



Cisco Nexus 9000v (9300v/9500v) Guide, Release 10.5(x)

First Published: 2024-07-27

Americas Headquarters

Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
<http://www.cisco.com>
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS REFERENCED IN THIS DOCUMENTATION ARE SUBJECT TO CHANGE WITHOUT NOTICE. EXCEPT AS MAY OTHERWISE BE AGREED BY CISCO IN WRITING, ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS DOCUMENTATION ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED.

The Cisco End User License Agreement and any supplemental license terms govern your use of any Cisco software, including this product documentation, and are located at: <http://www.cisco.com/go/softwareterms>. Cisco product warranty information is available at <http://www.cisco.com/go/warranty>. US Federal Communications Commission Notices are found here <http://www.cisco.com/c/en/us/products/us-fcc-notice.html>.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any products and features described herein as in development or available at a future date remain in varying stages of development and will be offered on a when-and if-available basis. Any such product or feature roadmaps are subject to change at the sole discretion of Cisco and Cisco will have no liability for delay in the delivery or failure to deliver any products or feature roadmap items that may be set forth in this document.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

The documentation set for this product strives to use bias-free language. For the purposes of this documentation set, bias-free is defined as language that does not imply discrimination based on age, disability, gender, racial identity, ethnic identity, sexual orientation, socioeconomic status, and intersectionality. Exceptions may be present in the documentation due to language that is hardcoded in the user interfaces of the product software, language used based on RFP documentation, or language that is used by a referenced third-party product.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: [www.cisco.com go trademarks](http://www.cisco.com/go/trademarks). Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

© 2024 Cisco Systems, Inc. All rights reserved.



CONTENTS

Trademarks ?

PREFACE

Preface	vii
Audience	vii
Document Conventions	vii
Related Documentation for Cisco Nexus 9000 Series Switches	viii
Documentation Feedback	viii
Communications, Services, and Additional Information	viii
Cisco Bug Search Tool	ix
Documentation Feedback	ix

CHAPTER 1

New and Changed Information	1
New and Changed Information	1

CHAPTER 2

Overview	3
About Cisco Nexus 9000v Platform Family	3
Minimum Memory Requirement on Host	3
Cisco Nexus 9300v Platform	4
Cisco Nexus 9300v Platform Components	5
Cisco Nexus 9300v Chassis	6
Cisco Nexus 9300v Line Card	7
vNIC Mapping	7
Cisco Nexus 9500v Platform	8
Cisco Nexus 9500v Platform Components	10
Cisco Nexus 9500v Chassis	10
Cisco Nexus 9500v Line Cards	11

- vNIC Mapping 12
- Nexus 9000v Throughput 14
- Nexus 9000v Feature Support 15
 - Layer 2 Features 15
 - Layer 3 Features 16
 - VXLAN and Segment Routing Features 17
 - Programmability Features 18
 - Guestshell Support 18
- Nexus 9000v Platform MIB Support 18
- Nexus 9000v Platform Guidelines and Limitations 19

CHAPTER 3

- Nexus 9000v Deployment 21**
 - Nexus 9000v Hypervisor Support 21
 - KVM/QEMU Attributes 21
 - ESXi Attributes 22
 - VirtualBox Attributes 22
 - Nexus 9000v Deployment Workflow for KVM/QEMU 22
 - Common Deployment Workflow 23
 - Platform Specific Workflow 25
 - Interconnecting Platforms 25
 - Nexus 9000v Deployment Workflow for ESXi 27
 - Common Deployment Workflow 27
 - Platform Specific Workflow 29
 - Interconnecting Platforms 29
 - Nexus 9000v Deployment Workflow for Vagrant 29
 - Common Deployment Workflow 30
 - Platform Specific Workflow 30
 - Support for Sync Folder in Vagrant 30
 - Interconnecting Platforms 33
 - Image Upgrade Workflow 33
 - Deploying from a New Artifact 34
 - Upgrading from a New NX-OS Image 34

CHAPTER 4

- Nexus 9300v and 9500v Lite NX-OS Image 35**

About Nexus 9300v and 9500v Lite NX-OS Image	35
Resource Requirements	35
Virtual Artifacts	36
Features Supported	36
Nexus 9300v Lite and 9500v Lite NX-OS Image Deployment	38

APPENDIX A

Troubleshooting the Cisco Nexus 9000v	39
Troubleshooting the Cisco Nexus 9000v Platform	39
Common Issues for All Hypervisors	39
ESXi Hypervisor Issues	40
KVM/QEMU Hypervisor Issues	42
Vagrant/VirtualBox Issues	43
Troubleshooting the Cisco Nexus 9000v Dataplane	44
Debug Commands	44
Event History Commands	45
Show Commands	45
Packet Capture Commands	48



Preface

This preface includes the following sections:

- [Audience, on page vii](#)
- [Document Conventions, on page vii](#)
- [Related Documentation for Cisco Nexus 9000 Series Switches, on page viii](#)
- [Documentation Feedback, on page viii](#)
- [Communications, Services, and Additional Information, on page viii](#)

Audience

This publication is for network administrators who install, configure, and maintain Cisco Nexus switches.

Document Conventions

Command descriptions use the following conventions:

Convention	Description
bold	Bold text indicates the commands and keywords that you enter literally as shown.
<i>Italic</i>	Italic text indicates arguments for which you supply the values.
[x]	Square brackets enclose an optional element (keyword or argument).
[x y]	Square brackets enclosing keywords or arguments that are separated by a vertical bar indicate an optional choice.
{x y}	Braces enclosing keywords or arguments that are separated by a vertical bar indicate a required choice.
[x {y z}]	Nested set of square brackets or braces indicate optional or required choices within optional or required elements. Braces and a vertical bar within square brackets indicate a required choice within an optional element.

Convention	Description
<i>variable</i>	Indicates a variable for which you supply values, in context where italics cannot be used.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string includes the quotation marks.

Examples use the following conventions:

Convention	Description
<code>screen font</code>	Terminal sessions and information the switch displays are in screen font.
boldface screen font	Information that you must enter is in boldface screen font.
<i>italic screen font</i>	Arguments for which you supply values are in italic screen font.
<>	Nonprinting characters, such as passwords, are in angle brackets.
[]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

Related Documentation for Cisco Nexus 9000 Series Switches

The entire Cisco Nexus 9000 Series switch documentation set is available at the following URL:

http://www.cisco.com/en/US/products/ps13386/tsd_products_support_series_home.html

Documentation Feedback

To provide technical feedback on this document, or to report an error or omission, please send your comments to nexus9k-docfeedback@cisco.com. We appreciate your feedback.

Communications, Services, and Additional Information

- To receive timely, relevant information from Cisco, sign up at [Cisco Profile Manager](#).
- To get the business impact you're looking for with the technologies that matter, visit [Cisco Services](#).
- To submit a service request, visit [Cisco Support](#).
- To discover and browse secure, validated enterprise-class apps, products, solutions, and services, visit [Cisco DevNet](#).
- To obtain general networking, training, and certification titles, visit [Cisco Press](#).
- To find warranty information for a specific product or product family, access [Cisco Warranty Finder](#).

Cisco Bug Search Tool

[Cisco Bug Search Tool](#) (BST) is a gateway to the Cisco bug-tracking system, which maintains a comprehensive list of defects and vulnerabilities in Cisco products and software. The BST provides you with detailed defect information about your products and software.

Documentation Feedback

To provide feedback about Cisco technical documentation, use the feedback form available in the right pane of every online document.



CHAPTER 1

New and Changed Information

- [New and Changed Information](#) , on page 1

New and Changed Information

The table provides a list of the features introduced/ modified for Release 10.5(x).

Feature	Description	Changed in Release	Where Documented
Reduced footprint N9Kv Lite image to 6G	Reduced the footprint of Nexus 9300v and 9500v Lite NX-OS image to 6G	10.5(1)F	Nexus 9300v and 9500v Lite NX-OS Image, on page 35



CHAPTER 2

Overview

This chapter contains the following sections:

- [About Cisco Nexus 9000v Platform Family, on page 3](#)
- [Cisco Nexus 9300v Platform, on page 4](#)
- [Cisco Nexus 9500v Platform, on page 8](#)
- [Nexus 9000v Throughput, on page 14](#)
- [Nexus 9000v Feature Support, on page 15](#)
- [Nexus 9000v Platform MIB Support, on page 18](#)
- [Nexus 9000v Platform Guidelines and Limitations, on page 19](#)

About Cisco Nexus 9000v Platform Family

The Cisco Nexus 9000v is a virtual platform family that is designed to simulate control plane aspects of a standalone switch running Cisco Nexus 9000 software. This platform family uses the same software image that runs the Cisco Nexus 9000 hardware platforms. Although the virtual platforms don't attempt to simulate any specific ASICs or hardware SKUs, they are aligned with their hardware counterparts. An optimized Cisco software data plane handles the traffic across the line card interfaces. The Cisco Nexus 9000v virtual platform family consists of two virtual platforms: Nexus 9300v and Nexus 9500v. The following sections describe the capabilities of these two platforms.

The virtual platforms in the Nexus 9000v platform family allows you to simulate their network in a cost-effective manner. Use the simulated network to validate configurations prior to their application on a production network. Use these platforms to rapidly develop and test network automation tools using Cisco NX-OS programmability interfaces.

Minimum Memory Requirement on Host

Nexus 9000v requires a minimum of 10GB of RAM to boot up. Ensure that the underlying host (or laptop) has additional memory available.

Cisco Nexus 9300v Platform

The Cisco Nexus 9300v platform simulates a single supervisor non-modular chassis with a single co-located line card. This virtual chassis closely aligns with the standalone Cisco Nexus 9300 hardware platform running in the 'lxc' mode. The following tables show the specifications for this virtual platform:

Form-Factor

Component/Parameter	Specification
Usage	Simulation
Binary	Same as NX-OS Hardware
Management Interface	1
Line Cards	1
Line Card Interfaces	64

Resource Requirements

Resource	Specification
Minimum RAM	10.0 G (basic bootup)
Recommended RAM	12.0 G (depending on the number of features)
Minimum vCPUs	4
Recommended vCPUs	4
Minimum vNICs	1
Maximum vNICs	65

Deployment Environment

- KVM/QEMU 4.2.0
- ESXI 8.0
- Vagrant 2.3.7

To deploy a Nexus 9300v platform, fetch the appropriate virtual artifacts from Cisco CCO. The following table documents the supported virtual artifacts. After deploying the virtual machine, it reports itself as a Nexus 9300v platform.



Note The Cisco Nexus 9300v platform supported only the 32-bit image until Cisco NX-OS Release 10.1(x). Beginning from Cisco NX-OS Release 10.2(1), it supports only 64-bit image.

The following table displays the virtual artifact(s):

Hypervisor	Virtual Artifact	Description
ESXI 8.0	nexus9300v.10.1.1.ova	Contains virtual disk, machine definition, and NXOS image.
KVM/QEMU 4.2.0	nexus9300v.10.1.1.qcow2	Contains virtual disk and NXOS image on bootflash.
Vagrant 2.3.7	nexus9300v.10.1.1.box	Contains a preinstalled NXOS image on a virtual disk along with a machine definition.

After the initial virtual machine deployment, you can upgrade the Cisco NX-OS image on the platform using the typical NX-OS workflow (example: **install all <>**).



Note When you upgrade an older Nexus 9000v to the current release, it's automatically transformed into the Nexus 9300v. Even after subsequent reloads and NX-OS image upgrades, the platform will continue to present itself as a Nexus 9300v.

Cisco Nexus 9300v Platform Components

The Cisco Nexus 9300v platform, like its reference hardware counterpart, consists of three key components: chassis, supervisor, and one line card. The following table presents the product identifications (PIDs) and the SNMP sysOID used associated with the platform

Component	Description	PID	sysOID
Chassis	Nexus9000 C9300v Chassis	N9K-C9300v	Note To ensure backward compatibility, the sysOID used in the previous release Nexus 9000v platform is reused for the Nexus 9300v platform.
Supervisor	Supervisor Module	N9K-vSUP	
Line Card	Nexus 9000v 64-port Ethernet Module	N9K-X9364v	

Cisco Nexus 9300v Chassis

The following references sample chassis-related information outputs from relevant **show** commands.

```
switch# sh version
Cisco Nexus Operating System (NX-OS) Software
TAC support: http://www.cisco.com/tac
Documents: http://www.cisco.com/en/US/products/ps9372/tsd_products_support_series_home.html
Copyright (c) 2002-2022, Cisco Systems, Inc. All rights reserved.
The copyrights to certain works contained herein are owned by
other third parties and are used and distributed under license.
Some parts of this software are covered under the GNU Public
License. A copy of the license is available at
http://www.gnu.org/licenses/gpl.html.
```

Nexus 9000v is a demo version of the Nexus Operating System

```
Software
BIOS: version
NXOS: version 10.2(3) [build 10.2(2.185)] [Feature Release]
BIOS compile time:
NXOS image file is: bootflash:///nxos64-cs.10.2.2.185.F.bin
NXOS compile time: 3/30/2022 13:00:00 [03/31/2022 00:30:59]
```

```
Hardware
cisco Nexus9000 C9300v Chassis
Intel(R) Xeon(R) CPU E5-2658 v4 @ 2.30GHz with 20499656 kB of memory.
Processor Board ID 9GFDLI2JD0R
Device name: switch
bootflash: 4287040 kB
```

Kernel uptime is 1 day(s), 23 hour(s), 35 minute(s), 21 second(s)

```
Last reset
Reason: Unknown
System version:
Service:
```

```
plugin
Core Plugin, Ethernet Plugin
```

Active Package(s):

```
switch#
switch# sh module
Mod Ports Module-Type Model Status
-----
1 64 Nexus 9000v 64 port Ethernet Module N9K-X9364v ok
27 0 Virtual Supervisor Module N9K-vSUP active *
```

```
Mod Sw Hw Slot
-----
1 10.2(2.185) 0.0 LC1
27 NA 0.0 SUP1
```

```
Mod MAC-Address(es) Serial-Num
-----
1 00-ed-c6-a2-01-01 to 00-ed-c6-a2-01-40 9JAUkW2T51G
27 00-ed-c6-a2-1b-01 to 00-ed-c6-a2-1b-12 9GFDLI2JD0R
```

```
Mod Online Diag Status
-----
1 Pass
```



```

27 Pass

* this terminal session
switch#
switch# sh inventory
NAME: "Chassis", DESCR: "Nexus9000 C9300v Chassis"
PID: N9K-C9300v , VID: , SN: 9ZQKP299FIZ

NAME: "Slot 1", DESCR: "Nexus 9000v 64 port Ethernet Module"
PID: N9K-X9364v , VID: , SN: 9JAUkW2T51G

NAME: "Slot 27", DESCR: "Supervisor Module"
PID: N9K-vSUP , VID: , SN: 9GFDLI2JD0R

switch#

```

Cisco Nexus 9300v Line Card

Cisco Nexus 9300v platform supports a single virtual line card with 64 virtual interfaces. The line card automatically populates when the platform boots. You can't insert or remove the line card from this chassis. The line card boot process starts after the Supervisor successfully boots and reaches the "active" state. Like its hardware counterpart, the line card boot-up starts with the "present" state and becomes fully functional when it reaches the "ok" state.

vNIC Mapping

On an actual Cisco Nexus 9300 hardware platform, you can "plug in" fibers to the front panel ports on a line card. On a virtual platform, like the Nexus 9300v, you must export the required number of virtual network interface cards/interfaces (vNICs) from the hypervisor into the Nexus 9300v platform.

The Nexus 9300v platform uses a sequential vNIC mapping. It maps the first vNIC passed in by the hypervisor into the Nexus 9300v management port. Subsequent vNICs are mapped sequentially into the line card interfaces. For example, if you export two vNICs onto the Nexus 9300v, the first vNIC is mapped to the NX-OS "mgmt" interface. The second vNIC is mapped to the "Ethernet1/1" interface.

vNIC Mapping Informational Show Commands

Show Platform vNIC Commands

On the Cisco Nexus 9300v platform, CLI commands are available to show the current vNIC mapping scheme, the number of vNICs mapped, and the mapping of MAC addresses to vNICs. Using these commands, you can ensure that the correct number of vNICs were passed to their virtual machine, and you can see which interfaces have been mapped.

Example outputs of the show vNIC platform commands:

show platform vnic mapped

```

v-switch# show platform vnic mapped

```

NXOS Interface	VNIC MAC-Address	Internal VNIC
Ethernet1/1	00c0.c000.0101	phyEth1-1
Ethernet1/2	00c0.c000.0102	phyEth1-2
Ethernet1/3	00c0.c000.0103	phyEth1-3
Ethernet1/4	00c0.c000.0104	phyEth1-4
Ethernet1/5	00c0.c000.0105	phyEth1-5
Ethernet1/6	00c0.c000.0106	phyEth1-6
Ethernet1/7	00c0.c000.0107	phyEth1-7
Ethernet1/8	00c0.c000.0108	phyEth1-8

```

Ethernet1/9      00c0.c000.0109    phyEth1-9
Ethernet1/10     00c0.c000.010a    phyEth1-10
Ethernet1/11     00c0.c000.010b    phyEth1-11
Ethernet1/12     00c0.c000.010c    phyEth1-12
Ethernet1/13     00c0.c000.010d    phyEth1-13
Ethernet1/14     00c0.c000.010e    phyEth1-14
Ethernet1/15     00c0.c000.010f    phyEth1-15
Ethernet1/16     00c0.c000.0110    phyEth1-16

```

show platform vnic info

```

v-switch# show platform vnic info
  VNIC Scheme: Sequential
  mgmt0 interface: eth1 (00c0.c000.aabb)
  Module      # VNICs Mapped
  -----
  16          16
  -----
  VNICs passed: 16
  VNICs mapped: 16
  VNICs unmapped: 0

```

Cisco Nexus 9500v Platform

The Cisco Nexus 9500v simulates a single-supervisor platform 16 slot modular chassis that supports dynamic line card insertion and removal. This virtual chassis closely aligns with the standalone Cisco Nexus 9500 hardware platform. This version of Nexus 9500v currently doesn't simulate the system controller or fabric card typically found on the modular hardware chassis. This platform supports four different form factors of the generic line cards. These line cards share the same Linux kernel and differ only in the supported number of interfaces. The following tables show the specifications for this virtual platform.

Form-Factor

Component/Parameter	Specification
Usage	Simulation
Binary	Same as NX-OS Hardware
Management Interface	1
Line Cards	Up to 16
Line Card Interfaces	Up to 400 interfaces in the KVM/QEMU environment

Resource Requirements

Resource	Specification
Minimum RAM	10.0G (basic bootup with one line card; 1.2G for each additional line card)
Recommended RAM	12.0G (depending on the number of features)

Resource	Specification
Minimum vCPUs	4 (if you configure 16 line cards, we recommend 6 vCPUs)
Minimum vNICs	1
Maximum vNICs	400 interfaces in the KVM/QEMU environment

Deployment Environment

- KVM/QEMU 4.2.0
- ESXI 8.0

To deploy a Cisco Nexus 9500v platform, fetch the appropriate virtual artifacts from Cisco CCO. The following table documents the supported virtual artifacts. Once you deploy the virtual machine, it reports itself as a Nexus 9500v.



Note The Cisco Nexus 9500v platform supports only the 64-bit image of the Cisco NX-OS Release 10.1(1).

The table displays the virtual artifact(s):

Hypervisor	Virtual Artifact	Description
ESXI 8.0	nexus9500v64.10.1.1.ova	Contains virtual disk, machine definition, and NXOS image The 64-bit .ova file boots the N9500v platform, which in turn boots up the 64-bit image of Cisco NX-OS Release 10.1(1) Software. Note The Supervisor is 64-bit, and the line card is 32-bit.
KVM/QEMU 4.2.0	nexus9500v64.10.1.1.qcow2	Contains virtual disk and NXOS image on bootflash.

After the initial virtual machine deployment, you can upgrade the Cisco NX-OS image on the platform using the typical NX-OS workflow (example: **install all <>**).



Note You can't transform a Cisco Nexus 9000v from a previous release into a Nexus 9500v platform. You can change it by applying the Nexus 9500v virtual artifact.

Cisco Nexus 9500v Platform Components

The Cisco Nexus 9500v platform, like its reference hardware counterpart, consists of three key components: chassis, supervisor, and line cards. The following table presents the product identifications (PIDs) and the SNMP sysOID used associated with the platform

Component	Description	PID	sysOID
Chassis	Nexus9000 C9500v Chassis	N9K-C9500v	EntPhysicalVendorType = cevChassisN9KC9500v
Supervisor	Supervisor Module	N9K-vSUP	
Line Card	Nexus 9000v 64-port Ethernet Module	N9K-X9564v	

Cisco Nexus 9500v Chassis

The following references sample chassis-related information outputs from relevant **show** commands, for a Nexus 9500v platform with a single line card.

```
switch# sh version
Cisco Nexus Operating System (NX-OS) Software
TAC support: http://www.cisco.com/tac
Documents: http://www.cisco.com/en/US/products/ps9372/tsd_products_support_series_home.html
Copyright (c) 2002-2022, Cisco Systems, Inc. All rights reserved.
The copyrights to certain works contained herein are owned by
other third parties and are used and distributed under license.
Some parts of this software are covered under the GNU Public
License. A copy of the license is available at
http://www.gnu.org/licenses/gpl.html.

Nexus 9000v is a demo version of the Nexus Operating System

Software
  BIOS: version
  NXOS: version 10.2(3) [build 10.2(2.191)] [Feature Release]
  BIOS compile time:
  NXOS image file is: bootflash:///nxos64-cs.10.2.2.191.F.bin
  NXOS compile time: 4/5/2022 11:00:00 [04/05/2022 22:45:26]

Hardware
  cisco Nexus9000 C9500v Chassis ("Supervisor Module")
  Intel(R) Xeon(R) CPU E5-2658 v4 @ 2.30GHz with 16395468 kB of memory.

...skipping 1 line
Device name: switch
bootflash: 4287040 kB

Kernel uptime is 0 day(s), 0 hour(s), 7 minute(s), 51 second(s)

Last reset
Reason: Unknown
System version:
Service:

plugin
Core Plugin, Ethernet Plugin
```

```

Active Package(s):

switch#
switch# sh module
Mod Ports Module-Type Model Status
-----
1 64 Nexus 9000v 64 port Ethernet Module N9K-X9564v ok
27 0 Virtual Supervisor Module N9K-vSUP active *

Mod Sw Hw Slot
-----
1 10.2(2.191) 0.0 LC1
27 10.2(2.191) 0.0 SUP1

Mod MAC-Address(es) Serial-Num
-----
1 00-66-1e-aa-01-01 to 00-66-1e-aa-01-40 9LKOT30P6A5
27 00-66-1e-aa-1b-01 to 00-66-1e-aa-1b-12 9V3ZW14YP0M

Mod Online Diag Status
-----
1 Pass
27 Pass

* this terminal session
switch# sh inventory
NAME: "Chassis", DESCR: "Nexus9000 C9500v Chassis"
PID: N9K-C9500v , VID: , SN: 9NNVBPJ9N8A

NAME: "Slot 1", DESCR: "Nexus 9000v 64 port Ethernet Module"
PID: N9K-X9564v , VID: , SN: 9LKOT30P6A5

NAME: "Slot 27", DESCR: "Supervisor Module"
PID: N9K-vSUP , VID: , SN: 9V3ZW14YP0M

switch#

```

Cisco Nexus 9500v Line Cards

The Cisco Nexus 9500v platform can support up to 16 virtual line cards. The platform supports five different forms of line cards. The differences between these virtual line cards are the number of NX-OS interfaces they support. By default, the Nexus 9500v platform boots with a single line card.

You can insert or remove virtual line cards using a CLI command. When using the sequential [Sequential vNIC Mapping Scheme](#), insert the line cards sequentially from module 1. The removal operation must be in the opposite order. However, when using [MAC-Encoded vNIC Mapping Scheme](#), you can insert line cards in any order. This mode supports sparse population of the line card.

The line card boot process starts after the Supervisor successfully boots and reaches the "active" state. Like their hardware counterparts, line card boot up starts with the "present" state and becomes fully functional when it reaches the "ok" state.

To support line insertion, the **platform insert module *number* linecard** command is used. A line card can be removed by prefixing the command with **no**. Once the chassis is configured with line cards, the chassis configuration remains persistent across switch reboots.

```

switch# platform insert ?
  module  Insert a specific module

switch# platform insert module?
  <1-30>  Please enter the module number

```

```

switch# platform insert module 2?
<CR>
linecard Linecard module

switch# platform insert module 2 linecard ?
N9K-X9516v Nexus 9000v 16 port Ethernet Module
N9K-X9532v Nexus 9000v 32 port Ethernet Module
N9K-X9536v Nexus 9000v 36 port Ethernet Module
N9K-X9548v Nexus 9000v 48 port Ethernet Module
N9K-X9564v Nexus 9000v 64 port Ethernet Module

```

vNIC Mapping

On actual Cisco Nexus 9500 hardware platforms, you can "plug in" fibers to the front panel ports on a line card. On a virtual platform, like the Nexus 9500v, export the required number of virtual network interface cards/interfaces (vNICs) from the hypervisor into the Nexus 9500v platform.

The Nexus 9500v platform supports two vNIC mapping schemes. Depending on the specified scheme, the system maps the vNIC passed in by the hypervisor into the appropriate Nexus 9500v NX-OS interface. The following sections describe the capabilities of both the vNIC mapping schemes.

Sequential vNIC Mapping Scheme

In this scheme, vNICs acquired from the hypervisor are mapped into the NX-OS interfaces sequentially. For example, if you export two vNICs to the Nexus 9500v, the first vNIC is mapped to the NX-OS "mgmt" interface. The second vNIC is mapped to the "Ethernet1/1" interface. This is the default mapping mode when the virtual switch boots up for the first time.

Other limitations are:

- This mode doesn't support sparse population of line cards or interfaces.
- The line cards must be inserted in sequence from module 1 and removed in the opposite order.

When a line card is removed, the vNICs remain within the system and are automatically remapped into the appropriate NX-OS interface upon reinsertion of the line card.

Use the **platform vnic scheme** command to select the vNIC mapping scheme. Once you select a mapping scheme, it remains persistent through a switch reload. By default, the Nexus 9500v switch boots up in the sequential vNIC mapping scheme. Changing this scheme requires a switch reload.

```

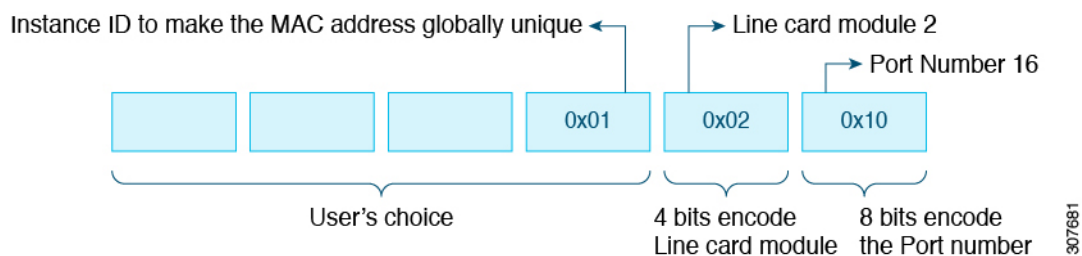
switch# platform vnic ?
scheme Virtual Network Interface Card allocation scheme

switch# platform vnic scheme ?
mac-encoded MAC address encoded allocation of vNICs to linecard modules
sequential Sequential allocation of vNICs to linecard modules

```

MAC-Encoded vNIC Mapping Scheme

In this scheme, vNICs acquired from the hypervisor are mapped to NX-OS interfaces based on the MAC address configured on the vNIC (at the hypervisor level). This mode allows you to map any vNIC to any NX-OS line card interface. To use this mode, add the line card module and port number into the last 2 bytes of the vNIC MAC address. This MAC address configuration must be performed at the hypervisor level before powering up the Nexus 9500v virtual switch. The following diagram presents the required vNIC MAC address format:



Other features:

- This mode supports sparse population of line cards and interfaces.
- While the Nexus 9500v is in this mode, you can insert and remove the line cards in any order.
- When a line card is removed, the vNICs remain within the system and are automatically remapped into the appropriate NX-OS interface upon reinsertion of the line card.

Use the **platform vnic scheme** command to select the vNIC mapping scheme. Once you select a mapping scheme, it remains persistent through the switch reload. By default, the Nexus 9500v switch boots up in the sequential vNIC mapping scheme. Changing this scheme requires a switch reload.

```
switch# platform vnic ?
  scheme  Virtual Network Interface Card allocation scheme

switch# platform vnic scheme ?
  mac-encoded  MAC address encoded allocation of vNICs to linecard modules
  sequential   Sequential allocation of vNICs to linecard modules
```



Note This scheme allows line card modules to be inserted and removed in any order. However, once the line cards have been inserted, the mapping scheme must not be changed to sequential unless absolutely necessary. If the mapping scheme is changed to "sequential", you must remove all line cards in non-sequential order and insert them back starting with module 1. If you insert line cards in non-sequential order, change the scheme to sequential, and reboot the switch, none of the line cards will come online. The switch, booted in the sequential scheme, expects any existing line cards to be present in serial order, starting with module 1. An error message similar to the following displays the mismatch in the vNIC scheme and the line cards inserted:

```
2020 Jul 15 14:44:03 N9Kv_3 %$ VDC-1 %$ %PLATFORM-2-MOD_INSERTION_FAILED:
Failed to insert module 6 (Nexus 9000v 64 port Ethernet Module - vNIC allocation scheme
is set to sequential, modules must be inserted in sequence)
```

You can validate the scheme by entering the **show platform vnic info** command. To recover from the above state, change the vNIC scheme back to MAC-encoded by entering the **platform vnic scheme mac-encoded** command and reboot the switch. If you require the vNIC scheme to be sequential, remove all line cards first before changing the scheme to sequential.

vNIC Mapping Informational Show Commands

Show Platform vNIC Commands

On the Cisco Nexus Nexus 9500v platform, CLI commands are available to show the current vNIC mapping scheme, the number of vNICs mapped, and the mapping of MAC addresses to vNICs. Using these commands, you can ensure that the correct number of vNICs were passed to their virtual machine, and you can see which interfaces have been mapped.

Example outputs of the show vNIC platform commands:

show platform vnic mapped

```
v-switch# show platform vnic mapped
  NXOS Interface      VNIC MAC-Address      Internal VNIC
  -----
Ethernet1/1          00c0.c000.0101        phyEth1-1
Ethernet1/2          00c0.c000.0102        phyEth1-2
Ethernet1/3          00c0.c000.0103        phyEth1-3
Ethernet1/4          00c0.c000.0104        phyEth1-4
Ethernet1/5          00c0.c000.0105        phyEth1-5
Ethernet1/6          00c0.c000.0106        phyEth1-6
Ethernet1/7          00c0.c000.0107        phyEth1-7
Ethernet1/8          00c0.c000.0108        phyEth1-8
Ethernet1/9          00c0.c000.0109        phyEth1-9
Ethernet1/10         00c0.c000.010a        phyEth1-10
Ethernet1/11         00c0.c000.010b        phyEth1-11
Ethernet1/12         00c0.c000.010c        phyEth1-12
Ethernet1/13         00c0.c000.010d        phyEth1-13
Ethernet1/14         00c0.c000.010e        phyEth1-14
Ethernet1/15         00c0.c000.010f        phyEth1-15
Ethernet1/16         00c0.c000.0110        phyEth1-16
```

show platform vnic info

```
v-switch# show platform vnic info
  VNIC Scheme: Mac-Encoded
  mgmt0 interface: eth1 (00c0.c000.aabb)
  Module      # VNICS Mapped
  -----
  1           16
  -----
  VNICs passed: 32
  VNICs mapped: 16
  VNICs unmapped: 16
```

Nexus 9000v Throughput

This section describes approximate throughput values for the Nexus 9000v virtual platform and the resource requirements to achieve the stated values.

4vCPU/16GB

The following table identifies the approximate throughput values for the Nexus 9000v with four virtual CPUs and 16 GB of RAM.

Feature(s)	Throughput on Stateless Traffic	Throughput on Stateful Traffic	
	UDP Protocol Traffic by Third-party Packet-generator-tool	TCP Protocol Traffic by file transfer	TCP Protocol Traffic by iPerf-tool
L2 switching, unicast forwarding	~700 Mbps	~50 Mbps	500 Mbps

Feature(s)	Throughput on Stateless Traffic	Throughput on Stateful Traffic	
	UDP Protocol Traffic by Third-party Packet-generator-tool	TCP Protocol Traffic by file transfer	TCP Protocol Traffic by iPerf-tool
L3 routing, unicast forwarding	~60 Mbps	~5 Mbps	50 Mbps

The following are the details of the throughput measurement:

Two Linux-VMs with iPerf-tools and one Third-party device were used for traffic generation and throughput measurement.

Topology used

- Nexus-9000v runs on a ESXI 8.0 hypervisor enabled UCS-device that were connected back-to-back to a Third-party Packet-generator-device.
- Nexus-9000v was connected back-to-back to Linux-VMs and all VMs were run on a ESXI 8.0 hypervisor enabled UCS-device.

Stateless (unidirectional) traffic testing

User defined UDP packets generated by Third-party device were sent through Nexus-9000v.

Stateful (bidirectional) traffic testing

- FTP based file (2GB sized) transfer was performed by Third-party device through Nexus-9000v.
- TCP traffic generated by iPerf-tools server/client VMs were sent through Nexus-9000v.

Nexus 9000v Feature Support

The Cisco Nexus 9000v platform family simulates a broad set of Nexus features. The forwarding plane of these features is implemented on a Cisco proprietary software data plane. Therefore, there can be some behavior differences. For example, the amount of system throughput is different between the virtual simulation platform and its hardware counterpart.

The following tables list the Cisco NX-OS features that have been tested on the Nexus 9000v platforms. You can configure and simulate untested NX-OS platform-independent features on the Nexus 9000v platforms. However, consider these features as unsupported. As more features are tested on the platform, the following table will be updated.

It's important to note that some of the chassis form-factor dependent feature commands can be available only on the corresponding Nexus 9000v platform. For example, NAT commands will be enabled only on the N9300 hardware platform and not on the N9500 hardware platform. It is also important to note that availability of a command does not imply that the feature is supported on the data plane. Please refer to the following feature tables for the supported features.

Layer 2 Features

The following table lists layer 2 feature support for the Nexus 9300v and Nexus 9500v platforms.

Feature	Nexus 9300v Support	Nexus 9500v Support
802.1AB LLDP	Yes	Yes
802.1Q VLANs/Trunk	Yes	Yes
802.1s RST	Yes	Yes
802.3ad LACP	Yes	Yes
L2 Multicast	Yes (as broadcast)	Yes (as broadcast)
vPC/MLAG	Yes	Yes
Port Channel	Yes	Yes
VLANs	Yes	Yes

Layer 3 Features

The following table lists layer 3 feature support for the Nexus 9300v and Nexus 9500v platforms.

Feature	Nexus 9300v Support	Nexus 9500v Support
OSPF	Yes	Yes
OSPFv3	Yes	Yes
BGP	Yes	Yes
MP-BGP	Yes	Yes
IS-IS	Yes (as broadcast)	Yes (as broadcast)
RIPv2	Yes	Yes
Equal Cost Multipath Routing (ECMP)	Yes	Yes
PIM-SM	Yes	Yes
HSRP	Yes	Yes
VRRP	Yes	Yes
MPLS	Yes	Yes
EIGRP	Yes	Yes
CDP	Yes	Yes
L3 SVI	Yes	Yes
Sub Interfaces	Yes	Yes

Feature	Nexus 9300v Support	Nexus 9500v Support
IPsec	Yes	Yes

VXLAN and Segment Routing Features

The following table lists VXLAN and segment routing feature support for the Nexus 9300v and Nexus 9500v platforms.

Feature	Nexus 9300v Support	Nexus 9500v Support
VXLAN flood and Learn BUM Replication (PIM/ASM, IR)	(Yes, Yes)	(Yes, Yes)
VXLAN EVPN BUM Replication (PIM/ASM, BIDIR, IR)	(Yes, No, Yes)	(Yes, No, Yes)
VXLAN EVPN Routing	Yes	Yes
VXLAN EVPN Bridging	Yes	Yes
VXLAN EVPN Anycast GW	Yes (as broadcast)	Yes (as broadcast)
VXLAN Tunnel Endpoint	Yes	Yes
VXLAN ARP Suppression	Yes	Yes
VXLAN EVPN Multi-Site BGW	Yes (with non-vPC on border gateway (BGW))	Yes (with non-vPC on border gateway (BGW))
VXLAN EVPN TRM	No	No
VXLAN IPv6 Underlay	Yes	Yes
MPLS Segment Routing (SRv4)	No	No
Downstream VNI	Yes	Yes
vPC with Fabric Peering 1	Yes	Yes
L3VNI	No	No

¹ The vPC Fabric Peering peer-link is established over the transport network (the spine layer of the fabric). DSCP is not supported on N9000v/N9300v/N9500v; hence, overloading the spine with traffic burst may result in disruptions on the vPC state of the leaf switches.

Programmability Features

The following table lists programmability feature support for the Nexus 9300v and Nexus 9500v platforms.

Feature	Nexus 9300v Support	Nexus 9500v Support
Bash shell access and scripting	Yes	Yes
RPM support	Yes	Yes
Programmatic access to system state (Python)	Yes	Yes
Guest Shell within OS	Yes	Yes
Docker within OS	Yes	Yes
NXAPI	Yes	Yes
DME	Yes	Yes
RESTCONF	Yes	Yes
NETCONF	Yes	Yes
YANG Models	Yes	Yes
Telemetry	Yes	Yes
GNMI	Yes	Yes
NxSDK	Yes	Yes

Guestshell Support

The Cisco Nexus 9000v supports Nexus Guestshell. However, this feature isn't enabled by default. You can use Guestshell functionality by following proper Nexus Guestshell documentation to enable it.



Note The Cisco Nexus 9000v virtual artifacts currently have enough bootflash size to fit two binary images. However, Guestshell requires more bootflash disk size once enabled. There may not be enough space in bootflash to boot two binary images respectively in released virtual artifacts. Plan to clear enough disk space if you need to enable Guestshell.

Nexus 9000v Platform MIB Support

The Nexus 9000v platforms support the Simple Network Management Protocol (SNMP) along with many of the Cisco NX-OS SNMP MIBs. Note that some of the managed objects may not be relevant to the simulation platform and may not be supported.

The following list shows supported platform-related MIBs:

- CISCO entity Asset MIB
- ceEXTEntityLEDTable
- ciscoEntityExtMIB
- ciscoRFMIB
- ciscoTSMIB
- ciscoEntityFRUControlMIB
- ciscoSyslogMIB

Nexus 9000v Platform Guidelines and Limitations

The following guidelines and limitations apply to the Nexus 9000v platforms:

- Hardware consistency checker isn't supported
- Nondisruptive ISSU isn't supported
- Link carrier status of NX-OS interface depends on the capability of the underlying hypervisor.

Nested VM use cases are supported. However, take care about deploying Nexus 9000v VMs in different environments, based on requirements. In a nested VM environment, performances in terms of bootup time, admin operation, and feature application, are degraded depending on the available vCPU and memory resources.

- If you want to simulate large number of nodes, or more than 10 Nexus 9000v nodes, use a Bare metal environment. VIRL2 is currently not supported.
- Bootup time takes longer when simulating large number of nodes.
- Sparse mode is currently not supported in the VIRL environment.
- Beginning with Cisco NX-OS Release 10.1(1), when multiple VMs are needed to boot on an ESXi at the same time, it is recommended to boot a cluster of 2 to 3 VMs at one time, then, after providing some time interval, the next cluster of 2 to 3 VMs should be booted. Otherwise, the VMs may become unresponsive.
- Beginning with Cisco NX-OS Release 10.2(3)F, fcoe/fcoe-npv feature set support is added on Cisco Nexus 9300v platform switches, but not on 9500v platform switches.
- Beginning with Cisco NX-OS Release 10.2(3)F, a lite image is introduced with a reduced memory footprint, resulting in a much smaller size image than the earlier images for Cisco Nexus 9300v and 9500v.
- Telemetry is not supported on N9kv lite image, so gNMI testing is not possible on N9kv lite image. However, beginning with Cisco NX-OS Release 10.4(2)F, this is supported.
- If the user upgrades ESXi version to 8.0 or later, modify the existing value (1) of Net.BlockGuestBDPU to zero (0) in the **Manage > System > Advance** setting of ESXi Host Management UI to ensure that LACP works on N9kv.



CHAPTER 3

Nexus 9000v Deployment

This chapter contains the following sections:

- [Nexus 9000v Hypervisor Support, on page 21](#)
- [Nexus 9000v Deployment Workflow for KVM/QEMU, on page 22](#)
- [Nexus 9000v Deployment Workflow for ESXi, on page 27](#)
- [Nexus 9000v Deployment Workflow for Vagrant, on page 29](#)
- [Image Upgrade Workflow, on page 33](#)

Nexus 9000v Hypervisor Support

Both platforms in the Nexus 9000v platform family are designed to run as virtual machines on the supported hypervisors. Limitations of the underlying hypervisor may restrict some of the platform capabilities. This section provides the level of support and associated limitations.

KVM/QEMU Attributes

The following table provides the supported attributes for the KVM/QEMU hypervisor.

Attribute	Support
QEMU Version	4.2.0
BIOS	OVMF version 16, https://www.kraxel.org/repos/jenkins/edk2/ This URL accesses an index page containing the latest OVMF RPM package files. An example of the file is: <code>edk2.git-ovmf-x64-0-20200515.1388.g9099dcbd61.noarch.rpm</code> Download and extract the package file with an RPM utility. The package contains a number of files. Locate <code>OVMF-pure-efi.fd</code> and use it as the BIOS file. You can rename it <code>bios.bin</code> if you want.
Linux Version	Ubuntu 20.0.4

Attribute	Support
Platform	Nexus 9300v deployment Nexus 9500v deployment
Line Cards	Nexus 9300v: 1 line card Nexus 9500v: up to 16 line cards
Line Card Interfaces	Nexus 9300v: up to 64 line card interfaces Nexus 9500v: up to 400 line cards interfaces

ESXI Attributes

The following table provides the supported attributes for the ESXI hypervisor.

Attribute	Support
Version	8.0.
Platform	Nexus 9300v deployment Nexus 9500v deployment
Line Card	Nexus 9300v: 1 line card Nexus 9500v: up to 16 line cards
Line Card Interface	Nexus 9300v: up to 9 line card interfaces Nexus 9500v: up to 9 line cards interfaces

VirtualBox Attributes

The following table provides the supported attributes for the VirtualBox hypervisor.

Attribute	Support
Version	7.0
Platform	Nexus 9300v deployment
Line Card	Nexus 9300v: 1 line card
Line Card Interface	Nexus 9300v: up to 4 line card interfaces

Nexus 9000v Deployment Workflow for KVM/QEMU

This section describes the steps required to deploy Nexus 9000v platforms on KVM/QEMU hypervisors. Three types of deployment are available:

- Common Deployment
- Platform-Specific Deployment
- Interconnecting Deployment

Common Deployment Workflow

You can deploy the Cisco Nexus 9000v platforms through the KVM/QEMU hypervisor. The following table lists the supported parameters for the Cisco Nexus 9000v deployment on KVM/QEMU.

Parameter	Example	Description
/path_to/qemu	/usr/bin/qemu-system-x86_64	Path to QEMU executable. (download the QEMU software from http://wiki.qemu.org/download for different versions.)
-nographic	-nographic	Recommended, as the Cisco Nexus 9000v platforms don't support VGA.
-bios file	-bios bios.bin	Required. Cisco Nexus 9000v platforms use EFI boot and require a compatible BIOS image to operate. We recommend using the latest OVMF BIOS file with the SATA controller for better performance in terms of disk operation. QEMU 2.6 is recommended with the SATA controller. For more information, see http://www.linux-kvm.org/page/OVMF .
-smp	-smp 4	Cisco Nexus 9000v platforms support one to four vCPUs (we recommend two to four).
-m memory	-m 10240	Memory in MB.
-serial telnet:host:port,server,nowait	-serial telnet:localhost:8888,server,nowait or -serial telnet:server_ip:8888,server,nowait	Requires at least one.

Parameter	Example	Description
<pre>-net ... -net ... or -netdev ... -device ...</pre>	<pre>-net socket,vlan=x,name=nl_s0,listen= localhost:12000 -net nic, vlan=x, model=e1000, macaddr=aaaa.bbbb.cccc -netdev socket,listen=localhost:12000,id=eth_s_f -device e1000,addr=s.f,netdev=eth_s_f, mac=aaaa.bbbb.cccc, multifunction=on,romfile= or -netdev tap,ifname=tap_s_f,script=no, downscript=no,id=eth_s_f -device e1000,addr=s.f,netdev=eth_s_f, mac=aaaa.bbbb.ccc, multifunction=on,romfile=</pre>	<p>The net/net or netdev/device pairs are for networking a virtual network interface card (vNIC).</p> <p>The <code>_s_f</code> represents the PCI slot number and function number. QEMU 2.0 or above can plug in at least 20 PCI slots and four functions, which accommodates about 80 vNICs in total. The slot range is 3-19, and the function number range is 0-3.</p> <p>The <code>mac=</code> option passes the MAC address of each vNIC MAC address to the VM interfaces. The first <code>-netdev</code> is automatically mapped to the <code>mgmt0</code> interface on the VM. The second <code>-netdev</code> is mapped to the <code>e1/1</code> interface, and so on, up to the 65th on <code>e1/64</code>. Check that the MAC addresses are unique for each network device.</p>
<code>-enable-kvm</code>	<code>-enable-kvm</code>	This flag is required for the Cisco Nexus 9000v.
<code>-drive ... -device ... (for the SATA controller)</code>	<pre>-device ahci, id=ahci0,bus=pci.0 -drive file=img.qcow2, if=none,id=drive-sata-disk0, format=qcow2 -device idebus=ahci0,drive=drive-sata-disk0</pre>	<p>Format to use for the SATA controller. We recommend using the SATA controller with QEMU 2.6.0 because this controller offers better performance than the IDE controller. However, if there's an early QEMU version that doesn't support the SATA controller, you can use the IDE controller.</p>
<code>-drive ... media=cdrom</code>	<code>-drive file=cfg.iso,media=cdrom</code>	<p>CD-ROM disk containing a switch configuration file applied after the Cisco Nexus 9000v platform comes up.</p> <ol style="list-style-type: none"> 1. Name a text file (<code>nxos_config.txt</code>). 2. Use Linux commands to make <code>cfg.iso</code>, <code>mkisofs -o cfg.iso -l --iso-level 2 nxos_config.txt</code>.

Platform Specific Workflow

The Cisco Nexus 9500v platform runs in two different modes: sequential and mac-encoded mode. The Nexus 9300v and Nexus 9500v sequential mode deployment steps are the exact same on KVM/QEMU hypervisor. The maximum interfaces for both platforms in this case are 401 interfaces (1 management or 400 data ports).

The Nexus 9500v emulates interface traffic on multiple line cards. The virtual switch uses a single VM on KVM/QEMU for up to a total number of 400 interfaces. Based on the Nexus 9500v mac-encoded schema, specify each network adapter MAC address with the encoded slot and port number when the KVM/QEMU CLI command is invoked.

Interconnecting Platforms

Interconnecting between Nexus 9000v platform instances or any other virtual platform is based on Linux bridges and taps. Prior to invoke any CLI commands, make sure that the following is available (example configuration provided).

In the configuration example below, you can create bridges and tap interfaces along with two N9Kv switches with one management and one data interface each. Management interfaces “interface mgmt0” are connected to management network with the bridge “mgmt_bridge. The data port interfaces “interface Eth1/1” from both switches are connected back to back by using the bridge “interconnect_br”.



Note The minimum QEMU version required is 4.2.0.

- Bridges (similar to vSwitch in ESXi hypervisor) are created and set to the "up" state.

Linux commands to create bridges and bring them up:

```
sudo brctl addbr mgmt_bridge
```

```
sudo brctl addbr interconnect_br
```

```
sudo ifconfig mgmt_bridge up
```

```
sudo ifconfig interconnect_br up
```

- Tap interfaces are created based on number of interfaces the Nexus 9000v is using.

Linux command to create tap interfaces:

```
sudo openvpn --mktun --dev tap_sw1_mgmt
```

```
sudo openvpn --mktun --dev tap_sw2_mgmt
```

```
sudo openvpn --mktun --dev tap_sw1_eth1_1
```

```
sudo openvpn --mktun --dev tap_sw2_eth1_1
```

- Bridges are connected to tap interfaces.

Linux commands to connect bridges to tap interfaces:

```
sudo brctl addif mgmt_bridge tap_sw1_mgmt
```

```
sudo brctl addif mgmt_bridge tap_sw2_mgmt
```

```
sudo brctl addif interconnect_br tap_sw1_eth1_1
```

```
sudo brctl addif interconnect_br tap_sw2_eth1_1
```

- All tap interfaces must be in the "up" state.

Linux commands for bringing tap interfaces up:

```
sudo ifconfig tap_sw1_mgmt up
```

```
sudo ifconfig tap_sw2_mgmt up
```

```
sudo ifconfig tap_sw1_eth1_1 up
```

```
sudo ifconfig tap_sw2_eth1_1 up
```

- Verify that all tap interfaces are connected to bridges

Linux commands to confirm that tap interfaces are connected to bridges:

```
brctl show
```

bridge name	bridge id	STP enabled	interfaces
interconnect_br	8000.1ade2e11ec42	no	tap_sw1_eth1_1 tap_sw2_eth1_1
mgmt_bridge	8000.0a52a9089354	no	tap_sw1_mgmt tap_sw2_mgmt

To bring up two Nexus 9000v platforms, connecting one interface each back to back, you can use the following commands as examples. The connection can be a socket-based or bridge-based connection. In this example, bridges are used to connect instances of management interface and one data port. Similarly, more Nexus 9000v data ports can be connected in the same way by adding more net device in the command line options. In this example, two interfaces each (interface mgmt0 and interface eth1/1) on both the Nexus 9000v instances are mapped.

For a Nexus 9000v first instance:

```
sudo qemu-system-x86_64 -smp 2 -m 8196 -enable-kvm -bios bios.bin
-device i82801b11-bridge,id=dmi-pci-bridge
-device pci-bridge,id=bridge-1,chassis_nr=1,bus=dmi-pci-bridge
-device pci-bridge,id=bridge-2,chassis_nr=2,bus=dmi-pci-bridge
-device pci-bridge,id=bridge-3,chassis_nr=3,bus=dmi-pci-bridge
-device pci-bridge,id=bridge-4,chassis_nr=4,bus=dmi-pci-bridge
-device pci-bridge,id=bridge-5,chassis_nr=5,bus=dmi-pci-bridge
-device pci-bridge,id=bridge-6,chassis_nr=6,bus=dmi-pci-bridge
-device pci-bridge,id=bridge-7,chassis_nr=7,bus=dmi-pci-bridge
-netdev tap,ifname=tap_sw1_mgmt,script=no,downscript=no,id=eth1_1_0
-device e1000,bus=bridge-1,addr=1.0,netdev=eth1_1_0,mac=00:b0:b0:01:aa:bb,multifunction=on,romfile=
-netdev tap,ifname=tap_sw1_eth1_1,script=no,downscript=no,id=eth1_1_1
-device e1000,bus=bridge-1,addr=1.1,netdev=eth1_1_1,mac=00:b0:b0:01:01:01,multifunction=on,romfile=
-device ahci,id=ahci0 -drive file=test1.qcow2,if=none,id=drive-sata-disk0,id=drive-sata-disk0,format=qcow2
-device ide-hd,bus=ahci0.0,drive=drive-sata-disk0,id=drive-sata-disk0
-serial telnet:localhost:9000,server,nowait -M q35 -daemonize
```

For a Nexus 9000v second instance:

```
sudo qemu-system-x86_64 -smp 2 -m 8196 -enable-kvm -bios bios.bin
-device i82801b11-bridge,id=dmi-pci-bridge
-device pci-bridge,id=bridge-1,chassis_nr=1,bus=dmi-pci-bridge
-device pci-bridge,id=bridge-2,chassis_nr=2,bus=dmi-pci-bridge
-device pci-bridge,id=bridge-3,chassis_nr=3,bus=dmi-pci-bridge
-device pci-bridge,id=bridge-4,chassis_nr=4,bus=dmi-pci-bridge
```

```

-device pci-bridge,id=bridge-5,chassis_nr=5,bus=dmi-pci-bridge
-device pci-bridge,id=bridge-6,chassis_nr=6,bus=dmi-pci-bridge
-device pci-bridge,id=bridge-7,chassis_nr=7,bus=dmi-pci-bridge
-netdev tap,ifname=tap_sw2_mgmt,script=no,downscript=no,id=eth1_1_0
-device e1000,bus=bridge-1,addr=1.0,netdev=eth1_1_0,mac=00:b0:b0:02:aa:bb,multifunction=on,romfile=
-netdev tap,ifname=tap_sw2_eth1_1,script=no,downscript=no,id=eth1_1_1
-device e1000,bus=bridge-1,addr=1.1,netdev=eth1_1_1,mac=00:b0:b0:02:01:01,multifunction=on,romfile=
-device ahci,id=ahci0 -drive file=test2.qcow2,if=none,id=drive-sata-disk0,id=drive-sata-disk0,format=qcow2
-device ide-hd,bus=ahci0.0,drive=drive-sata-disk0,id=drive-sata-disk0
-serial telnet:localhost:9100,server,nowait -M q35 -daemonize

```

The `qemu-system-x86_64` or above KVM command is equivalent depending on how Linux is deployed. After successful invocation, you should be able to access both instances of the serial console via “telnet localhost 9000” or “telnet localhost 9100” respectively.

To pass traffic for LLDP and LACP multicast-specific packets through a Linux bridge, set the following values on all bridges connecting to each instance:

- Set LLDP and LACP communication between the VMs:
`echo 0x4004 > /sys/class/net/br_test/bridge/group_fwd_mask`
- Allow Multicast packet flow through the Linux bridge:
`echo 0 > /sys/devices/virtual/net/br_test/bridge/multicast_snooping`

Nexus 9000v Deployment Workflow for ESXi

This section describes the steps required to deploy Nexus 9000v platforms on ESXi hypervisors. Three types of deployment are available:

- Common Deployment
- Platform-Specific Deployment
- Interconnecting Deployment

Common Deployment Workflow

Before you begin

The following procedure provisions a Cisco Nexus 9300v or 9500v platform in the ESXi hypervisor using the distributed OVA.

Ensure the following:

- You have installed the ESXi 8.0 hypervisor
- You have a valid license for ESXi 8.0 to run on both server and vCenter.
- The distributed OVA file has been downloaded to the desktop.

-
- Step 1** Log into the ESXi vCenter.
- Step 2** Right-click version 8.0 and select **Deploy OVF Template**.
- Note** Perform the self-guided instructions in the subsequent screens that appear.
- Step 3** In the **Need name** screen, choose **Local file** and click **Browse**. Choose the downloaded distribute OVA file from your desktop.
- Step 4** In the **need name** screen, choose the data center (or a folder and enter the VM name).
- Step 5** In the **need name** screen, select an ESXi server for the Virtual Machine to be deployed into, and click **Finish** after the validation.
- Step 6** In the **need name** screen, review the details, and click **Next**.
- Step 7** In the **Configuration** screen click **Next**.
- Step 8** In the **Select Storage** screen, select the data store, and click **Next**.
- Step 9** In the **Select Networks** screen, ensure that the following values are selected:
- Source Network name - mgmt 0
 - Destination Network - lab management LAN vSwitch

Don't select other vNIC destinations as the lab management LAN vSwitch. Failure to do so results in management connectivity issues because the Cisco Nexus 9000v data ports will conflict with the physical switches.

- Step 10** In the **Ready to Complete** screen, click **Finish**, and wait for the completion of the process.
- Step 11** Under the **Virtual Hardware** tab, select **Serial Port 1**. For the serial port type, select the **Use Network** panel, and select the following options:
- Direction - Server
 - Port URL - telnet://0.0.0.0:1000, where 1000 is the unique port number in this server.

Note Nexus 9000v only supports E1000 network adapters. When you add any network adapter, verify that the adapter type is E1000.

- Step 12** Under the **VM Options** tab, select the **Boot Options** panel, and choose **EFI**.
- Step 13** Under the **VM Options** tab, select the **Advance** panel and in the **Edit Configuration** screen, add the following values using the **Add Configuration Params** option:
- Name - efi.serialconsole.enabled
 - Value - TRUE

Click **OK** to view the boot up process in both the VGA and the serial console mode.

Note Nexus 9000v platforms require the serial console to be provisioned in order to access the switch prompt (although some of the initial grub boot messages are shown on VGA console). Ensure that the serial console is provisioned on the VM correctly. Successful bootup should show kernel boot up messages after “**Image Signature verification for Nexus9000v is not performed**” is displayed from the VGA or serial console if “efi.serialconsole.enabled=TRUE” is provisioned.

Step 14 Power on the virtual machine.

Platform Specific Workflow

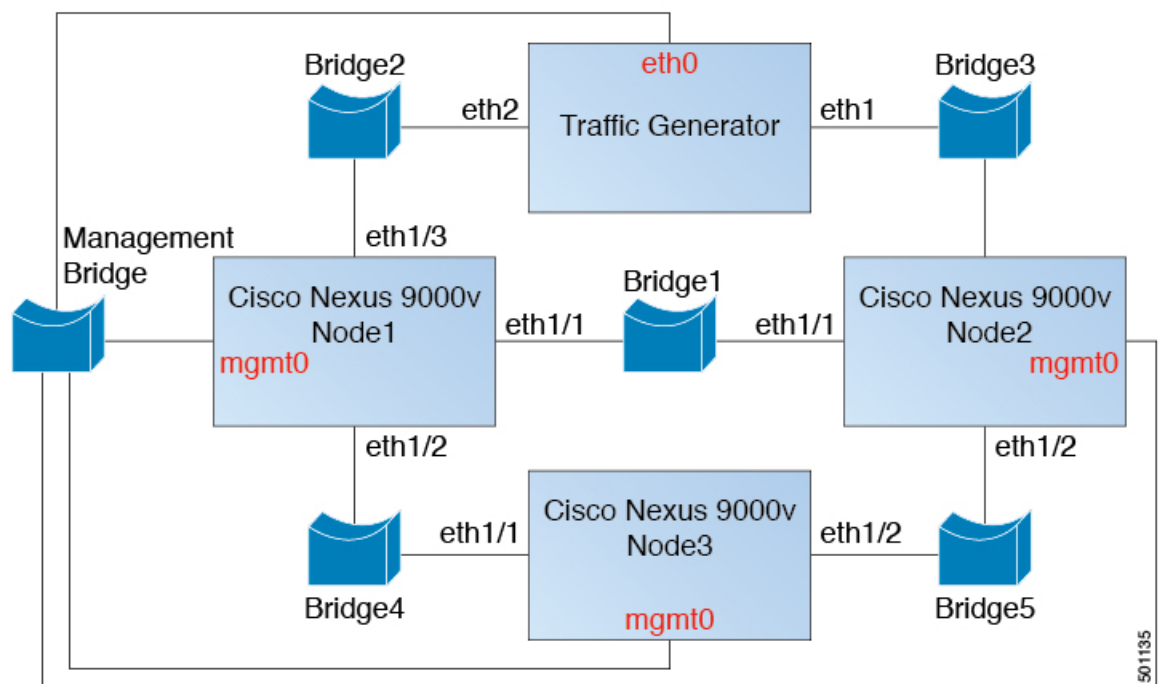
The Cisco Nexus 9500v runs in two different modes: sequential and mac-encoded mode. Nexus 9300v and Nexus 9500v sequential mode deployment steps are the exact same on ESXi hypervisor. The maximum number of interfaces for both platform types is 10 (one management port and nine data ports); this is a hypervisor limitation.

The Nexus 9500v emulates multiple-line-card interface traffic in single VM on ESXi hypervisor even though the total number of interfaces is limited to 10. If you choose to use the Nexus 9500v mac-encoded schema, change each network adapter MAC address to match slots and ports that are being emulated.

Interconnecting Platforms

Networking between Nexus 9300v and Nexus 9500v, or any other virtual platform, is based on vSwitch as the bridge on the ESXi hypervisor. You can have any topology as designed to simulate various customer use cases.

Figure 1: Interconnecting Cisco Nexus 9000v Platforms through ESXi



Nexus 9000v Deployment Workflow for Vagrant

This section describes the steps required to deploy Nexus 9000v platforms on Vagrant hypervisors. Three types of deployment are available:

- Common Deployment
- Platform-Specific Deployment
- Interconnecting Deployment

Common Deployment Workflow

You can't deploy the Cisco Nexus 9300v in the Vagrant/VBox environment. The virtual artifacts .box file is only available on distribution.

Platform Specific Workflow

Deploy the nexus9300v.9.3.3.IDI9.0.XXX.box on a VirtualBox. See the following customization guidelines and caveats for using Vagrant/Vbox:

- The user customization in Vagrant file isn't required.
- There's no need to change named pipe for Windows. Access the serial console using default port 2023, for both Mac or Windows. If needed, use this serial console via **telnet localhost 2023** to monitor the switch boot up process.
- The standard box process is used as any other appliance distribution. You can simply bring up a VM using the base box name.
- The box name can be changed to a different name other than "base" using the **config.vm.box** field from the Vagrant file.
- The bootstrap configuration is possible if you want to apply a different configuration on the switch, other than the existing generic configuration in **.box** from the release image file. In this case, use **vb.customize pre-boot**. For example:

```
vb.customize "pre-boot", [
  "storageattach", :id,
  "--storagectl", "SATA",
  "--port", "1",
  "--device", "0",
  "--type", "dvddrive",
  "--medium", "../common/nxosv_config.iso",
```

- Customize the VM interface MAC address by using the **config.vm.base_mac** field. This modification must be performed prior to entering the **vagrant up** CLI command and after entering the **vagrant init** CLI command. If you want to modify the MAC address after entering the **vagrant up** CLI command, or after the VM is created, use the box commands to modify the VM.

Support for Sync Folder in Vagrant

Starting with Release 10.1(1), Nexus 9300v supports Vagrant sync folder with which a directory/folder on a host machine can be shared with a Nexus 9300v machine. The **vagrant up** command in the Vagrant scripts logs into the virtual box and mounts the directory based on user configuration in the Vagrantfile. By default, the Vagrant scripts use the *vagrant* username, and expect bash to be the login shell. In order to facilitate this feature, the default login shell for pre-configured *vagrant* username has been changed to bash. However, you have the option to change the default shell (for user *vagrant*) to NX-OS CLI with explicit configuration in Nexus or in the Vagrantfile .

By default, Vagrant mounts the current working directory on the host at `directory/vagrant` on the Nexus 9300v. If you do not want the current folder on host to be shared with the Nexus 9300v, you must include the following line in the Vagrantfile.

```
config.vm.synced_folder ".", "/vagrant", disabled: true
```

Sample Vagrantfile - when you want to share the host folder, say, `/home/james/my_shared_folder/` on Nexus 9300v at `/bootflash/home/vagrant/`:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :
Vagrant.configure("2") do |config|
  # The most common configuration options are documented and commented below.
  # For a complete reference, please see the online documentation at
  # https://docs.vagrantup.com.

  # Every Vagrant development environment requires a box. You can search for
  # boxes at https://vagrantcloud.com/search.

  config.vm.define "n9kv1" do |n9kv1|

    n9kv1.vm.box = "10.1.1"

    n9kv1.ssh.insert_key = false
    n9kv1.vm.boot_timeout = 600

    if Vagrant.has_plugin?("vagrant-vbguest")
      config.vbguest.auto_update = false
    end

    config.vm.synced_folder ".", "/vagrant", disabled: true
    config.vm.synced_folder "/home/james/my_shared_folder" "/bootflash/home/vagrant/"
    config.vm.box_check_update = false

  end
end
```

Given below is Nexus 9300v platform-specific deployment example:

```
vagrant box add 10.1.1 nexus9300v.10.1.1.box

$ vagrant init 10.1.1
$ vagrant up

Bringing machine 'n9kv1' up with 'virtualbox' provider...
==> n9kv1: Importing base box '10.1.1'...
==> n9kv1: Matching MAC address for NAT networking...
==> n9kv1: Setting the name of the VM: vagrant_n9kv1_1605848223701_17342
==> n9kv1: Clearing any previously set network interfaces...
==> n9kv1: Preparing network interfaces based on configuration...
    n9kv1: Adapter 1: nat
==> n9kv1: Forwarding ports...
    n9kv1: 22 (guest) => 2222 (host) (adapter 1)
==> n9kv1: Booting VM...
==> n9kv1: Waiting for machine to boot. This may take a few minutes...
    n9kv1: SSH address: 127.0.0.1:2222
    n9kv1: SSH username: vagrant
    n9kv1: SSH auth method: private key
==> n9kv1: Machine booted and ready!
==> n9kv1: Checking for guest additions in VM...
    n9kv1: The guest additions on this VM do not match the installed version of
    n9kv1: VirtualBox! In most cases this is fine, but in rare cases it can
    n9kv1: prevent things such as shared folders from working properly. If you see
    n9kv1: shared folder errors, please make sure the guest additions within the
```

```

n9kv1: virtual machine match the version of VirtualBox you have installed on
n9kv1: your host and reload your VM.
n9kv1:
n9kv1: Guest Additions Version: 5.2.18 r123745
n9kv1: VirtualBox Version: 6.1
==> n9kv1: Mounting shared folders...
n9kv1: /bootflash/home/vagrant => /home/james/my_shared_folder

$ vagrant ssh

-bash-4.4$

```

Changing Default Shell to NX-OS CLI

When you need to login to NX-OS CLI, use one of these options:

- By manually executing the **vsh** command on bash prompt on every login.
- You may make use of a pre-packaged script in Nexus 9300v virtual box and execute it from Vagrantfile as shown below.

```

config.vm.synced_folder ".", "/vagrant", disabled: true
config.vm.synced_folder "/home/james/my_shared_folder" "/bootflash/home/vagrant/"

config.vm.box_check_update = false

config.vm.provision "shell", inline: "vsh -r /var/tmp/set_vsh_as_default.cmd"

```

- You may login with username *admin* instead of username *vagrant* (Username *vagrant* is used by default when you use the **vagrant ssh** command)

```
ssh -p 2222 admin@127.0.0.1
```

Using Ansible with Nexus 9300v

Vagrant is a generic orchestrator which supports configuration and management of boxes with various *provisioners* such as, Ansible, Shell scripts, Ruby scripts, Puppet, Chef, Docker, Salt etc.

Vagrant file may contain sections for one (or more) provisioners along with its configurations. An example for Ansible, is shown here.

```

n9kv1.vm.provision "ansible" do |ansible|
  ansible.playbook = "n9kv1.yml"
  ansible.compatibility_mode = "2.0"
end

```

These provisioners are automatically triggered every time a virtual box boots up or when triggered manually with the **vagrant provision** command or with the **vagrant provision --provision-wth** command. Provide login credentials in an Ansible host config file for Ansible to log into the virtual box and execute NX-OS CLIs. Since Ansible would expect to see NX-OS CLI after logging in, you can use the pre-configured username *admin* or create a new username manually, and use it in the Ansible host configuration files.

Shutdown VM

Use the following to shutdown the VM:

```

$ vagrant halt -f
==> default: Forcing shutdown of VM...

```

Destroy VM for cleanup

Use the following if you want to completely delete the VM instance:

```
$ vagrant box remove base
Removing box 'base' (v0) with provider 'virtualbox'...
$ vagrant destroy
    default: Are you sure you want to destroy the 'default' VM? [y/N] y
==> default: Destroying VM and associated drives..
```

Interconnecting Platforms

Networking between Nexus 9300v and other virtual platforms, is based on VBox Internal Network. See the following connection diagram:

Figure 2: Interconnecting Cisco Nexus 9000v Platforms through Vagrant VM

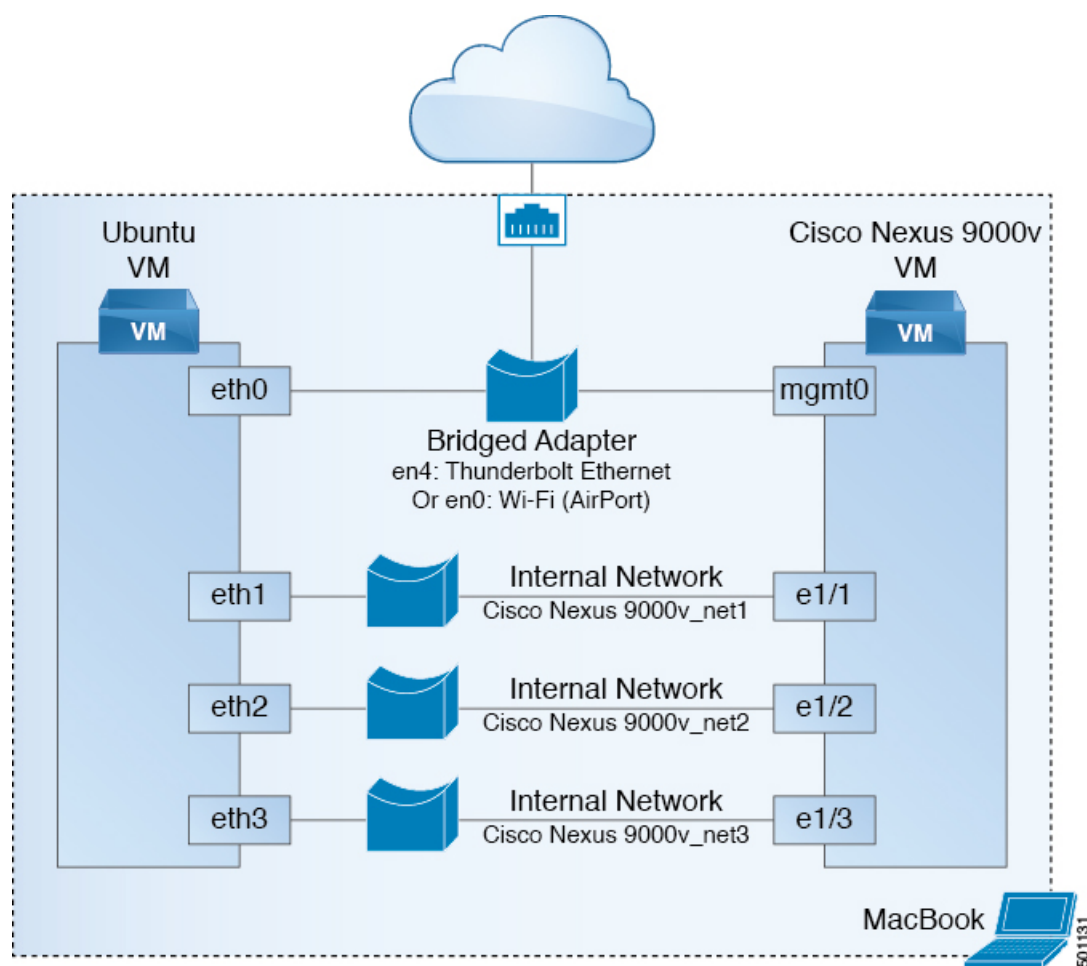


Image Upgrade Workflow

This section describes the typical upgrade steps for the Cisco Nexus 9000v platforms.

Deploying from a New Artifact

Depending on the environment, use the appropriate virtual artifact and refer to one of the following sections to deploy the VM:

- [Nexus 9000v Deployment Workflow for KVM/QEMU, on page 22](#)
- [Nexus 9000v Deployment Workflow for ESXi, on page 27](#)
- [Nexus 9000v Deployment Workflow for Vagrant, on page 29](#)

Upgrading from a New NX-OS Image

Nexus 9300v upgrades are only allowed from a VM created with virtual artifacts from Cisco Nexus 9000v, Release 9.3(1) and onwards. Before upgrading, ensure there's 400Mb + of new NX-OS binary image on the bootflash. To upgrade, copy the new binary to the bootflash and then upgrade using the standard NX-OS workflow (for example: 'install all nxos bootflash:///<nxos.bin>').

Nexus 9500v upgrades aren't supported as this is the first release of the platform.

For Nexus 9300v and 9500v lite, ISSU from earlier binary image to lite binary image is not supported. Even if you can bring up the image using cold boot, delete the previous configuration first, and then install the lite binary.



CHAPTER 4

Nexus 9300v and 9500v Lite NX-OS Image

- [About Nexus 9300v and 9500v Lite NX-OS Image, on page 35](#)
- [Features Supported, on page 36](#)
- [Nexus 9300v Lite and 9500v Lite NX-OS Image Deployment, on page 38](#)

About Nexus 9300v and 9500v Lite NX-OS Image

The NX-OS lite (nxos64-cs-lite.10.5.1.F.bin) image has a reduced memory footprint, resulting in a much smaller image size than the earlier images for Cisco Nexus 9300v and 9500v. This image can be installed during the boot-up time, and the image bring up is faster than the earlier images. However, before loading the lite image, perform a write-erase-reload, as some of the configurations may not be valid.



Note This image contains only base RPMs, which are needed to support deployment scenarios such as VXLAN EVPN, no F+L, with IPv4/IPv6 underlay (OSPF/IS-IS), and M-site functionality including vPC on BL, PC/vPC, LACP, and DME/YANG/telemetry. Rest of the RPMs are omitted from the lite NX-OS image.

Resource Requirements

The following table displays the resource requirements for Nexus 9300v and 9500v Lite:

Resource	Specification
Minimum RAM	6 G (basic bootup)
Recommended RAM	8 G (depending on the number of features)
Minimum vCPUs	2
Recommended vCPUs	4
Minimum vNICs	1
Maximum vNICs	65

Virtual Artifacts

The following table displays the virtual artifact(s) for Nexus 9300v and 9500v Lite:

VM Type	Hypervisor	Virtual Artifact	Description
N9300v Lite	ESXI 8.0	nexus9300v64-lite.10.5.1.Fova	Contains virtual disk, machine definition, and NXOS image.
N9300v Lite	KVM/QEMU 4.2.0	nexus9300v64-lite.10.5.1.Fqcow2	Contains virtual disk and NXOS image on bootflash.
N9300v Lite	Vagrant 2.3.7	nexus9300v64-lite.10.5.1.Fbox	Contains a preinstalled NXOS image on a virtual disk along with a machine definition.
N9500v Lite	ESXI 8.0	nexus9500v64-lite.10.5.1.Fova	Contains virtual disk, machine definition, and NXOS image.
N9500v Lite	KVM/QEMU 4.2.0	nexus9500v64-lite.10.5.1.Fqcow2	Contains virtual disk and NXOS image on bootflash.

Features Supported

The following sections provide information about the various categories of features supported by Nexus 9300v Lite and 9500v Lite platforms:

- [Layer 2 Features](#)
- [Layer 3 Features](#)
- [Programmability Features](#)

Layer 2 Features

The following table lists layer 2 feature support for the Nexus 9300v Lite and Nexus 9500v Lite platforms.

Feature	Nexus 9300v Lite Support	Nexus 9500v Lite Support
802.1AB LLDP	Yes	Yes
802.1Q VLANs/Trunk	Yes	Yes
802.1s RST	Yes	Yes
802.3ad LACP	Yes	Yes
L2 Multicast	Yes (as broadcast)	Yes (as broadcast)

Feature	Nexus 9300v Lite Support	Nexus 9500v Lite Support
MLAG	Yes	Yes
Port Channel	Yes	Yes
VLANs	Yes	Yes
VXLAN EVPN	Yes	Yes
No F+L with IPv4/Ipv6 underlay (OSPF/IS-IS)	Yes	Yes
Multi-Site functionality with non-vPC on BGW	Yes	Yes
HSRP	Yes	Yes
NGOAM	Yes	Yes
TACACS+	Yes	Yes

Layer 3 Features

The following table lists layer 3 feature support for the Nexus 9300v Lite and Nexus 9500v Lite platforms.

Feature	Nexus 9300v Lite Support	Nexus 9500v Lite Support
OSPF	Yes	Yes
OSPFv3	Yes	Yes
BGP	Yes	Yes
MP-BGP	Yes	Yes
IS-IS	Yes (as broadcast)	Yes (as broadcast)
Equal Cost Multipath Routing (ECMP)	Yes	Yes
PIM-SM	Yes	Yes
MPLS	Yes	Yes
CDP	Yes	Yes
L3 SVI	Yes	Yes
Sub Interfaces	Yes	Yes
IPsec	No	No

Programmability Features

The following table lists programmability feature support for the Nexus 9300v Lite and Nexus 9500v Lite platforms.

Feature	Nexus 9300v Lite Support	Nexus 9500v Lite Support
Bash shell access and scripting	Yes	Yes
RPM support	Yes	Yes
Programmatic access to system state (Python)	Yes	Yes
Docker within OS	Yes	Yes
NXAPI	Yes	Yes
DME	Yes	Yes
RESTCONF	Yes	Yes
NETCONF	Yes	Yes
YANG Models	Yes	Yes
Telemetry	Yes	Yes
GNMI	Yes	Yes
NxSDK	Yes	Yes

Nexus 9300v Lite and 9500v Lite NX-OS Image Deployment

For more information about the deployment procedures for KVM/QEMU, ESXi, and Vagrant, see the following:

- [Nexus 9000v Deployment Workflow for ESXi, on page 27](#)
- [Nexus 9000v Deployment Workflow for KVM/QEMU, on page 22](#)
- [Nexus 9000v Deployment Workflow for Vagrant, on page 29](#)



Note 9500v Lite is not supported on Vagrant. Additionally, ISSU is not supported for 9300v Lite and 9500v Lite. For more information, see [Image Upgrade Workflow, on page 33](#).



APPENDIX **A**

Troubleshooting the Cisco Nexus 9000v

This chapter contains the following sections:

- [Troubleshooting the Cisco Nexus 9000v Platform, on page 39](#)
- [Troubleshooting the Cisco Nexus 9000v Dataplane, on page 44](#)

Troubleshooting the Cisco Nexus 9000v Platform

General Troubleshooting/Debugging

The following CLI command provides troubleshooting help for both the Nexus 9300v and Nexus 9500v platforms:

show tech-support nexus9000v

The following is an example output of this command:

```
switch# show tech-support nexus9000v

----- Virtual Chassis Manager Debugs -----

#####
# /cmn/pss/virt_cmgr.log
#####
[19-12-10 20:42:34.160609]: virt_cmgr_startup_init called
[19-12-10 20:42:34.161351]: virt_cmgr_validate_file returned success
[19-12-10 20:42:34.161390]: Version 1, VNIC_scheme 2
[19-12-10 20:42:34.161404]: VM sup1: Module no 26, upg_version 1, type 1, card_i
ndex 0, image loc None
...
...
...
```

Common Issues for All Hypervisors

Boot when VM drops into "loader >" prompt

Generally, the initial boot is successful. However, the system boot could fail and drop into the "loader >" prompt on the VGA console or serial console, depending on how you provisioned the VM.

Example:

```

Loader Version 5.9

Loader > dir

bootflash::

  .rpmstore
  nxos.9.3.2.20.bin
  bootflash_sync_list
  .swtam
  eem_snapshots
  virtual-instance
  scripts
  platform-sdk.cmd

loader > boot nxos.9.3.2.20.bin

```

To continue the boot, enter the **boot nxos.9.3.2.20.bin** command at the "loader >" prompt

Prevent VM from dropping into "loader >" prompt

After you set up your Cisco Nexus 9000v (and following the set-up of the POAP interface), configure the boot image in your system to avoid dropping to the "loader >" prompt after reload/shut down.

Example:

```

nx-osv9000-2# config t
Enter configuration commands, one per line. End with CNTL/Z.
nx-osv9000-2(config)# boot nxos bootflash:nxos.9.3.2.20.bin
Performing image verification and compatibility check, please wait....
nx-osv9000-2(config)# copy running-config startup-config

```

Bootup Warning Message

During bootup, you may get a warning message similar to the following:

```

Checking all filesystems. **Warning** : Free memory available in bootflash is
553288 bytes
need at least 2 GB space for full image installation ,run df -h

```

This message generally indicates that the Nexus 9000v bootflash doesn't have enough memory space for holding another image. To eliminate this warning message, free up bootflash space to allow for the download of another binary image.

Nexus 9000v Mac-Encoded Mode Network Mapping Check

This check is only relevant if you explicitly enter the **platform vnic scheme mac-encoded** command on Nexus 9500v platform. This command enables the vNIC mac-encoded scheme. If any data traffic passes, or vNIC-mapped interfaces show the "Link not connected" state, refer to the Nexus 9000v informational show commands to verify correct vNIC mapping.

ESXi Hypervisor Issues

Nexus 9000v boot not seen after powering on the VM

The likely cause of this issue is that the EFI boot isn't set in the VM configuration. To resolve this issue, refer to the ESXi deployment guide to change "BIOS" to "EFI" in **Edit virtual machine settings > VM Options > Boot Options** after deployment using the distributed OVA virtual artifacts.

Bootup logs not seen after VGA output

A common problem during ESXi bootup is that the VGA console displays output similar to the following:

```
Sysconf checksum failed. Using default values
console (dumb)

Booting nxos.9.3.2.6.bin...
Booting nxos.9.3.2.bin
Trying diskboot
  Filesystem type is ext2fs, partition type 0x83
Image valid

Image Signature verification for Nexus9000v is not performed.

Boot Time: 12/5/2019 10:38:41
```

The issue is that, in the VGA console, there's no following activity in the bootup process. It's often misunderstood as a switch bootup process hang. To see the output of a switch bootup, connect to the provisioned serial console based on steps provided in the ESXi hypervisor deployment guide.

If nothing happens in the serial console, or you see the "telnet: Unable to connect to remote host: Connection refused" error message, it indicates one or more of the following issues:

- The serial console provisioning is incorrect in the VM configuration. Read and follow the instructions for serial console connectivity in the ESXi deployment guide.
- ESXi 8.0 deployment is the only version supported. Make sure that you have a valid license for ESXi vCenter and a valid UCS server license.
- Make sure that the "Security Profile" in the server has "VM serial port connected over network", both for incoming connections and outgoing connections.

No access to "loader>" prompt after powering down the VM

This issue occurs if you power on the VM and it boots up as expected, but the serial console wasn't correctly provisioned. Then the "config t; boot nxos bootflash:nxos.9.3.2.20.bin" configure is performed and saved. Powering up the VM again results in a drop to the VGA console.

The following recommendations help to avoid this issue in the ESXi hypervisor.

EFI BIOS defaults all input/output to the VM console. When a VM drops to the "loader >" prompt, go to the vSphere client or VGA console to access the "loader >" prompt to boot the image in the hard disk. You can change this behavior by adding an extra configuration in the ESXi VM editing mode. Use one of the following methods:

1. In the vSphere client Configuration Parameters window, add one row in the configuration (Edit Settings > VM Options > Advanced > Edit Configuration).
2. Add `efi.serialconsole.enabled = "TRUE"` to the `.vmx` file once the VM is created.

The vCenter or UCS server connectivity is lost as soon as the Cisco Nexus 9000v is up

**Caution**

When connecting a vNIC to a vSwitch or bridge, an incorrect network connection might result in losing the connectivity to your hypervisor server or vCenter on ESXi.

The Cisco Nexus 9000v uses vNICs entered from a graphical representation on ESXi for networking, either externally or internally within a hypervisor server. The first NIC is always used as the Cisco Nexus 9000v management interface.

The first NIC in the Cisco Nexus 9000v VM is the management interface. Connect it directly to your lab LAN physical switch or vSwitch (VM Network). Don't connect any data port vNIC to any physical switch conflicting with your server management connectivity.

Cisco Nexus 9000v data port isn't passing traffic in the ESXi server

To ensure a smooth operation, specific configuration settings on the vSwitch must be enabled:

- Ensure that all instances of the vSwitch connecting to the Cisco Nexus 9000v are in "Promiscuous Mode" = "Accept", and pointing to the UCS server. You can access this option through "Configuration > Properties > Edit" from the vSphere Client.
- Ensure that all instances of vSwitch pass through all VLANs. You can access this option through "Configuration > Properties > Edit" from the vSphere Client.

ESXi 8.0 hypervisor often defaults the network interfaces adapter to the "E1000E" type which isn't supported in the Nexus 9000v platform. After deployment, make sure that all Network adapter types are "E1000".

KVM/QEMU Hypervisor Issues

Understanding the KVM/QEMU command line options requires a basic Linux background. In order to deploy the Nexus 9000v in this hypervisor, follow the deployment instruction and pay attention to the following areas:

- Make sure that the user guide recommends bios.bin.
- If the command line supports multiple disk inputs, check that the bootable disk is set to bootindex=1 so that the VM doesn't try to boot from other devices.
- If you're attempting to implement a complicated command line, follow basic KVM/QEMU deployment instruction to bring up a simple switch instance first to verify the user environment.

Multicast on KVM or QEMU Hypervisor

The multicast feature on the Cisco Nexus 9000v is supported as broadcast. To make this feature to work properly, disable IGMP multicast snooping in this environment on all bridge interfaces.

The following example shows how to disable vxlan_br1, vxlan_br2, vxlan_br3, and vxlan_br4 from the linux prompt:

```
echo 0 > /sys/devices/virtual/net/vxlan_br1/bridge/multicast_snooping
echo 0 > /sys/devices/virtual/net/vxlan_br2/bridge/multicast_snooping
echo 0 > /sys/devices/virtual/net/vxlan_br3/bridge/multicast_snooping
echo 0 > /sys/devices/virtual/net/vxlan_br4/bridge/multicast_snooping
```

Follow the Linux bridge mask setup in the KVM/QEMU deployment guide, for passing L2 packets such as LLDP, LACP, and others.

Vagrant/VirtualBox Issues

Networking on VirtualBox/Vagrant

To use the dataplane interfaces on VirtualBox/Vagrant, ensure the following:

- The interfaces must be in "Promiscuous" mode.
- In the VirtualBox network settings, select "Allow All" for the Promiscuous mode.
- Ensure all instances of Cisco Nexus 9000v in your topology have unique MAC addresses by using the **show interface mac** command.

VM normal bootup on VirtualBox/Vagrant:

```
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Clearing any previously set forwarded ports...
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
      default: Adapter 1: nat
==> default: Forwarding ports...
      default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
      default: SSH address: 127.0.0.1:2222
      default: SSH username: vagrant
      default: SSH auth method: private key
```

The configured shell (config.ssh.shell) is invalid and unable to properly execute commands. The most common cause for this is using a shell that is unavailable on the system. Please verify you're using the full path to the shell and that the shell is executable by the SSH user.

The **vagrant ssh** command will access the Nexus 9000v switch prompt after the successful normal bootup.

The following is an example of one possible VM bootup failure:

```
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Importing base box 'base'...
==> default: Matching MAC address for NAT networking...
==> default: Setting the name of the VM: n9kv31_default_1575576865720_14975
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
      default: Adapter 1: nat
==> default: Forwarding ports...
      default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
      default: SSH address: 127.0.0.1:2222
      default: SSH username: vagrant
      default: SSH auth method: private key
```

Timed out while waiting for the machine to boot. This means that Vagrant was unable to communicate with the guest machine within the configured ("config.vm.boot_timeout" value) time period.

If you look above, you should be able to see the error(s) that Vagrant had when attempting to connect to the machine. These errors are usually good hints as to what may be wrong.

If you're using a custom box, make sure that networking is properly working and you're able to connect to the machine. It is a common problem that networking isn't setup properly in these boxes. Verify that authentication configurations are also setup properly,

as well.

If the box appears to be booting properly, you may want to increase the timeout ("config.vm.boot_timeout") value.

To troubleshoot this failure, check the following:

- Ensure that enough resources, such as memory and vCPU, are available. Close all applications that consume a significant amount of memory in your PC or server. Check the available free memory.
- Power down VM by entering **vagrant halt -f**
- Go to the VirtualBox GUI after powering down the VM. Enable the VM serial console to observe the boot up process and to view possible issues through "Ports" -> "Enable Serial Port".

Alternatively, use the following VBox command to enable this guest serial console. Find your VM name:

```
VBoxManage list vms
    "n9kv_default_1575906706055_2646" {0b3480af-b9ac-47a4-9989-2f5e3bdf263f}
```

Then enable serial console:

```
VBoxManage modifyvm n9kv_default_1575906706055_2646 --uart1 0x3F8 4
```

- Power up the VM again by entering “vagrant up” from the same terminal, where you entered the original “vagrant up”.
- To access the serial console, enter “telnet localhost 2023” from another terminal on your computer.
- Check the bootup issue by observing the output from the serial console.
- Turn off the serial console if the guest serial console is no longer needed. Either use the following VBox command or go to the VirtualBox GUI setting and de-select “Enable Serial Port”.

```
VBoxManage modifyvm n9kv_default_1575906706055_2646 --uart1 off
```

Troubleshooting the Cisco Nexus 9000v Dataplane

The debug and show commands in this section are available to troubleshoot both Cisco Nexus 9300v and Cisco Nexus 9500v platforms. These commands must be executed on the line card/module.

Debug Commands

- **debug l2fwder event**
- **debug l2fwder error**
- **debug l2fwder fdb**
- **debug l2fwder pkttrace**

To run any of these commands, attach to the line card by following this example:

```
switch# sh mod | inc Mod
Mod Ports      Module-Type      Model      Status
1    64    Nexus 9000v 64 port Ethernet Module  N9K-X9364v  ok
27   0    Virtual Supervisor Module  N9K-vSUP    active *
```

Mod	Sw	Hw	Slot	Serial-Num
Mod	MAC-Address (es)			

```

Mod Online Diag Status
switch# attach mod 1
Attaching to module 1 ...
To exit type 'exit', to abort type '$.'
module-1# debug l2fwder ?
  error      Configure debugging of l2fwder control and data path errors
  event      Configure debugging of l2fwder events over ipc
  fdb        Configure debugging of l2fwder events over fdb
  ha         Configure debugging of l2fwder events from sysmgr
  logfile    Enable file logging to /logflash/l2fwder.debug
  packet     Configure debugging of l2fwder packet forwarding information
  pkttrace   Configure debugging of l2fwder packet trace

module-1# debug l2fwder
    
```

Event History Commands

- **show system internal l2fwder event-history events**
- **show system internal l2fwder event-history errors**
- **show system internal l2fwder event-history fdb**

Show Commands

show system internal l2fwder table bd

```

v-switch# show system internal l2fwder table bd

vlan 1 member 3, 4, 5, untagged 3, 4, 5, STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan no
vlan 80 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan yes
vlan 90 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan yes
vlan 110 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan yes
vlan 210 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan yes
vlan 310 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan yes
vlan 410 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan yes
vlan 510 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan yes
vlan 550 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan no
vlan 560 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan no
vlan 610 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan yes
vlan 650 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan no
vlan 660 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan no
vlan 710 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan yes
vlan 810 member 3, 4, 5, untagged none STP ports 3, 4, 5, dis none blk_lis none
lrn none fwd 3, 4, 5, tid 1, 2, vxlan yes
    
```

show system internal l2fwder table if

v-switch# show system internal l2fwder table if

If_name	If_index	gport	fd	untagged	vlanid	Trunk	SVP Info	Native vlan
Ethernet1/1	0x1a000000	0x8000801	14	1	4095	0x0	none	4095
Ethernet1/2	0x1a000200	0x8000802	15	1	4095	0x0	none	4095
Ethernet1/3	0x1a000400	0x8000803	16	0	4045	0x1	none	40451
Ethernet1/4	0x1a000600	0x8000804	17	0	810	0x2	none	810
Ethernet1/5	0x1a000800	0x8000805	18	0	810	0x0	none	810
Ethernet1/6	0x1a000a00	0x8000806	0	1	4095	0x0	none	4095
Ethernet1/7	0x1a000c00	0x8000807	0	1	4095	0x0	none	4095
Ethernet1/8	0x1a000e00	0x8000808	0	1	4095	0x0	none	4095
Ethernet1/9	0x1a001000	0x8000809	0	1	4095	0x0	none	4095
Ethernet1/10	0x1a001200	0x800080a	0	1	4095	0x0	none	4095
Ethernet1/11	0x1a001400	0x800080b	0	1	4095	0x0	none	4095

show system internal l2fwder table port-channel

v-switch# show system internal l2fwder table port-channel

Port-channel	Count	Member-list
0x1	1	0x8002004
0x4	2	0x8005001 0x8000805
0x5	2	0x8002001 0x8000801

Port-channel	Count	Local member-list6
0x1	0	
0x4	1	0x8000805
0x5	1	0x8000801

show system internal l2fwder table vxlan peer

v-switch# show system internal l2fwder table vxlan peer

```

VXLAN Tunnel:
  src_ip: 6.6.6.6, Is VxLAN enabled = TRUE
  multisite: no, nve_tun_dci_sip: 0.0.0.0
VXLAN PEER: No of tunnels = 7
  peer_ip: 224.1.1.2, vxlan_port_id: 0x0,
    tunnel_id: 0x4c000000, is_dp: 0 is_dci: 0
  peer_ip: 224.1.1.4, vxlan_port_id: 0x0,
    tunnel_id: 0x4c000002, is_dp: 0 is_dci: 0
  peer_ip: 224.1.1.6, vxlan_port_id: 0x0,
    tunnel_id: 0x4c000004, is_dp: 0 is_dci: 0
  peer_ip: 224.1.1.8, vxlan_port_id: 0x0,
    tunnel_id: 0x4c000006, is_dp: 0 is_dci: 0
  peer_ip: 224.1.1.9, vxlan_port_id: 0x0,
    tunnel_id: 0x4c000008, is_dp: 0 is_dci: 0
  peer_ip: 224.1.1.10, vxlan_port_id: 0x0,
    tunnel_id: 0x4c00000a, is_dp: 0 is_dci: 0
  peer_ip: 6.5.5.5, vxlan_port_id: 0x80002db8,
    tunnel_id: 0x4c00050a, is_dp: 0 is_dci: 0
Tunnel_id entry:
  peer_ip: 224.1.1.2, tunnel_id: 0x4c000000
  peer_ip: 224.1.1.4, tunnel_id: 0x4c000002
  peer_ip: 224.1.1.6, tunnel_id: 0x4c000004
  peer_ip: 224.1.1.8, tunnel_id: 0x4c000006
  peer_ip: 224.1.1.9, tunnel_id: 0x4c000008
  peer_ip: 224.1.1.10, tunnel_id: 0x4c00000a
  peer_ip: 6.5.5.5, tunnel_id: 0x4c00050a
    
```



```
Vxlan_gport ucast-entry:
  peer_ip: 6.5.5.5, vxlan_port_id: 0x80002db8
```

show system internal l2fwder table vxlan vni

```
v-switch# show system internal l2fwder table vxlan vni
```

VNI	VLAN	DF
81000	810	no
51000	510	no
5001	1001	no
5002	1002	no
5003	1003	no
5004	1004	no
21000	210	no
71000	710	no
9000	90	no
41000	410	no
11000	110	no
61000	610	no
31000	310	no

show system internal l2fwder acl info

```
v-switch# show system internal l2fwder acl info
```

Inactive List:

Entry ID: 14596 Qualify: DstTrunk 4, Action: RedirectTrunk 5 Prio: 4

Active List:

Inactive List:

Active List:

Entry ID: 15873 Qualify: EtherType ARP ForwardingVlanId 110, 610, 710, 1001, 1003, Action: CopyToCpu SET Drop SET Prio: 1

show system internal l2fwder mac

```
v-switch# show system internal l2fwder mac
```

Legend:

- * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
- + - primary entry using vPC Peer-Link,
- (T) - True, (F) - False, C - ControlPlane MAC

VLAN	MAC Address	Type	Secu	NTF	Del	Ports	Station_id
* 1	008b.860d.1b08	static	F	F	0	0xc000005	0
G -	008b.860d.1b08	static	F	F	0	sup-eth1 (R)	508,
* 210	0000.4545.6767	dynamic	F	F	0	0xc000004	0
G 710	008b.bc90.1b08	static	F	F	0	sup-eth1 (R)	0
G 310	008b.bc90.1b08	static	F	F	0	sup-eth1 (R)	0
G -	0002:0002:0002	static	F	F	0	sup-eth1 (R)	1,
* 210	008b.860d.1b08	static	F	F	0	0xc000005	0
G 410	008b.bc90.1b08	static	F	F	0	sup-eth1 (R)	0
* 1003	008b.2b34.1b08	dynamic	F	F	1	nve (0x80002db9)	0
* 1002	008b.2b34.1b08	dynamic	F	F	1	nve (0x80002db9)	0
* 1001	008b.2b34.1b08	dynamic	F	F	1	nve (0x80002db9)	0

```
* 1004 008b.2b34.1b08 dynamic F F 1 nve(0x80002db9) 0
* 810 008b.860d.1b08 static F F 0 0xc000005 0
G 510 008b.bc90.1b08 static F F 0 sup-eth1(R) 0
* 610 008b.2b34.1b08 dynamic F F 1 nve(0x80002db9) 0
G 1 008b.bc90.1b08 static F F 0 sup-eth1(R) 0
G - 008b:bc90:1b08 static F F 0 sup-eth1(R) 511,
```

show system internal l2fwder port egress info

```
v-switch# show system internal l2fwder port egress info
```

```
Ingress port :          Blocked egress ports
+-----+-----+
0x8002001      1          5
0x8000801      1          5
0x8020821      1          5
```

show system internal l2fwder vpc info

```
v-switch# show system internal l2fwder vpc info

VPC role : Primary
```

Packet Capture Commands

The Cisco Nexus N9000v supports Ethalyzer similarly to the standalone Nexus 9000 hardware switch.