



JDM User Guide for NFX250 Network Services Platform



Modified: 2016-12-22

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

Juniper Networks, Junos, Steel-Belted Radius, NetScreen, and ScreenOS are registered trademarks of Juniper Networks, Inc. in the United States and other countries. The Juniper Networks Logo, the Junos logo, and JunosE are trademarks of Juniper Networks, Inc. All other trademarks, service marks, registered trademarks, or registered service marks are the property of their respective owners.

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

JDM User Guide for NFX250 Network Services Platform
Copyright © 2016, Juniper Networks, Inc.
All rights reserved.

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 <http://www.juniper.net/support/eula.html>. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

	About the Documentation	xi
	Documentation and Release Notes	xi
	Using the Examples in This Manual	xi
	Merging a Full Example	xii
	Merging a Snippet	xii
	Documentation Conventions	xiii
	Documentation Feedback	xv
	Requesting Technical Support	xv
	Self-Help Online Tools and Resources	xv
	Opening a Case with JTAC	xvi
Part 1	Architecture Overview	
Chapter 1	Architecture Overview	3
	Understanding Disaggregated Junos OS	3
	Disaggregated Junos OS VMs	5
	Understanding Virtio Usage	8
	Understanding SR-IOV Usage	10
	Comparing Virtio and SR-IOV	11
	Understanding Physical and Virtual Components	12
Part 2	Installation	
Chapter 2	Installation	17
	Managing Software Installation on NFX250 Network Services Platform	17
	Upgrading an Image on the Disaggregated Junos OS Platform	18
	Reverting the System to the Factory-Default Configuration	21
	Rebooting the System	21
Part 3	Management	
Chapter 3	Management	25
	Understanding the JDM CLI	26
	Accessing the JDM Shell, JDM CLI, and JCP Prompts in a Disaggregated Junos OS Platform	26
	Accessing the JDM CLI	27
	Accessing the JDM Shell	27
	Accessing the JCP Prompt from the JDM CLI	27
	Accessing the Hypervisor from the JDM CLI	28

Accessing the ipsec-nm from the JDM CLI	28
Understanding User Accounts	28
Root Account	28
Other User Accounts	29
User Authentication	29
Configuring JDM User Accounts and Authentication	29
Understanding JDM Management Interfaces	30
Console Interface	30
Out-of-Band Management Interface	30
In-Band Management Interface	31
Configuring the Out-of-Band Management Interface for JDM	31
Configuring the Out-of-Band Management Interface with IPv4 Addressing for JDM	32
Configuring the Out-of-Band Management Interface with IPv6 Addressing for JDM	32
Configuring the In-Band Management Interface for JDM	33
Configuring the Out-of-Band Management Interface for Hypervisor	34
Configuring the Out-of-Band Management Interface with IPv4 Addressing for Hypervisor	35
Configuring the Out-of-Band Management Interface with IPv6 Addressing for Hypervisor	35
Configuring SSH Service and NETCONF-Over-SSH Connections for Remote Access to the Disaggregated Junos OS Platform	35
Configuring HTTP Access to the Disaggregated Junos OS Platform	36
Configuring HTTPS Access to the Disaggregated Junos OS Platform	36
Configuring SNMP on JDM	36
Configuring SNMP Community	37
Configuring SNMP System Parameters	37
Configuring SNMP v3	37
Configuring SNMP Traps	38
Querying SNMP MIBs	38
Managing Traps	38
Configuring Enhanced Orchestration in the Disaggregated Junos OS Platform	38
Configuring IPSec in the Disaggregated Junos OS Platform	39
Viewing and Managing Centralized Log Files in a Disaggregated Junos OS Platform	39
Enabling Centralized Logging	39
Viewing Log Messages	40
Managing Core Files for a Disaggregated Junos OS Platform	40
Viewing Core Files	40
Chapter 4 Management Commands	43
enhanced-orchestration	44
http	44
https	44
ipsec-nm	45
netconf	45
outbound-ssh	46

	phone-home	47
	rest	48
	ssh	48
	system	49
	traceoptions	51
	upgrade-image-before-configuration	51
	show system inventory hardware cpu	52
	show system inventory hardware memory	55
	show system inventory hardware network	57
	show system inventory hardware storage	59
	show system inventory software vnf	62
	show system services ipsec-nm	63
	show system visibility cpu	64
	show system visibility host	67
	show system visibility jcp	72
	show system visibility jdm	75
	show system visibility memory	79
	show system visibility network	81
	show system visibility storage	84
	show system visibility vnf	87
Part 4	Virtual Network Functions	
Chapter 5	Virtual Network Functions	95
	Understanding Virtual Network Functions	95
	Managing the VNF Life Cycle	96
	Planning Resources for a VNF	97
	Managing the VNF Image	98
	Preparing the Bootstrap Configuration	98
	Launching a VNF	98
	Allocating Resources for a VNF	99
	Specifying CPU for VNF	99
	Allocating Memory for a VNF	99
	Configuring VNF Storage Devices	100
	Configuring VNF Interfaces and VLANs	100
	Managing VNF States	101
	Managing VNF MAC Addresses	102
	Accessing a VNF from JDM	102
	Viewing List of VNFs	102
	Displaying the VNF Details	102
	Deleting a VNF	103
Chapter 6	Virtual Network Functions Commands	105
	features	106
	hugepages	106
	image	107
	init-descriptor	108
	interfaces (JDM)	109
	ipsec-nm	110
	mapping	111

	memory	112
	no-autostart	112
	pci-address	113
	size	113
	storage	114
	type	115
	virtual-cpu	116
	virtual-network-functions	117
	vjunos0	120
	vnf-name	121
	show virtual-network-functions	123
	show vlans	125
Part 5	Service Chaining	
Chapter 7	Service Chaining	129
	Understanding Service Chaining on Disaggregated Junos OS Platforms	129
	Configuring Service Chaining Using VLANs	130
	Configuring Service Chaining Using DHCP Services on VLANs	131
	Example: Configuring Service Chaining Using VLANs on NFX250 Network Services Platform	131
	Example: Configuring Service Chaining Using SR-IOV on NFX250 Network Services Platform	136
Part 6	Index	
	Index	145

List of Figures

Part 1	Architecture Overview	
Chapter 1	Architecture Overview	3
	Figure 1: Position of the Juniper Device Manager	4
	Figure 2: Basic Disaggregated Junos OS Architecture	4
	Figure 3: Virtual Machine Monitors	6
	Figure 4: Containers—Overall Architecture	7
	Figure 5: VNF Bridging with Virtio	9
	Figure 6: VNF Communication Using SR-IOV	10
	Figure 7: Physical and Virtual Layers in the Disaggregated Junos OS	13
	Figure 8: Physical and Virtual Component Communication	14
Part 3	Management	
Chapter 3	Management	25
	Figure 9: Out-of-band Management Interface	31
	Figure 10: In-Band Management Interface Network	31
	Figure 11: In-Band Management Interface Example	33
Part 4	Virtual Network Functions	
Chapter 5	Virtual Network Functions	95
	Figure 12: Network Connections Between JDM and the VMs	95
Part 5	Service Chaining	
Chapter 7	Service Chaining	129
	Figure 13: Virtual Network Functions on a Disaggregated Junos OS Platform	129
	Figure 14: Service Chaining Using VLANs	132
	Figure 15: Service Chaining Using SR-IOV—Device Infrastructure	137

List of Tables

	About the Documentation	xi
	Table 1: Notice Icons	xiii
	Table 2: Text and Syntax Conventions	xiii
Part 3	Management	
Chapter 4	Management Commands	43
	Table 3: show system inventory hardware cpu Output Fields	52
	Table 4: show system inventory hardware memory Output Fields	55
	Table 5: show system inventory hardware network Output Fields	57
	Table 6: show system inventory hardware storage Output Fields	59
	Table 7: show system inventory software vnf Output Fields	62
	Table 8: show system services ipsec-nm Output Fields	63
	Table 9: show system visibility cpu Output Fields	64
	Table 10: show system visibility host Output Fields	67
	Table 11: show system visibility jcp Output Fields	72
	Table 12: show system visibility jdm Output Fields	75
	Table 13: show system visibility memory Output Fields	79
	Table 14: show system visibility network Output Fields	81
	Table 15: show system visibility storage Output Fields	84
	Table 16: show system visibility vnf Output Fields	87
Part 4	Virtual Network Functions	
Chapter 5	Virtual Network Functions	95
	Table 17: VNF Glossary	96
	Table 18: Physical CPU Allocation for NFX250-LS1	97
	Table 19: Physical CPU Allocation for NFX250	97
Chapter 6	Virtual Network Functions Commands	105
	Table 20: show virtual-network functions Output Fields	123
	Table 21: show virtual-network functions Output Fields	125

About the Documentation

- Documentation and Release Notes on page xi
- Using the Examples in This Manual on page xi
- Documentation Conventions on page xiii
- Documentation Feedback on page xv
- Requesting Technical Support on page xv

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 <http://www.juniper.net/techpubs/>.

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 <http://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.xml;
    }
  }
}
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.xml; }
```

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 xiii 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 xiii 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

Table 2: Text and Syntax Conventions (*continued*)

Convention	Description	Examples
<i>Italic text like this</i>	<ul style="list-style-type: none">Introduces or emphasizes important new terms.Identifies guide names.Identifies RFC and Internet draft titles.	<ul style="list-style-type: none">A policy <i>term</i> is a named structure that defines match conditions and actions.<i>Junos OS CLI User Guide</i>RFC 1997, <i>BGP Communities Attribute</i>
<i>Italic text like this</i>	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 <i>domain-name</i>
Text like this	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	<ul style="list-style-type: none">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 <i>metric</i> >;
(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 (<i>string1</i> <i>string2</i> <i>string3</i>)
# (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 [<i>community-ids</i>]
Indentation and braces ({ })	Identifies a level in the configuration hierarchy.	[edit] routing-options { static { route default { nexthop <i>address</i> ; retain; } } }
;(semicolon)	Identifies a leaf statement at a configuration hierarchy level.	
GUI Conventions		
Bold text like this	Represents graphical user interface (GUI) items you click or select.	<ul style="list-style-type: none">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, comments, and suggestions so that we can improve the documentation. You can provide feedback by using either of the following methods:

- Online feedback rating system—On any page of the Juniper Networks TechLibrary site at <http://www.juniper.net/techpubs/index.html>, simply click the stars to rate the content, and use the pop-up form to provide us with information about your experience. Alternately, you can use the online feedback form at <http://www.juniper.net/techpubs/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 <http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf>.
- Product warranties—For product warranty information, visit <http://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: <http://www.juniper.net/customers/support/>
- Search for known bugs: <http://www2.juniper.net/kb/>
- Find product documentation: <http://www.juniper.net/techpubs/>
- Find solutions and answer questions using our Knowledge Base: <http://kb.juniper.net/>
- Download the latest versions of software and review release notes: <http://www.juniper.net/customers/csc/software/>
- Search technical bulletins for relevant hardware and software notifications: <http://kb.juniper.net/InfoCenter/>

- Join and participate in the Juniper Networks Community Forum:
<http://www.juniper.net/company/communities/>
- Open a case online in the CSC Case Management tool: <http://www.juniper.net/cm/>

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

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 <http://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 <http://www.juniper.net/support/requesting-support.html>.

PART 1

Architecture Overview

- [Architecture Overview on page 3](#)

CHAPTER 1

Architecture Overview

- [Understanding Disaggregated Junos OS on page 3](#)
- [Disaggregated Junos OS VMs on page 5](#)
- [Understanding Virtio Usage on page 8](#)
- [Understanding SR-IOV Usage on page 10](#)
- [Comparing Virtio and SR-IOV on page 11](#)
- [Understanding Physical and Virtual Components on page 12](#)

Understanding Disaggregated Junos OS

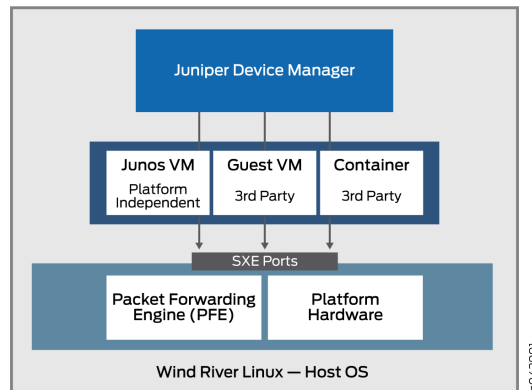
Many network equipment vendors have traditionally bound their software to purpose-built hardware and sold customers the bundled and packaged software–hardware combination. However, with the disaggregated Junos OS architecture, Juniper Network devices are now aligned with networks that are cloud-oriented,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

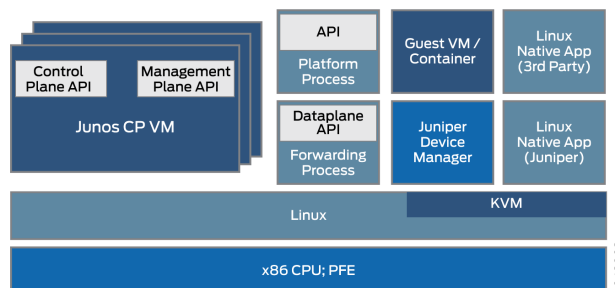
JDM enables:

- 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, Wireshark, 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 5](#)
- [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,

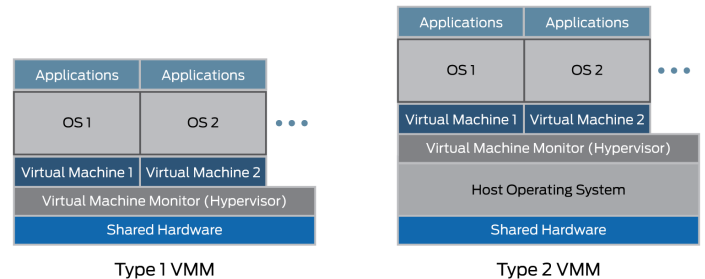
but also extension and migration of workloads at times of heavy use (when extension can be used) or physical maintenance (when migration can be used).

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 compiled 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.

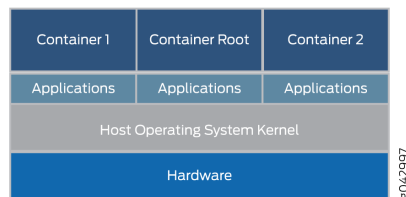
The biggest advantage of VNFs is that most applications can be easily ported to the hypervisor environment and run well without modification.

The biggest drawback is that, often the resource-intensive 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 is 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

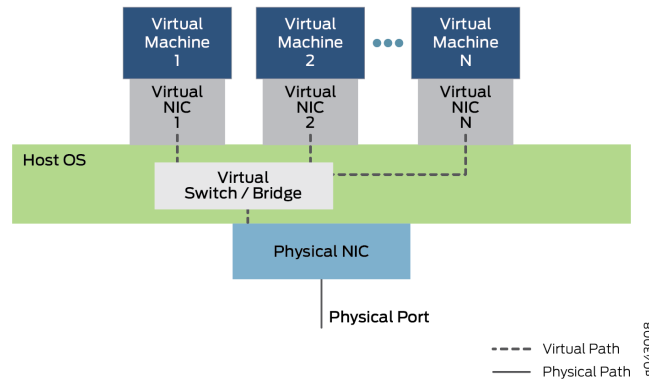


Figure 5 on page 9 shows traffic entering and leaving the device through the internal bridge. Virtual Machine 1 links its virtualized internal NIC interface to the physical external NIC interface. Virtual Machine 2 and Virtual Machine N link internal virtual NICs through the internal bridge. Note that 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 5](#)
- [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).

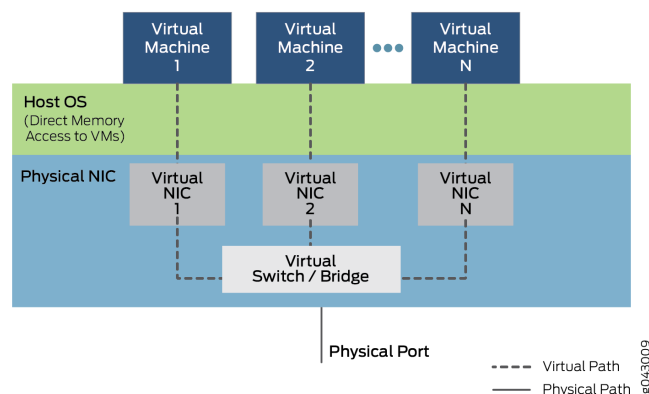
When a physical device is virtualized, both physical network interface card (NIC) interfaces and external physical switches as well as the virtual NIC interfaces and internal virtual switches coexist. So when the isolated virtual machines (VMs) or containers in the device, each with their own memory 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 that 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 10](#).

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 5](#)
- [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 5](#)
- [Understanding Virtio Usage on page 8](#)
- [Understanding SR-IOV Usage on page 10](#)

Understanding Physical and Virtual Components

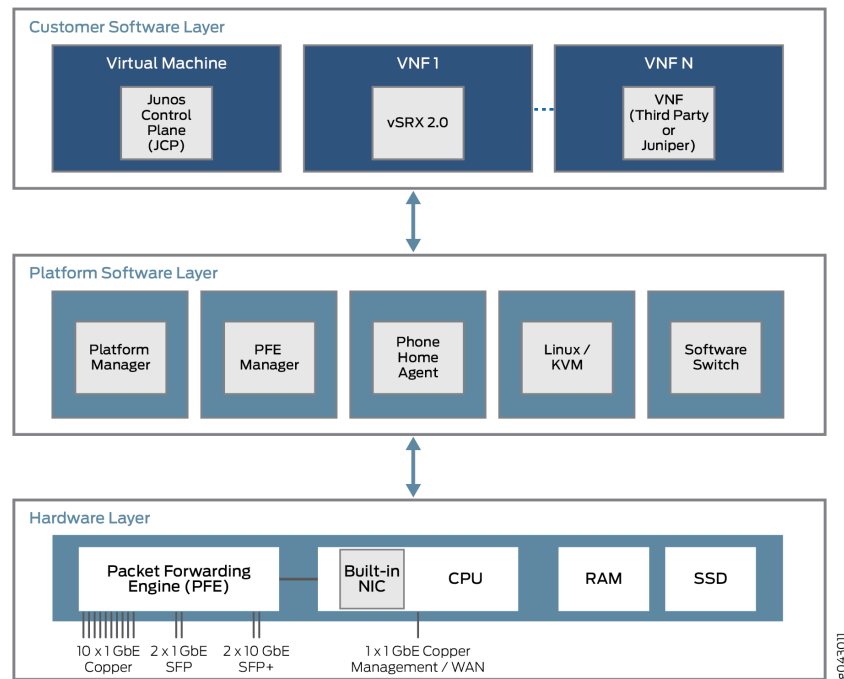
In the disaggregated Junos OS Network Functions Virtualization (NFV) environment, device components might be physical or virtual. The same physical-virtual distinction can be applied to interfaces (ports), the paths that packets or frames take through the device, and other aspects such as CPU cores or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

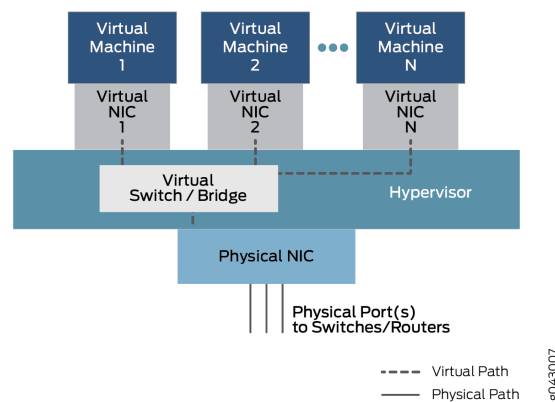
Above the platform software layer is the customer software layer, which performs various platform-independent functions. Some of the components might be Juniper Networks virtual machines, such as a virtual SRX device (vSRX) or the Junos Control Plane (JCP). The JCP works with the JDM to make the device 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 5](#)
 - [Understanding Virtio Usage on page 8](#)
 - [Understanding SR-IOV Usage on page 10](#)
 - [Comparing virtio and SR-IOV on page 11](#)

PART 2

Installation

- [Installation on page 17](#)

CHAPTER 2

Installation

- [Managing Software Installation on NFX250 Network Services Platform on page 17](#)
- [Upgrading an Image on the Disaggregated Junos OS Platform on page 18](#)
- [Reverting the System to the Factory-Default Configuration on page 21](#)
- [Rebooting the System on page 21](#)

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 26](#)
- [Accessing the JDM Shell, JDM CLI, and JCP Prompts in a Disaggregated Junos OS Platform on page 26](#)

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.LKb5WwriFu8junos_cli_upg/jinstall-nfx-2-junos-15.1X53-D45.3.secure-linux.tgz
...
upgrade_platform: Input package
/var/tmp/tmp.LKb5WwriFu8junos_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@porter-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 tcstd: 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 21](#)
 - [Rebooting the System on page 21](#)

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 18](#)
 - [Rebooting the System on page 21](#)

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 18](#)
 - [Reverting the System to the Factory-Default Configuration on page 21](#)

PART 3

Management

- [Management on page 25](#)
- [Management Commands on page 43](#)

CHAPTER 3

Management

- [Understanding the JDM CLI on page 26](#)
- [Accessing the JDM Shell, JDM CLI, and JCP Prompts in a Disaggregated Junos OS Platform on page 26](#)
- [Understanding User Accounts on page 28](#)
- [Configuring JDM User Accounts and Authentication on page 29](#)
- [Understanding JDM Management Interfaces on page 30](#)
- [Configuring the Out-of-Band Management Interface for JDM on page 31](#)
- [Configuring the In-Band Management Interface for JDM on page 33](#)
- [Configuring the Out-of-Band Management Interface for Hypervisor on page 34](#)
- [Configuring SSH Service and NETCONF-Over-SSH Connections for Remote Access to the Disaggregated Junos OS Platform on page 35](#)
- [Configuring HTTP Access to the Disaggregated Junos OS Platform on page 36](#)
- [Configuring HTTPS Access to the Disaggregated Junos OS Platform on page 36](#)
- [Configuring SNMP on JDM on page 36](#)
- [Configuring Enhanced Orchestration in the Disaggregated Junos OS Platform on page 38](#)
- [Configuring IPSec in the Disaggregated Junos OS Platform on page 39](#)
- [Viewing and Managing Centralized Log Files in a Disaggregated Junos OS Platform on page 39](#)
- [Managing Core Files for a Disaggregated Junos OS Platform on page 40](#)

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 added-value 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 the operational mode and configuration mode. You use the **configuration** command to get 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 26](#)
- [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 27](#)
- [Accessing the JDM Shell on page 27](#)
- [Accessing the JCP Prompt from the JDM CLI on page 27](#)

- [Accessing the Hypervisor from the JDM CLI on page 28](#)
- [Accessing the ipsec-nm from the JDM CLI on page 28](#)

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

By 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_dagrawal built 2015-08-24 05:13:01 UTC

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



NOTE: In the factory-default state, the default root password for vjunos0 is **juniper123**.

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 26](#)
- [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 JunosVM user accounts.

- [Root Account on page 28](#)
- [Other User Accounts on page 29](#)
- [User Authentication on page 29](#)

Root Account

In the factory-default configuration, 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 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 JDM. To do this, you must use the JDM CLI to do so. You cannot use the JDM shell to create user accounts.

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

User Authentication

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 29](#)
 - [Understanding the JDM CLI on page 26](#)

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 regress authentication load-key-file
URL-to-ssh-key-file
```

- To configure TACACS+ 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 28](#)
- [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 30](#)
- [Out-of-Band Management Interface on page 30](#)
- [In-Band Management Interface on page 31](#)

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 26.

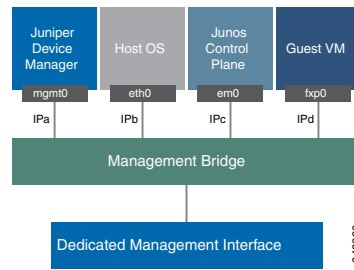
Out-of-Band Management Interface

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 31](#).

Figure 9: Out-of-band Management Interface

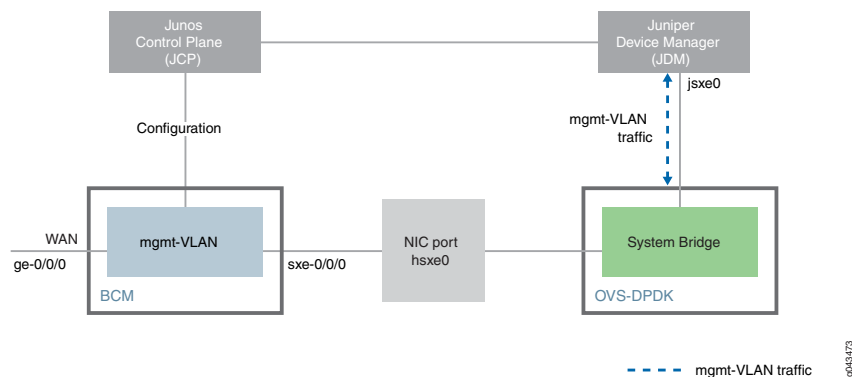


In-Band Management Interface

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 31 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 JDM In-Band Management Interface on page 33](#)
- [Understanding JDM Management Interfaces on page 30](#)

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 32](#)
- [Configuring the Out-of-Band Management Interface with IPv6 Addressing for JDM on page 32](#)

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

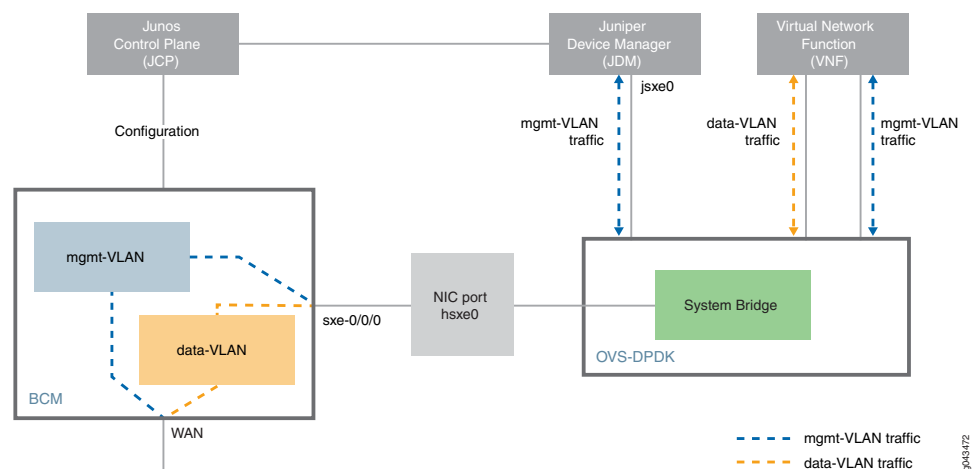
- [Configuring the JDM In-Band Management Interface on page 33](#)

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 31](#).

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 JDM Out-of-Band Management Interface on page 31](#)

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 35](#)
- [Configuring the Out-of-Band Management Interface with IPv6 Addressing for Hypervisor on page 35](#)

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
```

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 NETCONF-over-SSH connections. Regular SSH connections to the port are ignored.

Related Documentation

- [Understanding the JDM CLI on page 26](#)
- [Accessing the JDM Shell, JDM CLI, and JCP Prompts in a Disaggregated Junos OS Platform on page 26](#)

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 36](#)
- [Understanding the JDM CLI on page 26](#)
- [Accessing the JDM Shell, JDM CLI, and JCP Prompts in a Disaggregated Junos OS Platform on page 26](#)

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 36](#)
- [Understanding the JDM CLI on page 26](#)
- [Accessing the JDM Shell, JDM CLI, and JCP Prompts in a Disaggregated Junos OS Platform on page 26](#)

Configuring SNMP on JDM

There are several SNMP-enabled components in NFX (JDM, yypervisor, 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 37](#)
- [Configuring SNMP System Parameters on page 37](#)
- [Configuring SNMP v3 on page 37](#)
- [Configuring SNMP Traps on page 38](#)
- [Querying SNMP MIBs on page 38](#)
- [Managing Traps on page 38](#)

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 39](#)
- [Viewing Log Messages on page 40](#)

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** 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 40](#)

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 on page 40](#)

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 39](#)

CHAPTER 4

Management Commands

- [enhanced-orchestration on page 44](#)
- [http on page 44](#)
- [https on page 44](#)
- [ipsec-nm on page 45](#)
- [netconf on page 45](#)
- [outbound-ssh on page 46](#)
- [phone-home on page 47](#)
- [rest on page 48](#)
- [ssh on page 48](#)
- [system on page 49](#)
- [traceoptions on page 51](#)
- [upgrade-image-before-configuration on page 51](#)
- [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](#)
- [show system services ipsec-nm](#)
- [show system visibility cpu](#)
- [show system visibility host](#)
- [show system visibility jcp](#)
- [show system visibility jdm](#)
- [show system visibility memory](#)
- [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	<code>ipsec-nm;</code>
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	<pre>netconf { ssh { port <i>port-number</i>; } traceoptions { flag [all incoming outgoing]; file { <i>file-name</i>; size <i>file-size</i>; } } }</pre>
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	<p>ssh —Allow NETCONF connection over SSH.</p> <p><i>port-number</i>—Identifier of the service port.</p> <p>traceoptions —Options that are available for the NETCONF trace operation.</p> <p>flag —Parameters that can be specified for the NETCONF trace operation.</p> <p><i>file-name</i>—Name of file in which the trace information is available.</p> <p><i>file-size</i>—Maximum size of a trace file.</p>
Required Privilege Level	system—To view this statement in the configuration.

outbound-ssh

Syntax	<pre> outbound-ssh { client <i>client id</i> { <i>address</i>; device-id <i>device-id</i>; disable-ssh-security-settings; keep-alive; reconnect-strategy; secret; services; } traceoptions { flag [all configuration connectivity]; file { <i>file-name</i>; size <i>file-size</i>; } } } </pre>
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	<p><i>client- id</i>—Identifier of a client application that initiates the SSH connection.</p> <p><i>address</i>—Address of the client to which the connection must be established.</p> <p><i>device-id</i>—Unique ID used by the client to identify a device.</p> <p><i>traceoptions</i>—Options that are available for the outbound SSH trace operation.</p> <p><i>flag</i>—Parameters that can be specified for the outbound SSH trace operation.</p> <p><i>file-name</i>—Name of file in which the trace information is available.</p> <p><i>file-size</i>— Maximum size of a trace file.</p>
Required Privilege Level	system—To view this statement in the configuration.

phone-home

Syntax	<pre> 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; } } } </pre>
Hierarchy Level	[edit system]
Release Information	Statement introduced in Junos OS Release 15.1X53-D45 for the NFX250 Network Services Platform.
Description	<p>An option that is used for initial boot up and configuration of the device when the client device is switched on.</p> <p>The default configuration for phone-home is as follows:</p> <pre> user@jdm# set system phone-home server https://redirect.juniper.net user@jdm# set system phone-home upgrade-image-before-configuration </pre>
Options	<p>server—Name of the server.</p> <p>upgrade-image-before-configuration—Upgrades the image before applying the configuration received from the Network Activator.</p> <p>traceoptions—Options that are available for the phone-home trace operation</p> <p>no-remote-trace—</p> <p>file-name—Name of file in which the trace information is available</p> <p>file-size—Maximum size of a trace file.</p> <p>flag—Parameters that can be specified for the phone-home trace operation.</p>
Required Privilege Level	system—To view this statement in the configuration.

rest

Syntax	<pre>rest { control; http; traceoptions; }</pre>
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	<p>control—Control of the REST API process.</p> <p>traceoptions—Options that are available for the REST API trace operation.</p> <p>http—HTTP connection settings that are not encrypted.</p>
Required Privilege Level	system—To view this statement in the configuration.

ssh

Syntax	<pre>ssh;</pre>
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;
Hierarchy Level	<pre> system 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 <i>client id</i> { address; device-id <i>device-id</i>; 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 <i>port-number</i>; } traceoptions { flag [all incoming outgoing]; file { file-name; size file-size; } } } </pre>

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

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	<pre> traceoptions { no-remote-trace; file { file-name; size file-size; } flag { all; config; function; misc; socket; state-machine; } </pre>
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	<p>traceoptions—Options that are available for the phone-home trace operations.</p> <p>no-remote-trace—Trace operations for the phone-home is not supported.</p> <p>file-name—Name of file in which the trace information is available.</p> <p>file-size—Maximum size of a trace file.</p> <p>flag—Parameters that can be specified for the phone-home trace operation.</p>
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 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	<ul style="list-style-type: none"> • show system inventory hardware memory on page 55 • show system inventory hardware network on page 57 • show system inventory hardware storage on page 59
List of Sample Output	show system inventory hardware cpu on page 53
Output Fields	Table 3 on page 52 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 3: 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.
No of Max vCPUs	The maximum number of allowed virtual CPUs.
Fields for CPU Statistics	

Table 3: show system inventory hardware cpu Output Fields (*continued*)

Field Name	Field Description
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	<ul style="list-style-type: none"> • show system inventory hardware cpu on page 52 • show system inventory hardware network on page 57 • show system inventory hardware storage on page 59
List of Sample Output	show system inventory hardware memory on page 55
Output Fields	Table 4 on page 55 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 4: 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).
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
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	<ul style="list-style-type: none"> • show system inventory hardware cpu on page 52 • show system inventory hardware memory on page 55 • show system inventory hardware storage on page 59
List of Sample Output	show system inventory hardware network on page 57
Output Fields	Table 5 on page 57 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 5: 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	<ul style="list-style-type: none"> • show system inventory hardware cpu on page 52 • show system inventory hardware memory on page 55 • show system inventory hardware network on page 57
List of Sample Output	show system inventory hardware storage on page 60
Output Fields	Table 6 on page 59 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 6: 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	
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.

Table 6: show system inventory hardware storage Output Fields (*continued*)

Field Name	Field Description
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	<ul style="list-style-type: none"> • show system visibility vnf on page 87
List of Sample Output	show system inventory software vnf on page 62
Output Fields	Table 7 on page 62 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 7: 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	<ul style="list-style-type: none"> • show system visibility host on page 67 • show system visibility jcp on page 72 • show system visibility jdm on page 75 • show system visibility memory on page 79 • show system visibility network on page 81 • show system visibility storage on page 84 • show system visibility vnf on page 87
List of Sample Output	show system services ipsec-nm on page 63
Output Fields	Table 8 on page 63 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 8: 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	<ul style="list-style-type: none"> • show system visibility host on page 67 • show system visibility jcp on page 72 • show system visibility jdm on page 75 • show system visibility memory on page 79 • show system visibility network on page 81 • show system visibility storage on page 84 • show system visibility vnf on page 87
List of Sample Output	show system visibility cpu on page 65
Output Fields	Table 8 on page 63 lists the output fields for the show system visibility cpu command. Output fields are listed in the approximate order in which they appear.

Table 9: 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.
Service Time	The amount of service time, in seconds.

Table 9: show system visibility cpu Output Fields (*continued*)

Field Name	Field Description
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	<ul style="list-style-type: none"> • show system visibility cpu on page 64 • show system visibility jcp on page 72 • show system visibility jdm on page 75 • show system visibility memory on page 79 • show system visibility network on page 81 • show system visibility storage on page 84 • show system visibility vnf on page 87
List of Sample Output	show system visibility host on page 69
Output Fields	Table 10 on page 67 lists the output fields for the show system visibility host command. Output fields are listed in the approximate order in which they appear.

Table 10: 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.
Zombie	The total number of zombie processes.
Fields for Host CPU Information	

Table 10: show system visibility host Output Fields (*continued*)

Field Name	Field Description
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.
Fields for List of Host Interfaces	
Interfaces	The name of the interface.

Table 10: show system visibility host Output Fields (*continued*)

Field Name	Field Description
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_vjunos	/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
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	<ul style="list-style-type: none"> • show system visibility cpu on page 64 • show system visibility host on page 67 • show system visibility jdm on page 75 • show system visibility memory on page 79 • show system visibility network on page 81 • show system visibility storage on page 84 • show system visibility vnf on page 87
List of Sample Output	show system visibility jcp on page 73
Output Fields	Table 11 on page 72 lists the output fields for the show system visibility jcp command. Output fields are listed in the approximate order in which they appear.

Table 11: 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	
Interface	The name of the interface.
Address	The IP address of the interface.

Table 11: show system visibility jcp Output Fields (*continued*)

Field Name	Field Description
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	<ul style="list-style-type: none"> • show system visibility cpu on page 64 • show system visibility host on page 67 • show system visibility jcp on page 72 • show system visibility memory on page 79 • show system visibility network on page 81 • show system visibility storage on page 84 • show system visibility vnf on page 87
List of Sample Output	show system visibility jdm on page 77
Output Fields	Table 12 on page 75 lists the output fields for the show system visibility jdm command. Output fields are listed in the approximate order in which they appear.

Table 12: 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.
Fields for JDM Disk Usage Information	
Total	The total amount of disk usage space, in mebibytes (MiB).

Table 12: show system visibility jdm Output Fields (*continued*)

Field Name	Field Description
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—Virtual Memory	
Total	The total amount of virtual memory, in mebibytes (MiB).
Used	The amount of used virtual memory, in mebibytes (MiB).
Free	The amount of free virtual memory, in mebibytes (MiB).
Percentage Used	The percentage of used virtual memory.
Fields for JDM Memory Information—Swap Memory	
Total	The total amount of swap memory, in mebibytes (MiB).
Used	The amount of used swap memory, in mebibytes (MiB).
Free	The amount of free swap memory, in mebibytes (MiB).
Percentage Used	The percentage of used swap memory.
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.

Table 12: show system visibility jdm Output Fields (*continued*)

Field Name	Field Description
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.
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
-----

```

Virtual Memory:

Total (KiB): 15949116
Used (KiB): 5533316
Free (KiB): 10415800
Percent Used: 31.3

Swap Memory:

Total (KiB): 1249996
Used (KiB): 0
Free (KiB): 1249996
Percent Used: 0.0

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	<ul style="list-style-type: none"> • show system visibility cpu on page 64 • show system visibility host on page 67 • show system visibility jcp on page 72 • show system visibility jdm on page 75 • show system visibility network on page 81 • show system visibility storage on page 84 • show system visibility vnf on page 87
List of Sample Output	show system visibility memory on page 80
Output Fields	Table 13 on page 79 lists the output fields for the show system visibility memory command. Output fields are listed in the approximate order in which they appear.

Table 13: 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).
Free	The total amount of free virtual memory, in kibibytes (KiBs).
Percent Used	The percentage of buffer virtual memory used.
Fields for Memory Information—Swap Memory	
Total	The total amount of available swap memory, in kibibytes (KiBs).
Used	The total amount of used swap memory, in kibibytes (KiBs).
Free	The total amount of free swap memory, in kibibytes (KiBs).
Percent Used	The percentage of buffer swap memory used.

Sample Output

show system visibility memory

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

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

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	<ul style="list-style-type: none"> • show system visibility cpu on page 64 • show system visibility host on page 67 • show system visibility jcp on page 72 • show system visibility jdm on page 75 • show system visibility memory on page 79 • show system visibility storage on page 84 • show system visibility vnf on page 87
List of Sample Output	show system visibility network on page 82
Output Fields	Table 14 on page 81 lists the output fields for the show system visibility network command. Output fields are listed in the approximate order in which they appear.

Table 14: 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.
PF	The names of the Physical Functions available.

Table 14: show system visibility network Output Fields (*continued*)

Field Name	Field Description
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	<ul style="list-style-type: none"> • show system visibility cpu on page 64 • show system visibility host on page 67 • show system visibility jcp on page 72 • show system visibility jdm on page 75 • show system visibility memory on page 79 • show system visibility network on page 81 • show system visibility vnf on page 87 • show system visibility cpu on page 64 • show system visibility host on page 67 • show system visibility jcp on page 72 • show system visibility jdm on page 75 • show system visibility memory on page 79 • show system visibility network on page 81 • show system visibility vnf on page 87
List of Sample Output	show system visibility storage on page 85
Output Fields	Table 15 on page 84 lists the output fields for the show system visibility storage command. Output fields are listed in the approximate order in which they appear.

Table 15: show system visibility storage Output Fields

Field Name	Field Description
Fields for Disk Usage Information	
Disk	The name of the disk.
Total	The amount of total disk space, in mebibytes (MiB).
Used	The amount of used disk space, in mebibytes (MiB).

Table 15: show system visibility storage Output Fields (*continued*)

Field Name	Field Description
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 % Used	Total (MiB)	Used (MiB)	Free (MiB)
/dev/sda4 64.0	1409	904	416
/dev/sda5 5.0	5639	320	5009
/dev/sda2 72.0	976	705	271
/dev/sda1 26.0	189	50	139
/dev/sda3 1.0	1	0	1
/dev/mapper/vg0_vjunos-1v_junos 39.0	9951	3889	5534

```
/dev/mapper/vg0_vjunos-lv_var_third_party 159317 6888 144313
4.0
```

Disk I/O Information

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

VNF Disk Information

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

show system visibility vnf

Syntax	<code>show system visibility vnf <i>vnf name</i></code>
Release Information	Command introduced in Junos OS Release 15.1X53-D40 for the NFX250 Network Services Platform.
Description	<p>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.</p> <p>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.</p>
Required Privilege Level	view
Related Documentation	<ul style="list-style-type: none"> • show system visibility cpu on page 64 • show system visibility host on page 67 • show system visibility jcp on page 72 • show system visibility jdm on page 75 • show system visibility memory on page 79 • show system visibility network on page 81 • show system visibility storage on page 84
List of Sample Output	show system visibility vnf on page 89
Output Fields	Table 16 on page 87 lists the output fields for the show system visibility vnf command. Output fields are listed in the approximate order in which they appear.

Table 16: 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.
Fields for VNF Memory Usage	
Name	Name of the VNF.

Table 16: show system visibility vnf Output Fields (*continued*)

Field Name	Field Description
Maximum Memory	The maximum amount of memory, in kibibytes (KiBs).
Used Memory	The total amount of used memory, in kibibytes (KiBs).
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.
Model	The connectivity model.
MAC	The MAC address of the VNF.
Fields for List of VNF Disk Information	
VNF	The name of the VNF.

Table 16: show system visibility vnf Output Fields (*continued*)

Field Name	Field Description
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)
vnf1	104857	104857

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	0	vnet5	0	0	0	252654	0
vnf1	0	vnet6	0	0	0	8893085	0

PART 4

Virtual Network Functions

- [Virtual Network Functions on page 95](#)
- [Virtual Network Functions Commands on page 105](#)

CHAPTER 5

Virtual Network Functions

- [Understanding Virtual Network Functions on page 95](#)
- [Managing the VNF Life Cycle on page 96](#)

Understanding Virtual Network Functions

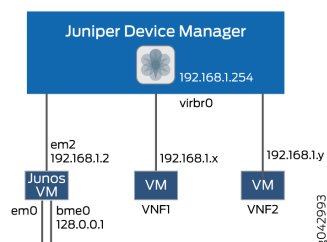
Virtualized network functions (VNFs) include all virtual entities that can be launched and managed from the Juniper Device Manager (JDM). Currently, virtual machines (VMs) are the only VNF type that is supported.

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.

JDM architecture provides an internal network that connects all VMs to JDM as shown in [Figure 12 on page 95](#).

Figure 12: Network Connections Between JDM and the VMs



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 17 on page 96](#) provides a glossary of commonly used VNF acronyms and terms.

Table 17: 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 95](#)
- [Managing the VNF Lifecycle on page 96](#)
- [Disaggregated Junos OS VMs on page 5](#)

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 on page 97](#)
- [Managing the VNF Image on page 98](#)
- [Preparing the Bootstrap Configuration on page 98](#)
- [Launching a VNF on page 98](#)
- [Allocating Resources for a VNF on page 99](#)
- [Managing VNF States on page 101](#)
- [Managing VNF MAC Addresses on page 102](#)

- [Accessing a VNF from JDM on page 102](#)
- [Viewing List of VNFs on page 102](#)
- [Displaying the VNF Details on page 102](#)
- [Deleting a VNF on page 103](#)

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 18 on page 97](#) provides the list of physical CPUs that are reserved for NFX250-LS1.

Table 18: Physical CPU Allocation for NFX250-LS1

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

[Table 19 on page 97](#) provides the list of physical CPUs that are reserved for NFX250.

Table 19: 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.

Allocating Resources for a VNF

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

- [Specifying CPU for VNF on page 99](#)
- [Allocating Memory for a VNF on page 99](#)
- [Configuring VNF Storage Devices on page 100](#)
- [Configuring VNF Interfaces and VLANs on page 100](#)

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.

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.

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.

4. 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.

5. 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, and any user-assigned MAC address in the VNF init-descriptor XML file is overwritten.

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
-	vsrc	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.

Displaying the VNF Details

To display VNF details:

```
user@jdm> show virtual-network-functions vnf-name
```

Virtual Machine Information

```
-----
Name:          vsrx
IP Address:    192.168.1.4
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.

CHAPTER 6

Virtual Network Functions Commands

- [features on page 106](#)
- [hugepages on page 106](#)
- [image on page 107](#)
- [init-descriptor on page 108](#)
- [interfaces \(JDM\) on page 109](#)
- [ipsec-nm on page 110](#)
- [mapping on page 111](#)
- [memory on page 112](#)
- [no-autostart on page 112](#)
- [pci-address on page 113](#)
- [size on page 113](#)
- [storage on page 114](#)
- [type on page 115](#)
- [virtual-cpu on page 116](#)
- [virtual-network-functions on page 117](#)
- [vjunos0 on page 120](#)
- [vnf-name on page 121](#)
- [show virtual-network-functions](#)
- [show vlans](#)

features

Syntax	<pre>features { hugepages; }</pre>
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	<ul style="list-style-type: none">• Understanding Virtual Network Functions on page 95• Managing the VNF Lifecycle on page 96• show virtual-network-functions on page 123

hugepages

Syntax	<pre>hugepages;</pre>
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	<ul style="list-style-type: none">• Understanding Virtual Network Functions on page 95• Managing the VNF Lifecycle on page 96• show virtual-network-functions on page 123

image

Syntax	<pre>image { file-path; bus-type [ide virtio]; image-type [qcow2 raw]; }</pre>
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	<p>file-path—Path of the image source file.</p> <p>image-type—Format of the image. Default value is qcow2.</p> <p>bus-type—Type of the system bus. Default value is virtio.</p>
Required Privilege Level	<p>routing—To view this statement in the configuration.</p> <p>routing-control—To add this statement to the configuration.</p>
Related Documentation	<ul style="list-style-type: none"> • Understanding Virtual Network Functions on page 95 • Managing the VNF Lifecycle on page 96 • show virtual-network-functions on page 123

init-descriptor

Syntax	<code>init-descriptor <i>file-path</i>;</code>
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	<i>file-path</i> —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	<ul style="list-style-type: none">• Understanding Virtual Network Functions on page 95• Managing the VNF Lifecycle on page 96• show virtual-network-functions on page 123

interfaces (JDM)

Syntax	<pre> interfaces <i>interface-name</i> { <i>pci-address</i> <i>pci-address</i>; mapping { hsxe0 { virtual-function { vlan-id <i>vlan-id</i>; } } hsxe1 { virtual-function { vlan-id <i>vlan-id</i>; } } vlan { members <i>vlan-name</i>; mode [access trunk]; native-vlan-id <i>vlan-id</i>; } } } </pre>
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	<p><i>interface-name</i>—Name of the VNF interfaces.</p> <p><i>pci-address</i>—PCI address for the VNF interfaces.</p> <p><i>vlan-id</i>—SR-IOV virtual function to use to attach a VNF to a physical interface.</p> <p><i>vlan members</i>—Membership for this interface.</p> <p><i>native-vlan-id</i>—Virtual LAN identifier for untagged frames. For example, 1...4095</p> <p><i>vlan-name</i>—Name of the VLAN members.</p>
Required Privilege Level	<p>interface—To view this statement in the configuration.</p> <p>interface-control—To add this statement to the configuration.</p>
Related Documentation	<ul style="list-style-type: none"> • Understanding Virtual Network Functions on page 95 • Managing the VNF Lifecycle on page 96 • show virtual-network-functions on page 123

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 95](#)
- [Managing the VNF Lifecycle on page 96](#)
- [show virtual-network-functions on page 123](#)

mapping

Syntax	<pre> mapping { hsxe0 { virtual-function { vlan-id <i>vlan-id</i>; } } hsxe1 { virtual-function { vlan-id <i>vlan-id</i>; } } vlan { members <i>vlan-name</i>; mode [access trunk]; native-vlan-id <i>vlan-id</i>; } } </pre>
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	<p><i>vlan-id</i>—SR-IOV virtual function to use to attach a VNF to a physical interface.</p> <p><i>vlan members</i>—Membership for this interface.</p> <p><i>native-vlan-id</i>—Virtual LAN identifier for untagged frames. For example, 1...4095</p> <p><i>vlan-name</i>—Name of the VLAN members.</p>
Required Privilege Level	<p>interface—To view this statement in the configuration.</p> <p>interface-control—To add this statement to the configuration.</p>
Related Documentation	<ul style="list-style-type: none"> • Understanding Virtual Network Functions on page 95 • Managing the VNF Lifecycle on page 96 • show virtual-network-functions on page 123

memory

Syntax	<pre>memory { size size; features { hugepages; } }</pre>
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	<ul style="list-style-type: none">• Understanding Virtual Network Functions on page 95• Managing the VNF Lifecycle on page 96• show virtual-network-functions on page 123

no-autostart

Syntax	<pre>no-autostart;</pre>
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	<ul style="list-style-type: none">• Understanding Virtual Network Functions on page 95• Managing the VNF Lifecycle on page 96• show virtual-network-functions on page 123

pci-address

Syntax	<code>pci-address <i>pci-address</i>;</code>
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	<ul style="list-style-type: none"> • Understanding Virtual Network Functions on page 95 • Managing the VNF Lifecycle on page 96 • show virtual-network-functions on page 123

size

Syntax	<code>size <i>size</i>;</code>
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	<ul style="list-style-type: none"> • Understanding Virtual Network Functions on page 95 • Managing the VNF Lifecycle on page 96 • show virtual-network-functions on page 123

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 95](#)
- [Managing the VNF Lifecycle on page 96](#)
- [show virtual-network-functions on page 123](#)

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	<ul style="list-style-type: none">• Understanding Virtual Network Functions on page 95• Managing the VNF Lifecycle on page 96• show virtual-network-functions on page 123

virtual-cpu

Syntax	<pre>virtual-cpu { virtual-cpu-number { physical-cpu <i>number</i> <i>range</i>; } count <i>number</i>; features { hardware-virtualization; } }</pre>
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	<p>virtual-cpu count <i>number</i>—Number of virtual CPUs. For example, 2.</p> <p>virtual-cpu features—Features of the virtual cpu. For example, hardware-virtualization.</p>
Required Privilege Level	<p>routing—To view this statement in the configuration.</p> <p>routing-control—To add this statement to the configuration.</p>
Related Documentation	<ul style="list-style-type: none">• Understanding Virtual Network Functions on page 95• Managing the VNF Lifecycle on page 96• show virtual-network-functions on page 123

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;`
 `}`
 `}`
 `}`
 `}`
 `}`
 `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 (JDM) 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;
        }
    }
}
}
}

```

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.

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.

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 95](#)
 - [Managing the VNF Lifecycle on page 96](#)
 - [show virtual-network-functions on page 123](#)

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 95](#)
 - [Managing the VNF Lifecycle on page 96](#)
 - [show virtual-network-functions on page 123](#)

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 (JDM) 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	<p>interfaces—Name of the interface. For example, em1.</p> <p>no-autostart—An option to disable auto-start of VNF on the VNF configuration commit.</p> <p>pci-address—An option to specify PCI address for the VNF interfaces.</p> <p>mapping—An option to map VNF interfaces on platforms running disaggregated Junos OS.</p> <p>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.</p>
Required Privilege Level	<p>routing—To view this statement in the configuration.</p> <p>routing-control—To add this statement to the configuration.</p>
Related Documentation	<ul style="list-style-type: none"> • Understanding Virtual Network Functions on page 95 • Managing the VNF Lifecycle on page 96 • show virtual-network-functions on page 123

show virtual-network-functions

Syntax	<pre>show virtual-network-functions show virtual-network-functions <i>vnf-name</i> show virtual-network-functions ipsec-nm show virtual-network-functions vjunos0</pre>
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	<p><i>vnf-name</i>—(Optional) Display information for a specific VNF.</p> <p><i>ipsec-nm</i>—(Optional) Display information of the system VNF IPsec.</p> <p><i>vjunos0</i>—(Optional) Display information of the system VNF vjunos0.</p>
Required Privilege Level	view
Related Documentation	<ul style="list-style-type: none"> • Understanding Virtual Network Functions on page 95 • Managing the VNF Lifecycle on page 96
List of Sample Output	show virtual-network-functions vjunos0 on page 124 show virtual-network-functions vsrx on page 124 show virtual-network-functions on page 124
Output Fields	Table 20 on page 123 describes the output fields for the show virtual-network-functions command. Output fields are listed in the approximate order in which they appear.

Table 20: show virtual-network functions Output Fields

Field Name	Field Description
Name	Name of the VNF
IP Address	IP address of the VNF
Status	Status of the VNF. Possible values are Running, Shutdown, or Undefined.
Liveliness	Indicates whether or not the IP address of the VNF is reachable.
VCPUs	Number of virtual CPUs
Maximum Memory	Maximum amount of memory available to the VNF
Used Memory	Amount of memory used by the VNF
Target	Source

Table 20: show virtual-network functions Output Fields (*continued*)

Field Name	Field Description
<i>device-name</i>	List of storage devices that are configured for VNF and its source file.

Sample Output

show virtual-network-functions vjunos0

```

user@host> show virtual-network-functions vjunos0
Virtual Machine Information
-----
Name:                vjunos0
IP Address:          192.168.1.2
Status:              Running
Liveliness:          Up
VCPUs:               1
Maximum Memory:      2000896
Used Memory:         2000896

```

Sample Output

show virtual-network-functions vsrx

```

user@host> show virtual-network-functions vsrx
Virtual Machine Information
-----
Name:                vsrx
IP Address:          192.168.1.4
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

```

Sample Output

show virtual-network-functions

```

user@host> show virtual-network-functions
ID      Name                State      Liveliness
-----
2       vjunos0             running    alive
4       vsrx1               running    alive
8931    jdm                 running    alive

```

show vlans

Syntax	<code>show vlans <i>vlan-name</i></code>
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	<p><i>vlan-name</i>—Display information for a specific VLAN.</p> <p>brief detail extensive terse—(Optional) Display the specified level of output.</p>
Required Privilege Level	view
List of Sample Output	show vlans on page 125
Output Fields	Table 20 on page 123 describes the output fields for the show virtual-network-functions command. Output fields are listed in the approximate order in which they appear.

Table 21: 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

- [Service Chaining on page 129](#)

CHAPTER 7

Service Chaining

- [Understanding Service Chaining on Disaggregated Junos OS Platforms on page 129](#)
- [Configuring Service Chaining Using VLANs on page 130](#)
- [Configuring Service Chaining Using DHCP Services on VLANs on page 131](#)
- [Example: Configuring Service Chaining Using VLANs on NFX250 Network Services Platform on page 131](#)
- [Example: Configuring Service Chaining Using SR-IOV on NFX250 Network Services Platform on page 136](#)

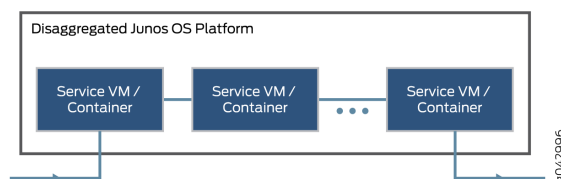
Understanding Service Chaining on Disaggregated Junos OS Platforms

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 93.

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 5](#)
 - [Understanding Virtio Usage on page 8](#)
 - [Understanding SR-IOV Usage on page 10](#)
 - [Understanding Virtual Network Functions on page 95](#)

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 ethx mapping vlan mode [access|trunk]`
`set virtual-network-functions vnf-name interfaces ethx mapping vlan members list`
3. Attach a native VLAN ID to the VNF interface:
`set virtual-network-functions vnf-name interfaces ethx mapping vlan native-vlan-id vlan-id`

- Related Documentation**
- [Understanding Service Chaining on Disaggregated Junos OS Platforms on page 129](#)
 - [Example: Configuring Service Chaining Using VLANs on NFX250 Network Services Platform on page 131](#)

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 ethx mapping vlan mode [access|trunk]
set virtual-network-functions vnf-name interfaces ethx mapping vlan members list
```

5. Enable the DHCP client on the VNF.

To check the assigned IP address, use the *show system visibility vnf <vnf>* command.

Related Documentation

- [Understanding Service Chaining on Disaggregated Junos OS Platforms on page 129](#)
- [Example: Configuring Service Chaining Using VLANs on NFX250 Network Services Platform on page 131](#)

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 on page 132](#)
- [Overview on page 132](#)
- [Configuration on page 133](#)

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.

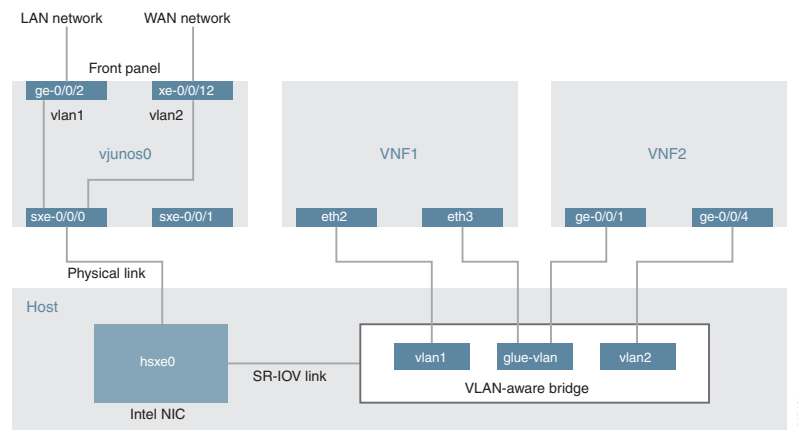
Overview

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 132](#).

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 on page 133](#)
- [Configuring the VNF Interfaces and Creating the Service Chain on page 135](#)

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 interface-mode trunk
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 Vlan2
```

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# user@jcp# set interfaces sxe-0/0/0.0 family ethernet-switching interface-mode trunk
user@jcp# 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 Vlan77;
    }
  }
}
```

```
[edit]
user@jcp# show interfaces xe-0/0/12
unit 0 {
  family ethernet-switching {
    interface-mode trunk;
    vlan {
      members Vlan1177;
    }
  }
}
```

```
[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
Vlan1177 {
  vlan-id 1177;
}
Vlan77 {
  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 vlans 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 VLAN1, 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 132](#).

Related Documentation

- [Understanding Service Chaining on Disaggregated Junos OS Platforms on page 129](#)
- [Disaggregated Junos OS VMs on page 5](#)
- [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 on page 136](#)
- [Overview on page 136](#)
- [Configuration on page 138](#)

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

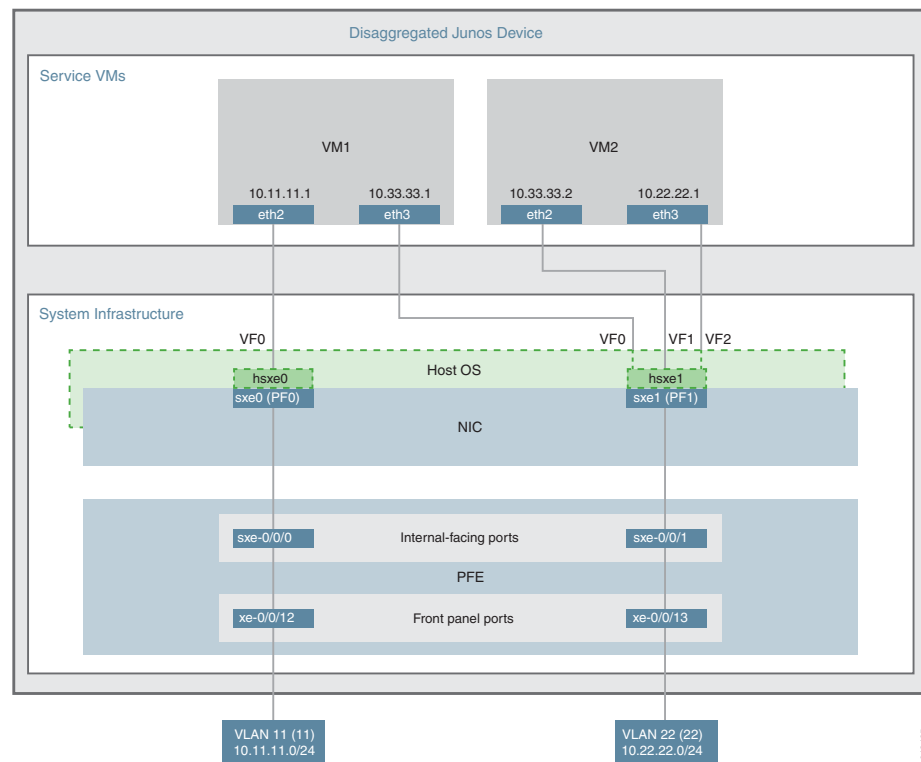
Overview

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 137](#).

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 138](#)
- [Creating the Service Chain on page 140](#)

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 sx-0/0/0.0 family ethernet-switching interface-mode trunk
set interfaces sx-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 sx-0/0/1.0 family ethernet-switching interface-mode trunk
set interfaces sx-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 137](#).

```
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 129](#)
- [Disaggregated Junos OS VMs on page 5](#)
- [Understanding SR-IOV Usage on page 10](#)

PART 6

Index

- [Index on page 145](#)

Index

Symbols

#, comments in configuration statements.....	xiv
(), in syntax descriptions.....	xiv
< >, in syntax descriptions.....	xiv
[], in configuration statements.....	xiv
{ }, in configuration statements.....	xiv
(pipe), in syntax descriptions.....	xiv

B

braces, in configuration statements.....	xiv
brackets	
angle, in syntax descriptions.....	xiv
square, in configuration statements.....	xiv

C

comments, in configuration statements.....	xiv
containers	
disaggregated Junos.....	5
conventions	
text and syntax.....	xiii
curly braces, in configuration statements.....	xiv
customer support.....	xv
contacting JTAC.....	xv

D

disaggregated Junos	
containers.....	5
overview.....	3
overview of virtual components.....	12
SR-IOV and virtio compared.....	11
SR-IOV usage.....	10
understanding.....	3
understanding virtual components.....	12
virtio usage.....	8
VMs.....	5
documentation	
comments on.....	xv

F

font conventions.....	xiii
-----------------------	------

M

manuals	
comments on.....	xv

P

parentheses, in syntax descriptions.....	xiv
--	-----

S

SR-IOV	
compared to virtio.....	11
used in disaggregated Junos.....	10
support, technical See technical support	
syntax conventions.....	xiii

T

technical support	
contacting JTAC.....	xv

V

virtio	
compared to SR-IOV.....	11
used in disaggregated Junos.....	8
VMs	
disaggregated Junos.....	5

