT) is used, this value is 4500. Otherwise, it is the standard IKE port, 500.
Gateway	Gateway address of the system.

Table 29: show security ipsec sa detail Output Fields

Field Name	Field Description
ID	Index number of the SA. You can use this number to get additional information about the SA.
Virtual-system	The virtual system name.
VPN Name	IPSec name for VPN.
Local Gateway	Gateway address of the local system.
Remote Gateway	Gateway address of the remote system.
Traffic Selector Name	Name of the traffic selector.
Local Identity	Identity of the local peer so that its partner destination gateway can communicate with it. The value is specified as an IP address, fully qualified domain name, e-mail address, or distinguished name (DN).
Remote Identity	IP address of the destination peer gateway.
Version	IKE version. For example, IKEv1, IKEv2.
DF-bit	State of the don't fragment bit: **set** or **cleared**.
Bind-interface	The tunnel interface to which the route-based VPN is bound.
Tunnel Events	
Direction	Direction of the SA; it can be inbound or outbound.
AUX-SPI	Value of the auxiliary security parameter index(SPI). • When the value is **AH** or **ESP**, **AUX-SPI** is always 0. • When the value is **AH+ESP**, **AUX-SPI** is always a positive integer.

Table 29: show security ipsec sa detail Output Fields (continued)

Field Name	Field Description
VPN Monitoring	If VPN monitoring is enabled, then the **Mon** field displays **U (up)** or **D (down)**. A hyphen (-) means VPN monitoring is not enabled for this SA. A V means that IPSec datapath verification is in progress.
Hard lifetime	The hard lifetime specifies the lifetime of the SA. • **Expires in seconds** - Number of seconds left until the SA expires.
Lifesize Remaining	The lifesize remaining specifies the usage limits in kilobytes. If there is no lifesize specified, it shows unlimited.
Soft lifetime	The soft lifetime informs the IPsec key management system that the SA is about to expire. Each lifetime of an SA has two display options, hard and soft, one of which must be present for a dynamic SA. This allows the key management system to negotiate a new SA before the hard lifetime expires. • **Expires in seconds** - Number of seconds left until the SA expires.
Mode	Mode of the SA: • **transport** - Protects host-to-host connections. • **tunnel** - Protects connections between security gateways.
Type	Type of the SA: • **manual** - Security parameters require no negotiation. They are static and are configured by the user. • **dynamic** - Security parameters are negotiated by the IKE protocol. Dynamic SAs are not supported in transport mode.
State	State of the SA: • **Installed** - The SA is installed in the SA database. • **Not Installed** - The SA is not installed in the SA database. For transport mode, the value of State is always Installed.

Protocol	Protocol supported.
	• Transport mode supports Encapsulation Security Protocol (ESP) and Authentication Header (AH).
	• Tunnel mode supports ESP and AH.
	  ◦ **Authentication** - Type of authentication used.
	  ◦ **Encryption** - Type of encryption used.

Anti-replay service	State of the service that prevents packets from being replayed. It can be **Enabled** or **Disabled**.
Replay window size	Configured size of the antireplay service window. It can be 32 or 64 packets. If the replay window size is 0, the antireplay service is disabled. The antireplay window size protects the receiver against replay attacks by rejecting old or duplicate packets.

Sample Output

show security ipsec sa

```
user@jdm> show security ipsec sa
Total active tunnels: 1
  ID    Algorithm       SPI      Life:sec/kb  Mon lsys Port  Gateway
  <67109793 ESP:aes-cbc-256/sha256 e651d79e 2578/ unlim - root 500 2.2.2.2

  >67109793 ESP:aes-cbc-256/sha256 8ac9ce8 2578/ unlim - root 500 2.2.2.2
```

show security ipsec sa detail

```
user@jdm> show security ipsec sa detail
ID: 67109793 Virtual-system: root, VPN Name: VPN1
  Local Gateway: 3.3.3.2, Remote Gateway: 2.2.2.2
  Traffic Selector Name: VPN1_TS1
  Local Identity: ipv4(51.0.1.0-51.0.1.255)
  Remote Identity: ipv4(41.0.1.0-41.0.1.255)
  Version: IKEv1
  DF-bit: clear, Copy-Outer-DSCP Disabled, Bind-interface: st0.1
  Port: 500, Nego#: 0, Fail#: 0, Def-Del#: 0 Flag: 0x2c608b29
  Tunnel events:
    Wed Aug 16 2017 23:50:07 -0700: IPSec SA negotiation successfully completed
(1 times)
    Wed Aug 16 2017 23:50:07 -0700: IKE SA negotiation successfully completed (1
 times)
    Wed Aug 16 2017 23:49:46 -0700: Negotiation failed with error code
AUTHENTICATION_FAILED received from peer (2 times)
    Wed Aug 16 2017 23:49:30 -0700: Tunnel is ready. Waiting for trigger event
or peer to trigger negotiation (1 times)
  Direction: inbound, SPI: e651d79e, AUX-SPI: 0, VPN Monitoring: -
    Hard lifetime: Expires in 2552 seconds
    Lifesize Remaining:  Unlimited
    Soft lifetime: Expires in 1988 seconds
    Mode: Tunnel(0 0), Type: dynamic, State: installed
   Protocol: ESP, Authentication: hmac-sha256-128, Encryption: aes-cbc (256 bits)

    Anti-replay service: counter-based enabled, Replay window size: 64
  Direction: outbound, SPI: 8ac9ce8, AUX-SPI: 0, VPN Monitoring: -
    Hard lifetime: Expires in 2552 seconds
    Lifesize Remaining:  Unlimited
    Soft lifetime: Expires in 1988 seconds
    Mode: Tunnel(0 0), Type: dynamic, State: installed
   Protocol: ESP, Authentication: hmac-sha256-128, Encryption: aes-cbc (256 bits)

    Anti-replay service: counter-based enabled, Replay window size: 64
```

show security ipsec statistics

Syntax show security ipsec statistics

Release Information Command introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description Display IPSec statistics.

Required Privilege Level view

Related Documentation
- ipsec on page 212

List of Sample Output show security ipsec statistics on page 227

Output Fields Table 30 on page 226 lists the output fields for the **show security ipsec statistics** command. Output fields are listed in the approximate order in which they appear.

Table 30: show security ipsec statistics

Field Name	Field Description
ESP Statistics	
Encrypted bytes	Total number of bytes encrypted by the local system across the IPSec tunnel.
Decrypted bytes	Total number of bytes decrypted by the local system across the IPSec tunnel.
Encrypted packets	Total number of packets encrypted by the local system across the IPSec tunnel.
Decrypted packets	Total number of packets decrypted by the local system across the IPSec tunnel.
AH Statistics	
Input bytes	Total number of bytes received by the local system across the IPSec tunnel.
Output bytes	Total number of bytes transmitted by the local system across the IPSec tunnel.
Input packets	Total number of packets received by the local system across the IPSec tunnel.
Output packets	Total number of packets transmitted by the local system across the IPSec tunnel.
Errors	
AH authentication failures	Total number of authentication header (AH) failures. An AH failure occurs when there is a mismatch of the authentication header in a packet transmitted across an IPSec tunnel.

Table 30: show security ipsec statistics (continued)

Field Name	Field Description
Replay errors	Total number of replay errors. A replay error is generated when a duplicate packet is received within the replay window.
ESP authentication failures	Total number of Encapsulation Security Payload (ESP) failures. An ESP failure occurs when there is an authentication mismatch in ESP packets.
ESP decryption failures	Total number of ESP decryption errors.
Bad headers	Total number of invalid headers detected.
Bad trailers	Total number of invalid trailers detected.

Sample Output

show security ipsec statistics

```
user@jdm> show security ipsec statistics
ESP Statistics:
  Encrypted bytes:          265920
  Decrypted bytes:          249360
  Encrypted packets:           240
  Decrypted packets:           240
AH Statistics:
  Input bytes:                   0
  Output bytes:                  0
  Input packets:                 0
  Output packets:                0
Errors:
  AH authentication failures: 0, Replay errors: 0
  ESP authentication failures: 0, ESP decryption failures: 0
  Bad headers: 0, Bad trailers: 0
```

show security ipsec inactive-tunnels

Syntax show security ipsec inactive-tunnels

Release Information Command introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description Display information about IPSec tunnels that are inactive on a disaggregated Junos OS platform.

Required Privilege Level view

Related Documentation
- ipsec on page 212

List of Sample Output show security ipsec inactive-tunnels on page 228

Output Fields Table 31 on page 228 lists the output fields for the **show security ipsec inactive-tunnels** command. Output fields are listed in the approximate order in which they appear.

Table 31: show security ipsec inactive-tunnels Output Fields

Field Name	Field Description
Total inactive tunnels	Total number of inactive IPSec tunnels.
Total inactive tunnels with establish immediately	Total number of inactive IPsec tunnels that can establish a session immediately.
ID	Identification number of the inactive tunnel. You can use this number to get more information about the inactive tunnel.
Port	If Network Address Translation (NAT) is used, this value is 4500. Otherwise, it is the standard IKE port, 500.
Gateway	IP address of the remote gateway.
Tunnel Down Reason	Reason for which the tunnel is inactive.

Sample Output

show security ipsec inactive-tunnels

```
user@jdm> show security ipsec inactive-tunnels
```

```
Total inactive tunnels: 1
  Total inactive tunnels with establish immediately: 1
  ID           Port   Gateway          Tunnel Down Reason
```

```
  67109793     500    2.2.2.2          Negotiation failed with error code
AUTHENTICATION_FAILED received from peer (2 times)
```

show security ipsec tunnel-events-statistics

Syntax show security ipsec inactive-tunnels

Release Information Command introduced in Junos OS Release 15.1X53-D47 for the NFX250 Network Services Platform.

Description Display tunnel event statistics.

Required Privilege Level view

Related Documentation
- ipsec on page 212

List of Sample Output show security ipsec tunnel-events-statistics on page 230

Sample Output

show security ipsec tunnel-events-statistics

```
user@jdm> show security ipsec tunnel-events-statistics
DPD detected peer as down. Existing IKE/IPSec SAs cleared
     : 4
Tunnel configuration is deleted. Corresponding IKE/IPSec SAs are deleted
     : 4
Tunnel configuration changed. Corresponding IKE/IPSec SAs are deleted
     : 4
No response from peer. Negotiation failed
     : 6273
Negotiation failed with error code AUTHENTICATION_FAILED received from peer
     : 2
IKE SA negotiation successfully completed
     : 4
IPSec SA negotiation successfully completed
     : 13
Tunnel is ready. Waiting for trigger event or peer to trigger negotiation
     : 5
```